République Algérienne Démocratique et Populaire
Ministère de L'Enseignement Supérieur et de la Recherche Scientifique
Université Saad Dahleb – Blida 1
Faculté des Sciences
Département d'informatique

Mémoire

Présenté pour l'obtention du **diplôme** de **MASTER**

En : Informatique

Spécialité : Ingénierie du Logiciel

Réalisé par : MERDAOUI Mohamed Islam Jugurta

Sujet

Conception et Réalisation d'un Système d'Aide à la Décision basé sur un Data Lake : Application à l'Aquaculture Marine en Algérie

Soutenu publiquement, le 02/07/2023, devant le jury composé de :

Mme.	H. ABED	Professeur	à l'U de Blida 1	Présidente
M.	M. BALA	Maître de Conférences B	à l'U de Blida 1	Promoteur
Mme.	K. SEMAR-BITAH	Maître de Recherche B	au CDTA	Encadrante
Mme.	S. BOUDRAA	Maître de Conférences B	à l'U de Blida 1	Examinatrice

Année universitaire
2022 – 2023

The higher the ignorance, the poorer the decisions
— Islam Merdaoui

Remerciements

Je tiens tout d'abord à exprimer ma gratitude à Allah, Le Miséricordieux et Le Tout-Puissant, pour m'avoir donné le courage et la force de mener à bien ce travail.

Je souhaite exprimer ma reconnaissance envers ma mère. Sa présence constante, ses encouragements et ses sacrifices ont été la raison de ma réussite. Sans son soutien inconditionnel, je n'aurais jamais pu terminer ce projet.

Je tiens à remercier sincèrement mon promoteur M. Bala Mahfoud pour ses précieux conseils et pour avoir partagé son expérience et son expertise. J'apprécie son dévouement à mon égard et j'admire son encadrement.

Je suis également reconnaissant envers mon encadreur Mme. Semar-Bitah Kahina pour m'avoir proposée ce sujet. Sa confiance en mes capacités et son soutien m'ont encouragé à me surpasser. Son encadrement attentif lors d'un stage précédent a été très constructif. Sans elle, je n'aurais jamais eu l'opportunité d'explorer un sujet aussi intéressant.

Je tiens à exprimer ma sincère gratitude au Chef de Département M. Milla Toufik pour son accueil chaleureux, ainsi qu'au Chef de Service Mme. Bougrid Dalila pour ses précieux enseignements qui m'ont éclairé sur les aspects techniques de l'aquaculture.

Mes remerciements vont également aux membres du jury, Mme. Abed Hafida et Mme. Boudraa Sawsen, pour avoir consacré leur temps et leur expertise à évaluer mon travail.

Sans oublier de remercier tous les membres de ma famille, mon père, mes sœurs et mes grands-parents, Jida Louisa et Jeddis Yahia, qui ont toujours été prêts à se sacrifier pour mon succès.

Résumé

Ce projet a été proposé dans le cadre d'une collaboration entre le Centre de Développement des Technologies Avancées (CDTA) et le Centre National de Recherche et de Développement de la Pêche et de l'Aquaculture (CNRDPA). Disposant d'une base de données relationnelle contenant des informations sur l'aquaculture marine, le CNRDPA souhaite intégrer ses données sur l'aquaculture continentale en Algérie. Cependant, les différences structurelles entre les deux types d'activités ne permettent pas sa réalisation. Ils souhaitent également introduire des données hétérogènes (Excel, Word, PDF, etc.) provenant de différentes sources à l'échelle nationale (14 DPA (Directions de la Pêche et de l'Aquaculture) parmi lesquelles figure le CNRDPA).

Comme réponse à cette problématique, une approche basée sur un Data Lake a été proposée. Une architecture fonctionnelle ainsi qu'une architecture technique a été conçue pour répondre à leurs besoins. Notre approche comprend un nouveau schéma d'ingestion, un processus d'ingestion en temps réel, une modélisation orientée objet des données brutes et leur migration vers un modèle orienté document. Un processus de capture et de transformation des données a également été développé, ainsi qu'une solution de visualisation et de restitution des données. Autrement dit, tout un système d'aide à la décision basé sur un Data Lake pour l'exploitation et l'analyse des données brutes et hétérogènes.

Mots Clés : Lac de données, Système d'Aide à la Décision, Aquaculture.

Abstract

This project was proposed as part of a collaboration between the Center for Development of Advanced Technologies (CDTA) and the National Center for Research and Development of Fisheries and Aquaculture (CNRDPA). With a relational database containing information about marine aquaculture, CNRDPA aims to integrate its data on inland aquaculture in Algeria. However, the structural differences between the two types of activities prevent its implementation. They also want to incorporate heterogeneous data (Excel, Word, PDF, etc.) from various sources at the national level (14 Fisheries and Aquaculture Directorates (DPA) among which the CNRDPA is included).

In response to this challenge, an approach based on a Data Lake was proposed. Both a functional architecture and a technical architecture were designed to meet their needs. Our approach includes a new ingestion schema, a real-time ingestion process, an object-oriented modeling of raw data, and their migration to a document-oriented model. A data capture and transformation process was also developed, along with a solution for data visualization and presentation. In other words, a comprehensive decision support system based on a Data Lake for the exploitation and analysis of raw and heterogeneous data.

Keywords: Data Lake, Decision Support System, Aquaculture.

ملخص

تم اقتراح هذا المشروع في إطار التعاون بين مركز تنمية التكنولوجيات المتطورة (CDTA) والمركز الوطني للبحث و التنمية في الصيد البحري و تربية المائيات (CNRDPA). حيث يملك CNRDPA قاعدة بيانات علاقية تحتوي على معلومات عن الزراعة البحرية، ويرغب في دمج بياناته حول الزراعة القارية للأسماك في الجزائر. ومع ذلك، لا تسمح الاختلافات الهيكلية بين النشاطين بتحقيق ذلك. كما يرغبون في إدخال بيانات متنوعة (ملفات Excel و Word و PDF وما إلى ذلك) من مصادر مختلفة (14 مديرية للصيد والاستزراع السمكي (DPA) ومن بينها CNRDPA).

للإجابة لهذه الإشكالية، تم اقتراح منهجية قائمة على بحيرة البيانات (Data Lake). تم تصميم هيكل وظيفي وهيكل تقني لتلبية احتياجاتهم. يتضمن نهجنا مخطط استيعاب جديد، وعملية استيعاب في الوقت الحقيقي، ونمذجة موجهة نحو الكائنات للبيانات الخام وترحيلها إلى نموذج موجه نحو الوثائق. تم أيضاً تطوير عملية لالتقاط وتحويل البيانات، بالإضافة إلى حل لتصور البيانات واستعادتها. يعد هذا نظام متكامل يعتمد على بحيرة البيانات للاستفادة وتحليل البيانات الخامة والمتنوعة لدعم عمليات اتخاذ القرار.

الكلمات المفتاحية: بحيرة البيانات، نظام المساعدة في اتخاذ القرارات، الاستزراع السمكي.

Table des matières

Table des figures

Liste des tableaux

Nomenclature

Acronymes

BSON	Binary JSON
CDTA	Centre de Développement des Technologies Avancées
CNRDPA	Centre National de Recherche et de Développement de la Pêche et de l'Aquaculture
DL	Data Lake
DW	Data Warehouse
DPA	Directions de la Pêche et de l'Aquaculture
ECLT	Extract Classify Load Transform
ELT	Extract Load Transform
ETL	Extract Transform Load
HDFS	Hadoop Distributed File System
IoT	Internet of Things
JSON	JavaScript Object Notation
KPI	Key Performance Indicators
MQTT	Message Queue Telemetry Transport
NoSQL	Not Only Structured Query Language
SAD	Système d'Aide à la Décision
SID	Systèmes d'Information Décisionnels
ZAA	Zone Allouée à l'Aquaculture

Autres Symboles

N/A	Non Applicable

Introduction Générale

Contexte

En 2010, quelques années après l'apparition des données massives, un nouveau type de stockage a vu le jour [6], il s'agit des Data Lakes.

Contrairement aux entrepôts de données, ils ont apporté aux entreprises la facilité et l'efficacité du traitement des données non structurées. En effet, le lac de données se caractérise principalement par sa capacité de stocker des mégadonnées à l'état brut et de divers formats. La combinaison entre le Data Lake et les systèmes décisionnels est une approche stratégique et complémentaire. Elle permet aux entreprises d'exploiter pleinement le potentiel de leurs données afin de faciliter la prise de décision.

Problématique

Le Centre de Développement des Technologies Avancées (CDTA) en collaboration avec le Centre National de Recherche et de Développement de la Pêche et de l'Aquaculture (CNRDPA) nous ont proposé un sujet qui s'intitule « Conception et réalisation d'un Système d'Aide à la Décision basé sur un Data Lake : Application à l'aquaculture marine en Algérie ».

Actuellement, le CNRDPA dispose d'une base de données relationnelle contenant des informations sur l'aquaculture marine. Cependant, ils souhaitent également introduire des données sur l'aquaculture continentale ainsi que des données hétérogènes (Excel, Word, PDF, etc.) provenant de différentes sources (CNRDPA et 13 DPA à l'échelle nationale). Malheureusement, les rubriques et les structures des données diffèrent entre les deux types d'activités, rendant impossible une simple combinaison des bases de données existantes.

Au lieu de créer une nouvelle base de données relationnelle, l'équipe du CNRDPA souhaite trouver une solution alternative. Ils envisagent donc un système capable de saisir automatiquement les données à partir de leurs sources brutes, en tenant compte des différences structurelles spécifiques à chaque type d'activité. Cette approche permettrait une gestion unifiée des informations tout en évitant la complexité d'une nouvelle base de données relationnelle.

À partir de là, nous nous sommes posé les questions suivantes :

— Quel référentiel de stockage serait la solution à la problématique du CNRDPA ?
— Quel type de base de données serait la plus appropriée pour combiner les données de l'aquaculture marine et continentale, compte tenu des différences de structure entre les deux types d'activités ?
— Quel type d'architecture conviendrait le mieux pour la mise en place du système de gestion des données combinées de l'aquaculture marine et continentale ?
— Est-il essentiel de mettre en œuvre un système de surveillance en temps réel des données environnementales dans le domaine aquacole ?

Objectifs

Afin de répondre à toutes ces questions, nous avons planifié la réalisation des étapes suivantes :

— Conception d'une architecture fonctionnelle pour notre système ;
— Proposition d'un nouveau schéma d'ingestion de données : ECLT (Extract Classify Load Transform) ;
— Conception d'un processus d'ingestion en temps réel ;
— Modélisation orientée objet des données brutes ;
— Migration du modèle orienté objet vers un modèle orienté document ;
— Proposition d'un processus de capture et de transformation de données ;
— Mise en place d'une solution de visualisation et de restitution de données ;

Organisation du mémoire

Ce mémoire est organisé en quatre chapitres :

— **Chapitre 1** : Nous l'avons consacré aux fondamentaux de l'informatique décisionnelle où nous définissons ses différents composants ainsi qu'aux concepts liés aux Big Data.
— **Chapitre 2** : Nous avons examiné les travaux connexes réalisés dans divers domaines d'application des Data Lakes, mettant en lumière cinq projets pertinents. Ces projets se distinguent par leurs architectures et les outils utilisés pour la gestion des données.
— **Chapitre 3** : Offre une vue d'ensemble sur l'architecture fonctionnelle de notre système.
— **Chapitre 4** : Ce chapitre est consacré à la description de l'architecture technique du système. Nous exposons les différents modules de traitement et concluons par la présentation de l'interface web.

Chapitre 1

Informatique décisionnelle et données massives

1.1 Introduction

Nous présentons, dans ce chapitre, une vue d'ensemble sur l'informatique décisionnelle et plus particulièrement les principaux modes de stockage : les entrepôts de données et les Data Lakes. Une étude comparative permettra par la suite de mettre en relief les principes des deux référentiels de stockage.

Nous introduisons également la notion de données massives (Big Data) et leurs fameuses caractéristiques. Enfin, nous abordons l'aspect de la prise de décision dans un environnement Data Lake.

1.2 Les systèmes d'information décisionnels

Les entreprises sont perpétuellement confrontées à différentes situations et problématiques, et par conséquent, à des choix entre différentes options ou voies possibles pour les résoudre.

La plupart de leurs données sont massives, hétérogènes, diffuses dans divers services et départements. La prise de décision stratégique nécessite l'accès à toutes ces données pour pouvoir les croiser en vue d'une exploitation efficace et pertinente [7].

Ainsi, la majorité des entreprises considèrent les systèmes d'information décisionnels (SID) comme étant l'ensemble des processus et outils de prise de décision indispensables pour atteindre leurs objectifs. A. Gorry et M. Scott Morton [8] ont défini le système d'aide à la décision (SAD) comme un « *système informatisé interactif aidant le décideur à manipuler des données et des modèles pour résoudre des problèmes mal structurés* ».

Le principe du système décisionnel est d'offrir un ensemble de méthodes et de techniques qui permettent aux décideurs de collecter, modéliser et restituer les données de leur entreprise, afin de les aider à prendre des décisions stratégiques. Pour ce faire, il est impératif que le SID permette l'extraction de données, leur stockage et leur restitution sous une forme exploitable [7].

1.3 Les entrepôts de données (Data Warehouses)

Un entrepôt de données, en anglais Data Warehouse (DW), est une base de données, ou un ensemble de bases de données, qui centralise les informations d'une entreprise issues de plusieurs et diverses sources internes ou externes, avec l'objectif de les stocker, les gérer et les rendre disponibles pour une utilisation et une analyse exploratoire [9].

Selon la définition d'Inmon [10] « A data warehouse is a subject-oriented, nonvolatile, integrated, time-variant collection of data in support of management decisions » c'est-à-dire : « un entrepôt de données est une collection de données orientée sujet, non volatile, intégrée et variant dans le temps à l'appui des décisions de gestion ».

En résumé, un Data Warehouse est caractérisé essentiellement par les principes suivants :

- orientée sujet : signifie que les données sont structurées et organisées conformément aux besoins et aux intérêts des décideurs ;
- non volatile : les données ne sont pas modifiées ou supprimées, elles sont conservées ;
- intégrée : désigne le processus de rassembler des sources de données hétérogènes vers une destination unifiée et homogène ;
- variant dans le temps : les données sont historisées pour suivre leur évolution dans le temps.

En d'autres termes, un DW est conçu pour permettre d'analyser des données structurées issues de diverses sources hétérogènes [9].

1.4 Les données massives (Big Data) et les lacs de données (Data Lakes)

Big Data

Le terme "Big Data" ou "données massives" fait référence à des volumes considérables d'informations provenant de diverses sources et générées à une échelle et à une vitesse inhabituelle. Le concept des sept « V » : Volume, Vélocité, Variété, Variabilité, Véracité,

Visualisation et Valeur constituent autant de dimensions de complexité qui caractérisent le Big Data [11].

Parmi ces sept dimensions de complexité, trois d'entre elles jouent un rôle clé dans sa définition [12] :

- Le volume fait référence à la quantité de données générées et collectées, pouvant atteindre des unités importantes telles que péta-octets, exa-octets voire plus ;
- La vélocité fait référence à la vitesse avec laquelle les données sont générées, traitées et analysées. Avec la prolifération des objets connectés (IoT : Internet of Things) et des systèmes en temps réel, les données peuvent être générées à un rythme très rapide ;
- La variété représente la diversité des types de données, qui peuvent provenir de différentes sources, telles que du texte, des images, des vidéos, des fichiers audio, des données de capteurs, etc.

La compréhension et l'application de ces concepts sont essentiels pour exploiter le potentiel des données massives et en tirer des avantages significatifs. Cependant, il existe des défis liés à la gestion et à l'analyse de celles-ci qui peuvent être résumés comme suit :

- Pour le volume : gérer la croissance exponentielle du volume de données requiert des infrastructures et des systèmes adaptés ;
- La vélocité : traiter les données en temps réel pour prendre des décisions dans des délais serrés est essentiel ;
- La variété : intégrer et gérer des données provenant de sources diverses, structurées et non structurées, avec des formats et des structures variables ;
- La véracité : assurer l'exactitude et la fiabilité des données provenant de multiples sources grâce à la validation et à la vérification des données.

En surmontant ces défis, les organisations peuvent exploiter les opportunités offertes par les données massives. Cela leur permettra de prendre des décisions éclairées et de renforcer leur avantage concurrentiel.

Data Lake

Le Data Lake, Lac de données en français, est un espace de stockage de données basé sur une architecture plate qui permet d'entreposer des données brutes de divers formats et en grande quantité [13].

La puissance des Data Lakes réside dans le fait que les données peuvent être de toutes natures, structurées (lignes et colonnes), non-structurées (emails, documents, PDF) ou semi-structurées, comme les formats CSV, XML et JSON et provenir de diverses sources [14]. L'exploitation du contenu du Data Lake est très large et peut intéresser

divers professionnels tels que les data scientists, les analystes de données, les chercheurs et les experts en intelligence artificielle.

Le Data Lake représente une solution de gestion de données hybrides et variées avec comme objectif de stocker de manière rapide et peu coûteuse une grande quantité de données brutes [15]. La principale qualité du Data Lake réside dans sa flexibilité. En effet, il permet le stockage de données sans prétraitement de celles-ci, mais également indépendamment de leur utilisation future. [16].

Architecture d'un Data Lake

L'architecture d'un lac de données est organisée en couches (également appelées zones) comme le représente la figure 1.1.

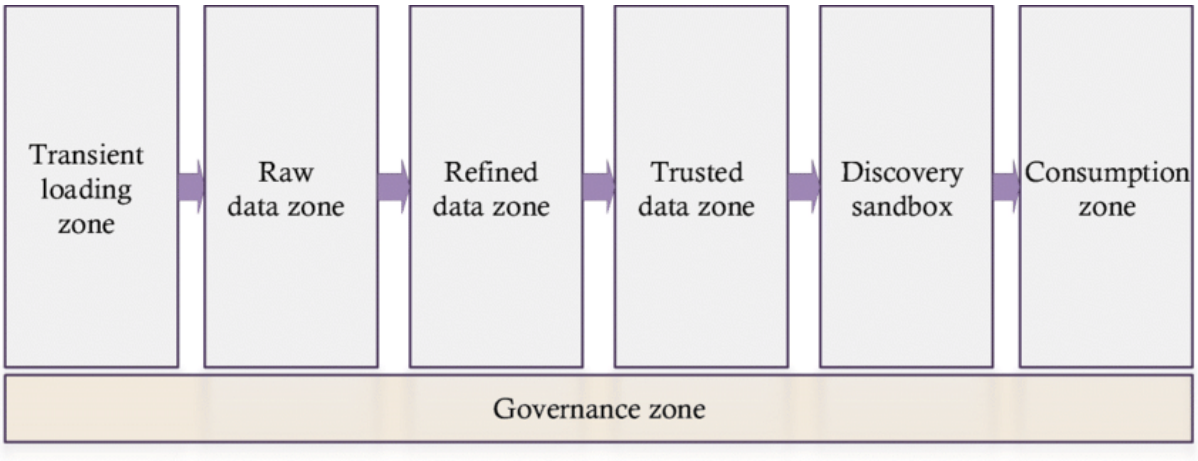


Figure 1.1: L'architecture en couches d'un Data Lake [6]

Voici une explication des différentes zones présentes dans l'architecture d'un Data Lake [17] :

1. **Zone Brute (Raw Zone)** : Son principal objectif est d'ingérer les données brutes le plus rapidement et le plus efficacement possible sans effectuer de transformations. Ces dernières ne sont pas prêtes à l'emploi, elles nécessitent des traitements ultérieurs afin de les exploiter.
2. **Zone Conforme (Trusted Zone)** : Cette couche est considérée comme l'une des parties les plus complexes où les données subissent du nettoyage, de la transformation, de la dénormalisation et de la consolidation.
3. **Zone de Raffinement (Refined Zone)** : Dans cette zone, les données subissent les dernières transformations avant d'être utilisées pour répondre aux besoins de plusieurs départements d'activité d'une entreprise. Les données sont consommables

et stockées dans des fichiers ou des tables et organisées par objectif, type et structure de fichier.

4. **Couche Sandbox** : Les données peuvent être importées dans la Sandbox à partir de n'importe quelle zone. Cette couche est destinée aux analystes et scientifiques de données intéressés par effectuer des expériences et rechercher des modèles ou des corrélations.

Il est important de souligner que d'autres couches supplémentaires peuvent être présentes. Ces zones varient en fonction des besoins spécifiques et des exigences propres à chaque entreprise.

1.5 Intégration de données

Deux approches sont utilisées pour gérer et traiter des données dans un environnement informatique : ETL (Extract Transform Load) et ELT (Extract Load Transform).

1.5.1 ETL (Extraction, Transformation, Chargement)

L'ETL est une méthode traditionnelle très largement utilisée dans les systèmes d'entreposage de données, « *Le processus ETL est considéré aujourd'hui comme étant le coeur du système décisionnel puisque toutes les données destinées pour l'analyse transitent par celui-ci* » [18]. C'est une approche qui commence d'abord par l'extraction des données de différentes sources, puis par leur transformation dans le but de répondre aux besoins spécifiques d'une entreprise. Les données sont ensuite chargées dans, généralement, un entrepôt de données. Cependant, le plus souvent, cette approche nécessite une étape de transformation complexe pour nettoyer, filtrer et structurer les données avant leur chargement [19]. La figure 1.2 ci-dessous schématise le processus ETL.

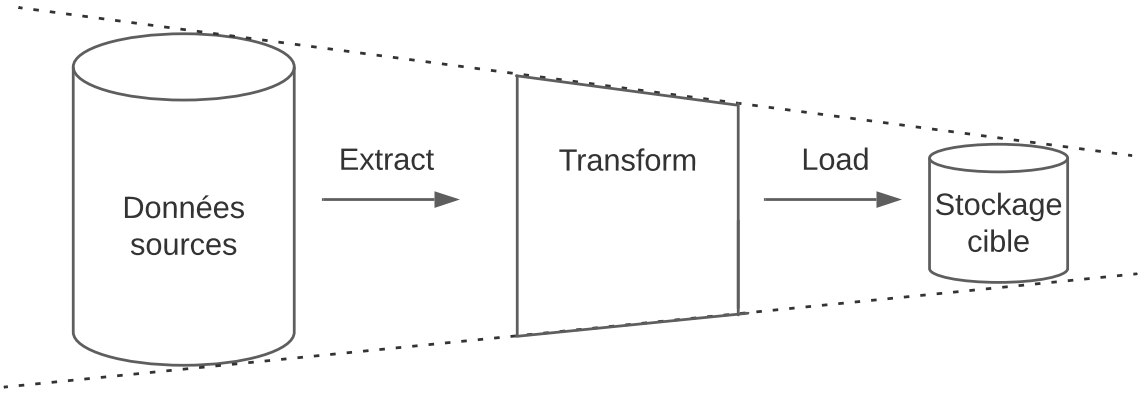


Figure 1.2: Le processus ETL (Extract Transform Load)

Le schéma sous forme d'entonnoir représente l'approche ETL orientée sur un objectif précis respectant un schéma de données défini.

1.5.2 ELT (Extraction, Chargement, Transformation)

L'ELT est une approche plus récente et évolutive. Dans le processus ELT, les données sont extraites des sources pour être directement chargées telles quelles "as is". Cette approche, permet un accès plus rapide et plus flexible aux données brutes car les transformations et les analyses sont effectuées ultérieurement. L'ELT offre ainsi une alternative moderne et efficace au processus ETL traditionnel [19]. La figure 1.3 suivante représente de manière schématique le processus ELT.

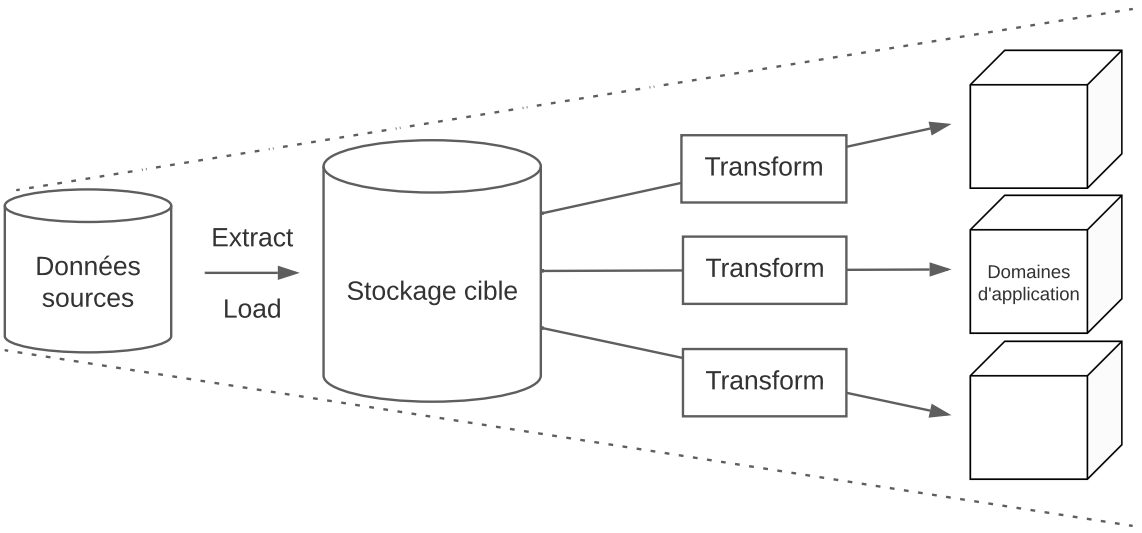


Figure 1.3: Le processus ELT (Extract Load Transform)

Contrairement à ETL, ce schéma met l'accent sur la souplesse de l'approche ELT. Elle ouvre la voie à une multitude de domaines d'application.

1.6 Étude comparative des entrepôts de données traditionnels et des lacs de données

Les Data Warehouses et les Data Lakes ont plusieurs similarités en termes de leur fonction. Tous deux sont des espaces de stockage capable de détenir une énorme quantité de données. Les deux peuvent être intégrés à des outils d'analyse permettant la compréhension et la visualisation des données. Cependant, les DW et les DL diffèrent sur plusieurs aspects.

Parmi ces différences, il convient de souligner les points suivants [20,21] (Tableau. 1.1) :

Critères	**Data Warehouse**	**Data Lake**
Données sources	Bases de données opérationnelles	Capteurs, réseaux sociaux, textes, images, vidéos, etc.
Caractéristiques des données	Structurées, homogénéisées	Format brut
Changements dans les données	Les données sont statiques. Les mises à jour sont peu fréquentes et coûteuses	Les données sont dynamiques. L'ingestion continue des données et mises à jour fréquentes
Approche d'intégration de données	ETL (Extract Transform Load)	ELT (Extract Load Transform)
Type de schéma	Schéma à l'écriture (schema on write)	Schéma à la lecture (schema on read)
Évolutivité	Rigide	Flexible
Utilisation	Axé sur la Business Intelligence	Principalement le Machine Learning et Data Science
Objectif	L'objectif est connu à l'avance	Objectif non déterminé
Complexité	Jointures complexes	Traitements complexes

Tableau 1.1: Comparaison entre le Data Warehouse et le Data Lake

Pour conclure, les caractéristiques qui différencient les deux espaces de stockage permettent aux organisations de choisir la solution la mieux adaptée à leurs besoins en matière de gestion, d'exploitation de données et de prise de décisions.

1.7 Aide à la décision dans un environnement Data Lake

Selon Madera [7], les lacs de données et les systèmes décisionnels sont deux entités distinctes.

Contrairement au systèmes décisionnels, dirigés par l'information, les lacs de données sont plutôt dirigés par les données ("Data Driven"). Cela signifie que les décisions sont prises à partir des données brutes collectées. D'autre part, les techniques "Information Driven" prennent des décisions grâce aux informations émergeantes des données sources. Pour arriver à cela, ces données sont collectées, nettoyées, agrégées à des fins analytiques.

Généralement, les Data Lakes se focalisent sur la gouvernance des données. Ils regroupent ainsi tous les aspects du cycle de vie de la donnée, à savoir, sa collecte, son stockage, sa classification, sa transformation et son utilisation.

En somme, les principaux objectifs des lacs de données sont axés vers la Data Science et l'apprentissage automatique, en raison de la nature prédominante des données non structurées. En effet, environ 80% des données au sein d'une entreprise sont non structurées [22]. C'est pour cela, qu'il s'avère intéressant d'exploiter celles-ci à des fins décisionnelles.

1.8 Conclusion

Après avoir mené une étude comparative entre les entrepôts de données et les lacs de données, nous remarquons que ces deux entités présentent des différences intéressantes voir complètement opposées. Cependant, le choix entre ces deux référentiels de stockage dépend de la nature des besoins spécifiques de l'entreprise. Si nous nous trouvons dans une situation où une certaine liberté et flexibilité s'impose, les Data Lakes sont adaptés à ce type de problématique grâce à leur nature versatile.

Chapitre 2

Travaux connexes

2.1 Introduction

Dans ce chapitre, nous présentons cinq travaux connexes sur les différentes architectures Big Data basées sur un Data Lake. Ces approches ont été réalisées dans différents domaines tels que la santé, les réseaux intelligents, l'agriculture et la pisciculture.

Chaque travail propose une architecture spécifique pour la gestion, le traitement et l'analyse des données. Nous synthétisons dans un tableau les différents outils utilisés par chaque approche. De plus, nous élaborons un autre tableau qui présente les critères de comparaison utilisés pour évaluer celles-ci.

2.2 Approche de Rangarajan et al. (2018) [1]

Pour implémenter leur lac de données, Rangarajan et al. [1] ont opté pour le *framework* Apache Hadoop pouvant stocker des types de données structurées, semi-structurées et non structurées grâce au module Hadoop Dsitributed File System (HDFS). En outre, Hadoop est tolérant aux pannes, évolutif et facile à développer.

De nombreux outils disponibles permettent d'injecter des données dans HDFS à partir de diverses sources. Dans le secteur de la santé, Rangarajan et al. ont utilisé trois types de sources de données, tels que des données massives provenant des centres de génomique et bio-banques ; et des flux de données provenant de laboratoires cliniques, centres de radiologie, réseaux sociaux, appareils portables, etc.

Architecture du lac de données

La figure 2.1 ci-dessous schématise les différentes couches de leur architecture Data Lake.

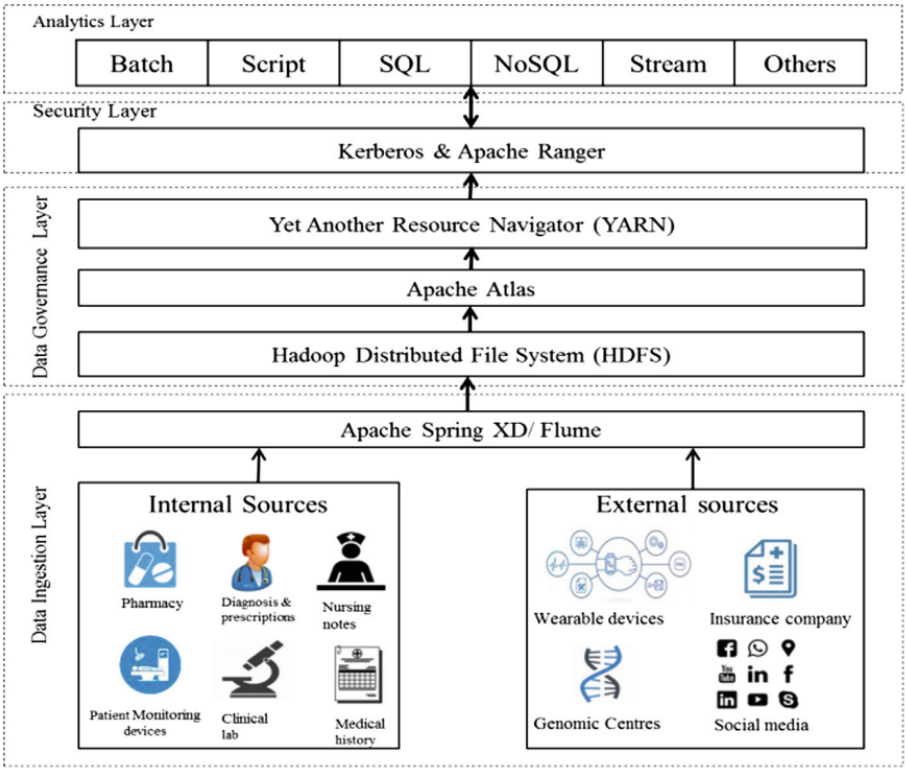


Figure 2.1: Architecture de Rangarajan et al. [1]

1. **Couche d'Ingestion :** Pour l'ingestion, ils ont utilisé l'outil Spring XD de Hadoop pour transférer des données en masse, gérer le flux de données et éviter les silos de données, ainsi que Apache Flume pour collecter, agréger et déplacer efficacement les données massives.
2. **Couche de Sécurité :** Pour contenir des informations très sensibles, les données de santé nécessitent un environnement hautement sécurisé. Dans ce sens, Rangarajan et al. ont prévu de doter HDFS d'un système de sécurité plus efficace : le protocole Kerberos et l'outil Apache Ranger.
3. **Couche de Gouvernance de Données :** Ils ont utilisé cette couche dans le but d'organiser, de gérer et d'accéder à toutes les données collectées. Pour gérer la gouvernance des métadonnées ils ont opté pour Apache Atlas. Cet outil possède des fonctionnalités importantes tels que la classification des données et l'audit centralisé.

4. **Couche Analytique :** Ils ont utilisé l'algorithme de partitionnement K-means avec l'environnement de programmation MATLAB et le SVM (Support Vector Machine), un modèle d'apprentissage automatique supervisé, dans le but d'agir comme un système pour les futures recommandations de médicaments.

Selon Rangarajan et al., l'entrepôt de données n'est plus adapté à l'analyse des soins de santé en raison de son schéma à l'écriture. Ils considèrent que le Data Lake (DL) est très efficace à l'utilisation des données massives. De plus, selon leurs expériences, le DL en surpasse nettement le DW en termes de temps d'ingestion des données, avec un temps de chargement et de stockage près de 50% inférieur à celui du DW.

2.3 Approche de A. Munshi et Y. Mohamed (2018) [2]

A. Munshi et Y. Mohamed [2] ont présenté un écosystème « Smart Grid Big Data » basé sur l'architecture Lambda. Ils considèrent que cette dernière est capable d'effectuer des opérations parallèles en batch (par lots) et en temps réel. Ils estiment qu'il gère des quantités massives de données de réseau intelligent en collectant puis stockant celles-ci dans un Data Lake sur le Cloud.

Architecture de l'approche

L'architecture des données du réseau intelligent se décompose en cinq étapes successives (voir figure 2.2) :

1. **Génération de données :** Les données du réseau intelligent sont considérées comme étant volumineuses, rapides et très variées.
2. **Collecte des données :** Ils ont utilisé l'outil Apache Flume pour l'ingestion des données transmises à une plate-forme de stockage Cloud.
3. **Stockage et traitement des données :** Dans la couche batch layer, les données ingérées sont stockées dans Hadoop HDFS et pré calculées à l'aide de Hadoop MapReduce (un modèle de traitement distribué).
   Dans la couche appelée speed layer, ils ont utilisé Apache Spark pour le traitement de données en temps réels.
4. **Interrogation des données :**Ils ont utilisé trois outils d'interrogation de données : Apache Hive qui utilise des opérations MapReduce, Apache Impala et Spark SQL. La fusion de ces trois composants d'interrogation des données rend l'écosystème du Big Data du réseau intelligent conforme aux couches de l'architecture Lambda.

5. **Analyse des données :** Trois outils analytiques ont été utilisés : Radoop pour le Data Mining, MATLAB pour l'organisation des tableaux et Tableau pour l'analyse visuelle. Ces outils comprennent l'exploration de données et la découverte de connaissances, ainsi que l'exploitation de statistiques.

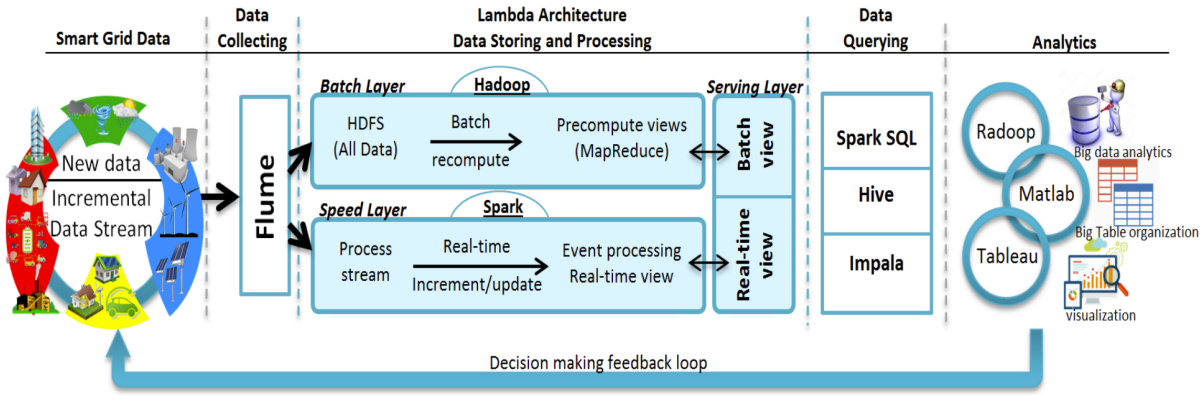


Figure 2.2: Architecture Lambda de A. Munshi et Y. Mohamed [2]

Pour faciliter l'exploration des données, ce type d'écosystème possède les compétences de réunir divers types de données du réseau intelligent : les images, les vidéos et les données de compteurs intelligents.

2.4 Approche de Sarramia et al. (2022) [3]

Sarramia et al. [3] ont développé une plateforme environnementale appelée "Cloud environnemental au profit de l'agriculture" (CEBA). Cette plateforme permet le partage, la recherche, le stockage et la visualisation de données scientifiques hétérogènes liées à l'environnement et à la recherche agricole. La conception de leur lac de données repose sur les quatre principales fonctionnalités (ingestion, stockage, traitement et accès), avec une gestion des métadonnées comme le détail la figure 2.3 ci-dessous.

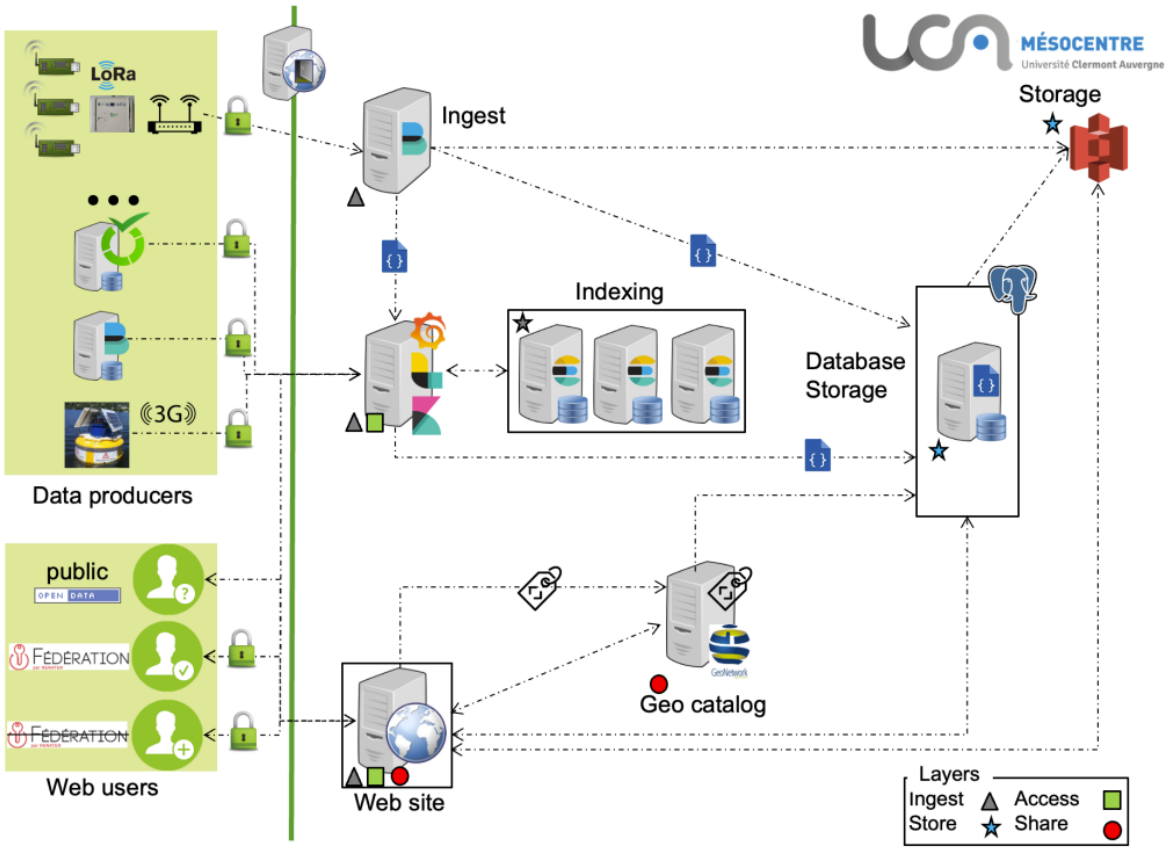


Figure 2.3: Architecture de Sarramia et al. [3]

Parmi ses principales caractéristiques : sa facilité d'utilisation et l'accessibilité à tous les types de données. Ils ont utilisé :

- Une base de données stockant des données géographiquement étendues dans JSONB (JSON Binaries) en utilisant PostgreSQL ;
- Un catalogue pour gérer les ressources référencées spatialement avec Geonetwork (Catalogue géo-spatial) ;
- Pour l'ingestion des données : Beats (expéditeurs de données) et Logstash (un moteur qui collecte et unifie les données provenant de plusieurs sources) ;
- Elasticsearch, un moteur de recherche distribué ;
- Kibana, un outil d'interface utilisateur d'Elasticsearch pour la visualisation.

Sarramia et al. ont créé un référentiel centralisé basé sur le concept du lac de données : stocker tous types de données à leur format d'origine. Leur plateforme permet la gestion des coordonnées géographiques, la gestion des métadonnées et l'ingestion de flux de données IoT. Ils ont également combiné des systèmes de stockage relationnels et NoSQL (Not Only Structured Query Language).

2.5 Approche de Ouafiq et al. (2022) [4]

L'agriculture intelligente, Smart Farming en anglais, est passé d'un simple concept en un besoin persistant. Dans ce sens, Ouafiq et al. [4] proposent une conception dédiée aux solutions basées sur la gestion des données pour le Smart Farming. Une stratégie de migration de données a été proposée pour gérer les différentes sources de données agriculturales avec une forte présence de composants Internet of Things (IoT).
Ils ont comme objectif principal fournir aux agriculteurs des solutions qui leur permettent le maintien de leurs processus liés à la distribution spatiale, à la gestion de l'eau et à la maintenance des systèmes mécaniques. Ils ont étudié les contraintes techniques liées au traitement des données dans un environnement Big Data. Ils ont proposé une solution qui repose sur trois principales étapes :

1. Définir les sources de données : elle comprend les Systèmes de Gestion de Base de Données (SGBD), les composants IoT, etc. ;
2. La conception de l'architecture qui peut gérer ces sources diverses, le traitement des données par lots et en temps réel ;
3. Fournir des Dashboards (tableaux de bord) et des rapports aux agriculteurs afin d'avoir une visibilité sur leur processus agricole.

Environnement logiciel

- Les outils Spark, Python, Kylin, HDFS, Hive et Impala seront utilisés pour assurer le traitement, la transformation et le stockage des données de la ferme.
- Les outils Apache Oozie et/ou Apache Airflow seront utilisés pour la création de Data-Workflows (flux de données).
- Des pipelines de données seront construits avec les logiciels Spark, Kafka, Flume, NiFi pour consommer, transformer et stocker des données en continu.

Architecture du lac de données

Leur lac de données est composé de cinq zones (Figure. 2.4) :

1. **Zone Partagée** : Classée sous la forme d'un groupe de dossiers structurés ;
2. **Zone Brute** : Où les données brutes seront stockées directement à partir de différentes sources ;
3. **Zone Structurée** : Où les données seront nettoyées, auront une structure spécifiques, et stockées conformément à un modèle en flocons (méthode de modélisation qui permet de créer une structure hiérarchique en forme de flocon de neige) ;

4. **Zone de Confiance** : Les données seront stockées en fonction de la logique métier ;
5. **Zone d'Enrichissement** : Cette couche est conçue pour la transformation des données et leur enrichissement avec les calculs des KPI (Key Performance Indicators) du Dashboard.

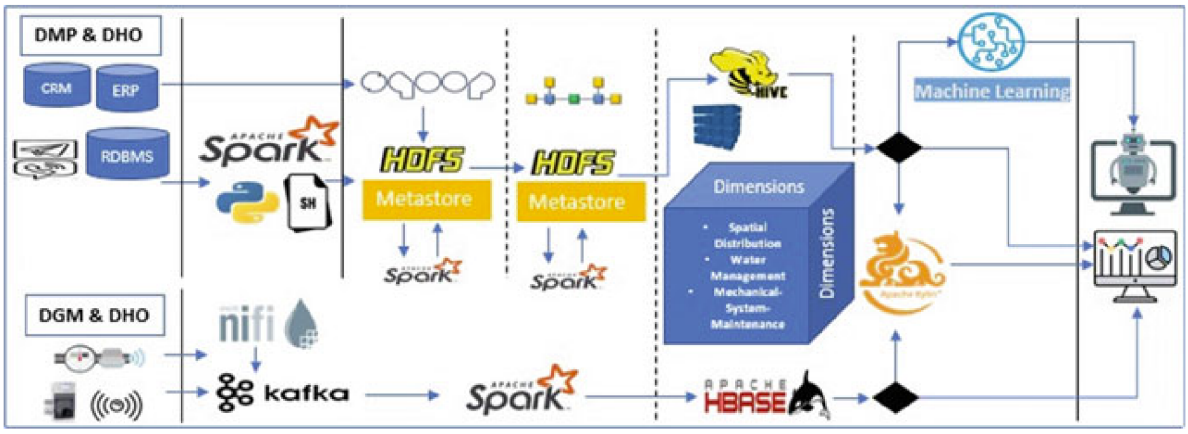


Figure 2.4: Architecture de Ouafiq et al. [4]

Afin de remédier aux limitations des systèmes traditionnels, Ouafiq et al. ont choisi l'écosystème Hadoop comme solution pour la prise de décision agricole. Ce système permet de gérer l'énorme quantité de données provenant de diverses sources.

L'architecture a été construite en couches pour permettre une meilleure visibilité sur les données et faciliter la gestion et la qualité des flux de données.

2.6 Approche de Benjelloun et al. (2023) [5]

L'objectif de Benjelloun et al. [5] est de proposer une architecture fonctionnelle Data Lake et de l'étendre à une architecture technique pour initier une stratégie de pisciculture basée sur les données.

Architecture fonctionnelle

Afin de gérer des mégadonnées piscicoles, leur architecture fonctionnelle est constituée de trois principales phases : la génération et la collecte de données, l'ingestion et le traitement des données et l'analyse et l'exploration des données (voir figure 2.5).

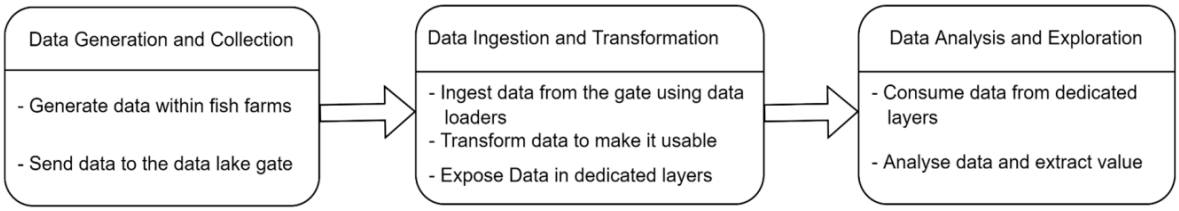


Figure 2.5: Architecture fonctionnelle de Benjelloun et al. [5]

Le choix de ce type d'architecture vise à transformer le domaine piscicole traditionnel en un domaine piloté par les données qui, à partir de ces données massives collectées auprès de diverses fermes piscicoles, permet d'analyser et d'extraire des informations précieuses. Ces dernières, grâce au Data Lake, seront accessibles à divers utilisateurs, même sans solide bagage technique.

Architecture technique

Leur architecture est organisée en plusieurs couches comme nous pouvons le voir dans la figure 2.6 :

1. **Zone Brute (Raw Zone)** : Dans HDFS, trois répertoires (données structurées, semi-structurées et non structurées) sont créés dans lesquels les données sont organisées par source. Ces trois répertoires HDFS constituent la zone brute.
2. **Zone Conforme (Trusted Zone)** : Pour passer à cette zone, différentes technologies ont été utilisées dépendamment du type de données. Les données structurées sont directement enregistrées dans des tableaux Hive structurés. Les données semi-structurées, telles que des fichiers JSON, XML et autres, sont transformées en utilisant Apache Spark. Chaque source de données semi-structurées est transformée et enregistrée dans la zone conforme, dans un tableau Hive structuré.
   Dans le cas de leur Data Lake sur la pisciculture, les données non structurées (images, données géo-spatiales, ...) sont généralement utilisées dans le domaine de la Data Science, c'est pour cette raison qu'ils ont consacré des répertoires dédiés à ce domaine.
3. **Zone d'Accès (Access Zone)** : Cette zone permet d'assurer que tous les outils externes ou équipes peuvent se connecter au Data Lake et lire les données conformément aux autorisations d'accès accordées. À ce niveau, une couche est créée pour chaque besoin et l'accès aux données est contrôlé par Apache Ranger. Ces couches peuvent être un groupe de tableaux Hive ou des répertoires spécifiques contenant des données non structurées.

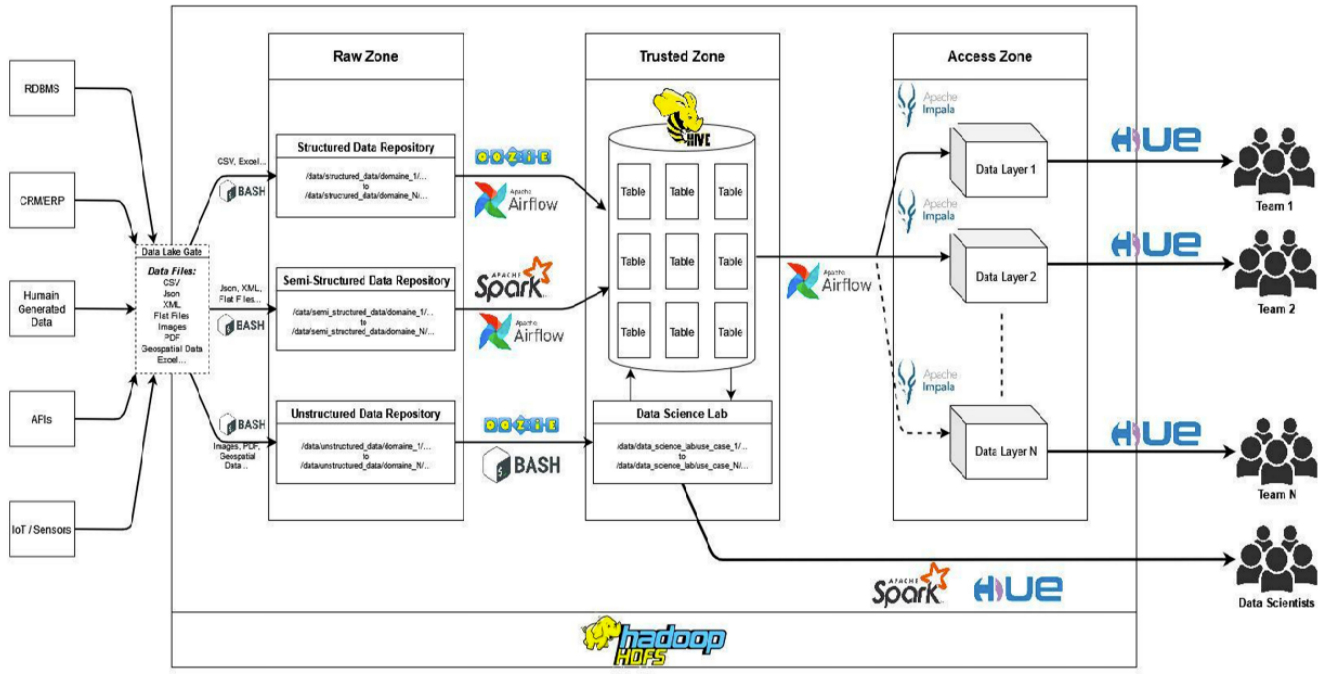


Figure 2.6: Architecture technique de Benjelloun et al. [5]

Benjelloun et al. estiment que les stratégies axées sur les données représentent le principal facteur de succès dans de nombreux domaines. Elles permettent une analyse avancée des données, dans le domaine agricole, et en particulier dans la pisciculture. Ces stratégies doivent être largement adoptées. Ils ont expliqué comment les technologies du Big Data offrent de multiples avantages pour la prise de décision.

Avec leur approche, ils visent à éclaircir la situation dans le domaine de la pisciculture en mettant en lumière les données massives générées dans les fermes piscicoles.

2.7 Synthèse et discussion

Le premier tableau 2.1 fournit une synthèse des technologies utilisées dans chaque approche, regroupées par type, ce qui permet de visualiser les outils adoptés par chaque étude.

Depuis ce tableau, nous constatons que la plupart des approches exploitent les technologies de l'écosystème Hadoop, ce qui témoigne de leur pertinence dans le traitement des données massives.

Aspects	Technologies	[1]	[2]	[3]	[4]	[5]
Stockage de données	Hadoop HDFS	✓	✓		✓	✓
	Apache Hbase				✓	✓
	Amazon S3			✓		
	PostgreSQL			✓		
Ingestion de données	Apache Flume	✓			✓	✓
	Apache Sqoop				✓	✓
	Apache Kafka				✓	
	Apache Nifi				✓	
	Hadoop Spring XD	✓				
	Beats			✓		
	Logstash			✓		
Scripting	Apache Hive		✓		✓	✓
	Apache Impala		✓		✓	✓
Traitement / Analyse de données	Apache Spark		✓		✓	✓
	Apache Kylin				✓	
	Elasticsearch			✓		
	Radoop		✓			
	MATLAB	✓	✓			
	Python				✓	
	Hadoop MapReduce		✓		✓	✓
Gouvernance des données / Workflow	Hadoop YARN	✓	✓			
	Apache Atlas	✓				
	Apache Oozie				✓	✓
	Apache Airflow				✓	✓
Sécurité	Apache Ranger	✓				✓
	Kerberos	✓				
Visualisation	Apache Hue					✓
	Kibana			✓		
	Grafana			✓		
	Tableau		✓			

Tableau 2.1: Tableau synthétisant les technologies utilisés par chaque approche

Le deuxième tableau 2.2 identifie les critères de comparaison utilisés pour évaluer chaque approche.

Critères	[1]	[2]	[3]	[4]	[5]
Domaine d'application	Santé et Médecine	Réseaux Intelligents	Agriculture	Agriculture Intelligente	Pisciculture
Mode d'ingestion	Par lots et en temps réel	Par lots et en temps réel	En temps réel	Par lots et en temps réel	Par lots et en temps réel
Langages utilisés	MATLAB	MATLAB, HiveQL, Spark SQL	R	Python, HiveQL	HiveQL
Hébergement	Modèle	Cloud	Cloud	Modèle	Modèle
Taille du data set	N/A	N/A	3 Go – 5 Go	14.4 Go	Modèle
Utilisateurs Potentiels	Docteurs, Pharmaciens	Spécialistes, Data Analysts	Utilisateurs Web	Agriculteurs	Data Scientists, Biologistes et autres
Système d'exploitation	N/A	Linux 64-bit	N/A	N/A	N/A
Système de fichiers	Distribuée	Distribuée	N/A	Distribuée	Distribuée
Catalogue de données				✓	
Gouvernance des données	✓		✓	✓	✓
Couche Sandbox					✓
Visualisation des données		✓	✓		✓
Sécurité	✓				✓
Variété	✓	✓		✓	✓
Scalabilité		✓		✓	✓

Tableau 2.2: Comparaison des approches par critères

À partir du tableau 2.2, nous remarquons que le type d'ingestion de données, pour la majorité des travaux, est par lots et en temps réel. Cependant, la sécurité des systèmes de ces approches est négligée par la plupart. Certains travaux sont dans la phase de conception et ne sont pas encore implémenté et déployés.

Dans les points suivants, nous récapitulons les aspects clés de chaque approche :

- L'approche de Rangarajan et al. (dans le secteur de la santé) souligne les avantages des lacs de données par rapport aux entrepôts de données en raison de leur flexibilité.
- L'approche de A. Munshi et Y. Mohamed (dans le contexte des réseaux intelligents) combine deux systèmes d'intégration de données à la fois. Il s'agit de l'architecture Lambda, qui offre la possibilité de tirer des informations provenant de diverses sources.
- L'approche de Sarramia et al. (dans le domaine de l'agriculture) intègre la gestion des coordonnées géographiques, la gestion des métadonnées et l'ingestion de flux de données IoT.
- Ouafiq et al., dans le cadre du smart farming, pensent qu'un environnement Data Lake centré sur Hadoop s'avère avantageux pour la prise de décision agricole. Ils considèrent que cet écosystème peut surmonter les limitations traditionnelles des systèmes décisionnels tel que la difficulté à gérer la variété des données.
- L'approche de Benjelloun et al. (dans le domaine de la pisciculture) se concentre sur la stratégie "Data Driven" en vue d'améliorer la gestion et les performances de l'élevage piscicole.

Il est important de noter que l'approche de Sarramia et al s'est penchée vers le service de stockage Amazon S3 et la base de données PostgreSQL pour le stockage des données. Cela pourrait poser des défis d'intégration avec d'autres outils couramment utilisés dans les architectures Data Lake.

2.8 Conclusion

L'étude des travaux connexes nous a offert un aperçu sur les différentes architectures existantes dans le domaine des Data Lakes. Cette étude nous a permis d'en tirer des enseignements qui nous ont servi de base pour la conception et l'implémentation de notre propre système.

Chapitre 3

Conception d'une architecture d'analyse de l'activité aquacole basée sur un Data Lake

3.1 Introduction

Après avoir établi une étude de l'existant, nous présentons, dans cette section, une architecture fonctionnelle de notre système avec comme référentiel de stockage : un Data Lake.

Notre contribution consiste en :

- Proposition d'un nouveau schéma d'ingestion de données : ECLT (Extract Classify Load Transform) ;
- Conception d'un processus d'ingestion en temps réel ;
- Modélisation orientée objet des données brutes ;
- Migration du modèle orienté objet vers un modèle orienté document ;
- Proposition d'un processus de capture et de transformation de données ;
- Mise en place d'une solution de visualisation et de restitution de données.

3.2 Étude de cas

3.2.1 Présentation des organismes d'accueil

Le Centre de Développement des Technologies Avancées (CDTA), lié par une convention avec le Centre National de Recherche et de Développement de la Pêche et de l'Aquaculture (CNRDPA), est un Établissement Public à caractère Scientifique et Technologique

(EPST) sous tutelle du Ministère de l'Enseignement Supérieur et de la Recherche Scientifique (MESRS) et la Direction Générale de la Recherche Scientifique et du Développement Technologique (DGRDST).

Le CNRDPA est le Centre National de Recherche et de Développement de la Pêche et de l'Aquaculture. Placé sous la tutelle du Ministère de la pêche et des productions halieutiques, c'est aussi un Établissement Public à caractère Scientifique et Technologique (EPST) qui se concentre sur la recherche et le développement dans le domaine de la pêche et de l'aquaculture.

Ses missions principales incluent l'évaluation des ressources halieutiques, le développement de l'aquaculture, l'étude des écosystèmes aquatiques et la valorisation des ressources aquatiques.

3.2.2 Processus de gestion du CNRDPA

La figure 3.1 ci-dessous schématise le processus de gestion des projets aquacoles.

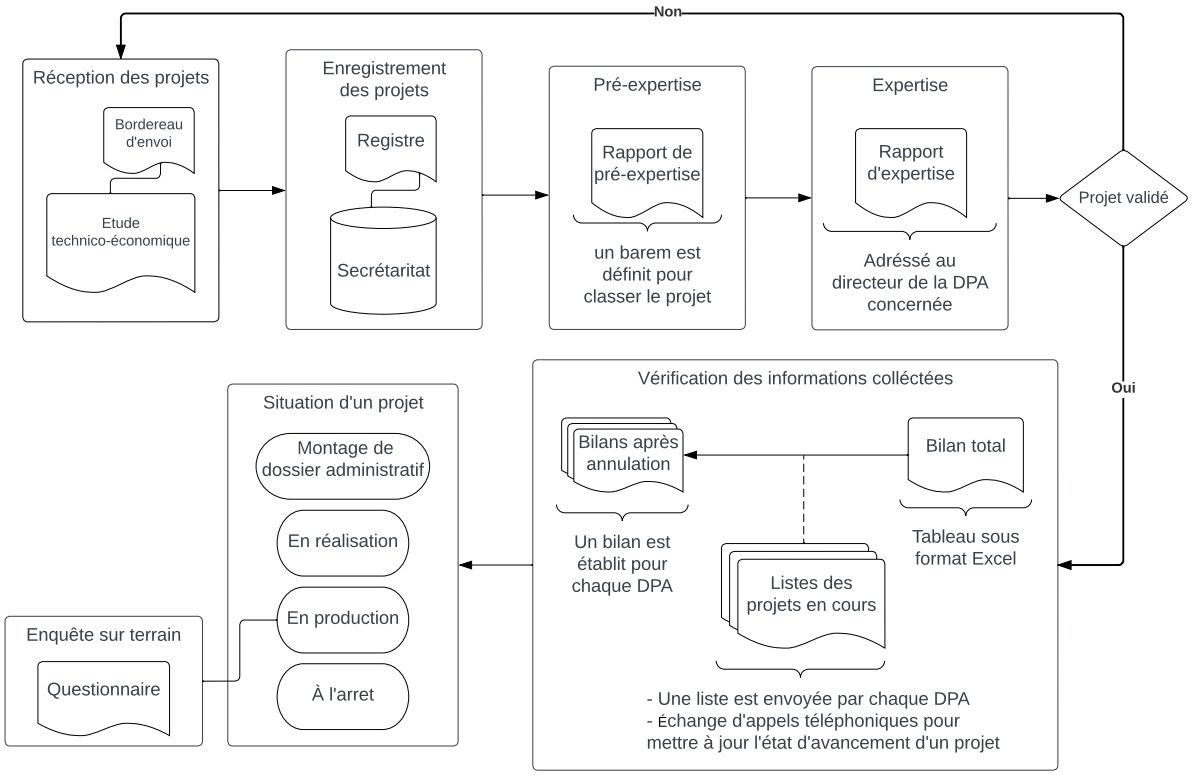


Figure 3.1: Processus de gestion du CNRDPA

Le CNRDPA est en charge de traiter tous les projets aquacoles au niveau national. Il reçoit les études technico-économiques des promoteurs de sa propre wilaya (Tipaza) ainsi que celles transmises par les 13 autres DPA (Directions de la Pêche et de l'Aquaculture)

des différentes wilayas. À leur arrivée, ces études sont enregistrées au niveau du secrétariat dans un registre manuscrit. Ensuite, elles suivent un cheminement défini de la manière suivante :

- Pré-expertise : Un ingénieur effectue une évaluation initiale où un rapport est rédigé, incluant la classification du projet.
- Expertise : Analyse approfondie de la faisabilité du projet. Un rapport est adressé aux DPA concernées contenant la validité du projet. Si celui-ci n'est pas validé, le promoteur est informé de réviser l'étude.
- Les données collectées des projets validés sont regroupées dans des fichiers Word. À partir de ceux-ci, un bilan total est réalisé, sous forme d'un tableau Excel.
- Le CNRDPA est informé des projets annulés, et des "Bilans après annulation" sont établis pour chaque DPA.
- Les projets validés peuvent être en cours de montage administratif, en réalisation, en production ou à l'arrêt.
- Des enquêtes sur terrain sont effectuées avec des questionnaires (documents papiers) pour les projets en production.

En résumé, les données sont sous forme de fichiers Excel, Word, PDF et documents papiers. L'ensemble de ces fichiers constitue nos « Données sources ».

3.2.3 Objectifs

Le CNRDPA collecte des données hétérogènes (Excel, Word, PDF, etc.) provenant des 14 DPA réparties sur le territoire Algérien. Dans le cadre du développement de l'aquaculture marine et continentale, l'objectif global du CNRDPA est de disposer d'un système d'information qui facilite l'exploitation, l'analyse et la visualisation de ces données.

Ce système vise à soutenir une meilleure prise de décision pour les investissements dans le domaine de l'aquaculture à l'échelle nationale.

3.3 Proposition d'une architecture fonctionnelle du système

Après analyse de l'existant de la filière aquaculture en Algérie, il ressort que les données sont majoritairement non structurés (Word, PDF). La quantité des données risque d'évoluer de façon très importante. De ce fait, nous préconisons une architecture Big Data avec un stockage sous forme de Data Lake.

Cette architecture est de type Lambda : une approche hybride consistant à intégrer et

traiter les données sources à la fois par lots et en temps réel.

La figure 3.2 représente l'architecture fonctionnelle.

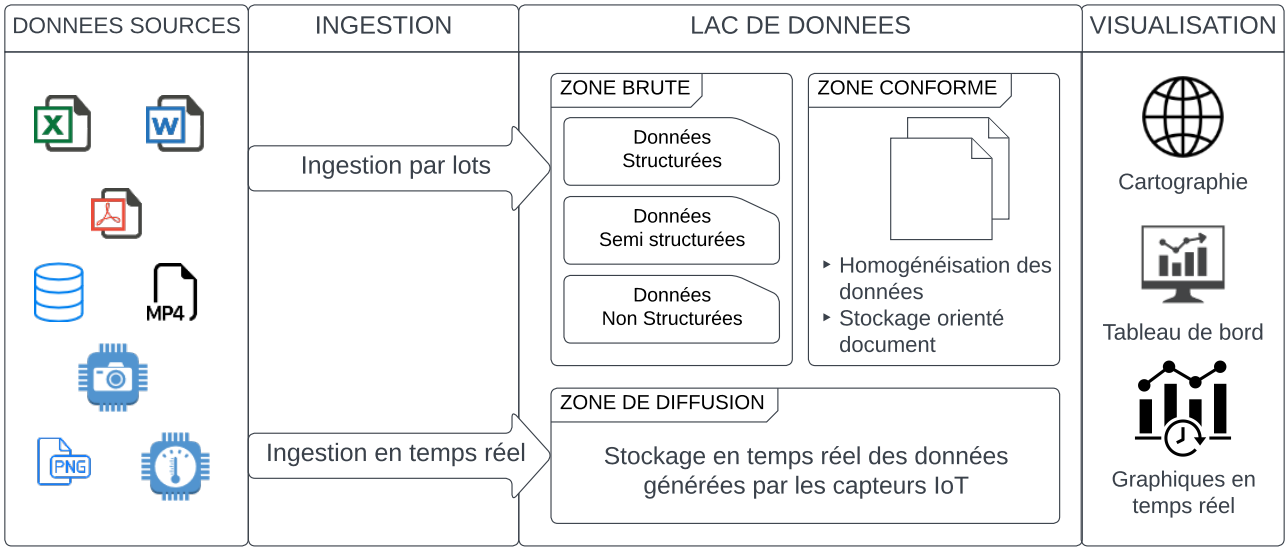


Figure 3.2: Architecture fonctionnelle du système

Le cheminement des données se synthétise par quatre principales phases :

a. Données sources

En plus des données générées à partir des études technico-économiques (Word, PDF, etc.), il existe un autre type de données dont la diffusion est progressive et constante au fil du temps. Il s'avère nécessaire de les collecter au moment de leur génération afin de les exploiter à des fins multiples.

b. Ingestion

Il existe différents types de processus d'ingestion, parmi eux : l'ingestion par lots, qui collecte les données périodiquement, et l'ingestion en temps réel qui reste à l'écoute des données en diffusion.

c. Lac de données

Ce lac de données est partitionné en trois zones distinctes :

1. **Zone Brute :** Cette zone contient les données ingérées à leur état brute ("as is") afin de respecter le concept clé d'un Data Lake. Mais, il est indispensable de les organiser pour alléger les traitements ultérieurs.

2. **Zone Conforme :** À partir de la zone brute, les données sont traitées, homogénéisées, et stockées sous forme de documents. Elle est appelée "conforme" car les données sont standardisées et adaptées à des fins d'analyse et de visualisation.
3. **Zone de Diffusion :** À ce niveau, les données interceptées sont stockées au fil du temps conformément à la fréquence de diffusion des données sources pour une visualisation quasi instantanée.

d. Visualisation

Le système doit être capable de restituer les données homogènes depuis la zone conforme, ainsi que les données progressivement générées au fil du temps depuis la zone de diffusion. Cette restitution peut être sous forme de graphiques tels que des diagrammes en barres, des diagrammes circulaires, etc., ou de cartographie pour les données géo-spatiales.

Dans les prochaines sections, nous présenterons une explication détaillée sur les composants et fonctionnalités de l'architecture.

3.4 Les données sources

Dans cette section, nous étudions la manière dont les données sources sont structurées et comment elles peuvent être optimisées pour répondre aux besoins de notre système.

3.4.1 Modélisation des données sources

Nous avons opté pour la modélisation orientée objet en raison de son formalisme puissant, idéal pour représenter efficacement les données sources. Grâce à l'utilisation d'un diagramme de classes, nous sommes en mesure de comprendre de manière claire la structure des données.

La figure 3.3 représente le diagramme de classes relatif aux données de l'activité aquacole en Algérie.

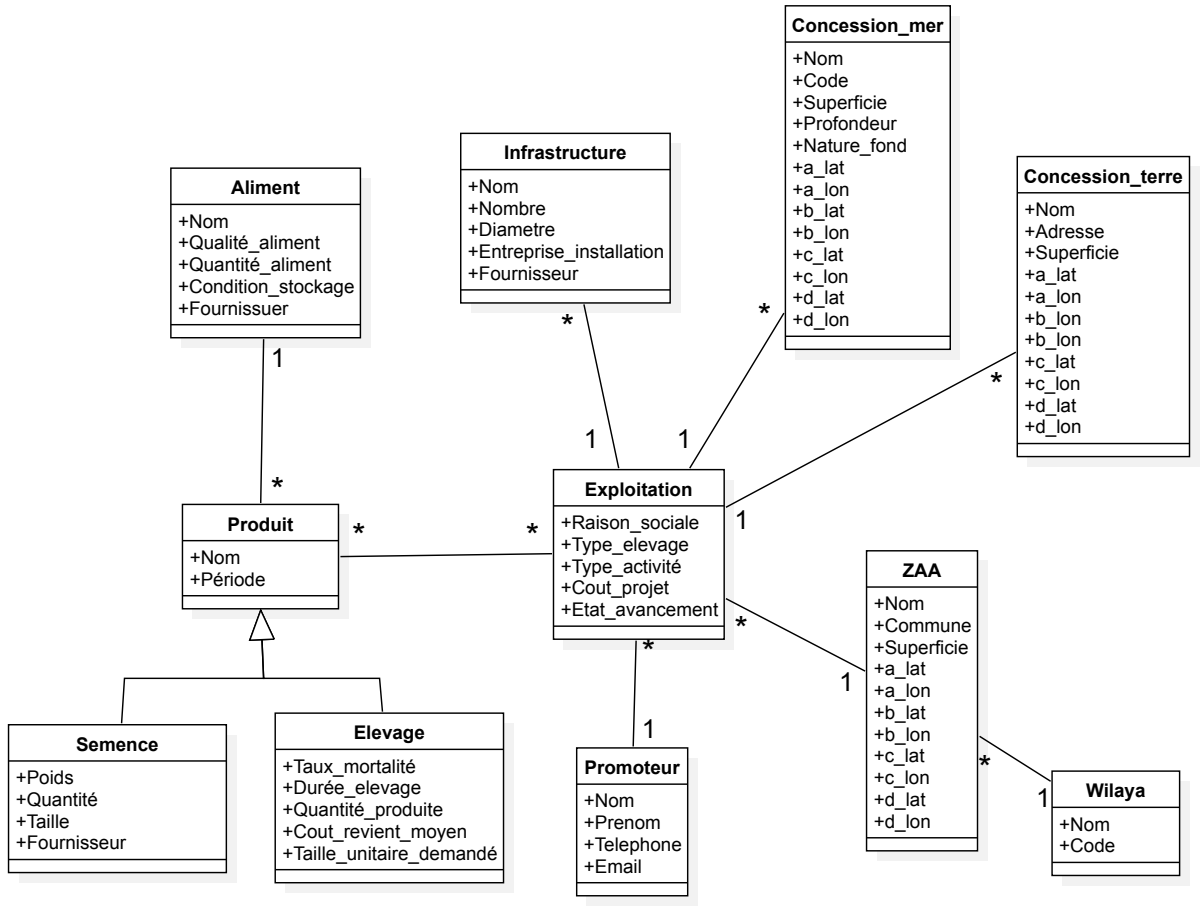


Figure 3.3: Diagramme de classes relatif à l'activité aquacole

Relations entre classes

La classe *Exploitation* représente une société engagée dans l'activité aquacole. Elle appartient à la classe Promoteur (le propriétaire de celle-ci). Une exploitation possède une ou plusieurs concessions en mer, et concession à terre. Les concessions en mer se trouvent dans des zones allouées à l'aquaculture (ZAA) et qui, à leur tour, appartiennent à une Wilaya. La classe *Produit* est une superclasse de *Semence* et *Elevage*. La classe *Semence* regroupe différents types de semences aquacoles tels que les alevins, les naissains et les post-larves. La classe *Elevage* représente la phase d'élevage, de récolte, et de commercialisation des poissons, coquillages, ou crevettes.

Attributs

Le tableau 3.1 explique les attributs des différentes classes.

Classe	Attribut(s)	Description
Exploitation	Raison_sociale	Nom de la Société (Personne physique ou morale)
	Type_elevage	Cages flottantes en mer, bassins à terre
	Type_activité	Type de la discipline (Pisciculture, Conchyliculture, Crevetticulture)
	Cout_projet	Coût total du projet en Dinar Algérien (DA)
	Etat_avancement	Situation d'avancement du projet
Promoteur	Nom, Prénom, Telephone, Email	Coordonnées du promoteur
Concession_mer	Code	Code du projet (année de validation plus le numéro de la wilaya plus le numéro du projet)
	Profondeur	Profondeur moyenne du site en mer
	Nature_fond	Typologie du fond : sableux, vaseux, etc.
Infrastructure	Nom	Cage, Filière, Bassin, etc.
	Nombre	Nombre existant dans la ferme
	Diamètre	En mètre (M)
	Entrep_instal	Coordonnées de l'entreprise d'installation
Semence	Poids	En gramme (G)
	Taille	En centimètre (CM)
Elevage	Taux_mortalité	En pourcentage (%)
	Durée_elevage	Durée du cycle d'élevage en mois
	Quantité_produite	En tonnes (T)
	Cout_rev	Coût en (DA) par (Kg) de matière biologique (poisson, coquillage ou crevette)
	Taille_unit	En centimètre (CM)

Tableau 3.1: Description des attributs des différentes classes.

Nous remarquons que certains attributs sont partagés par plusieurs classes :

- ***a_lat, a_lon, ..., d_lon :*** Il s'agit des coordonnées géographiques (latitude, longitude) en degrés décimaux, délimitant une zone allouée à l'aquaculture (ZAA), une concession en mer, et une concession à terre.
- ***Période :*** cet attribut représente la période de semencement pour la classe *Semence*, et la période de récolte pour la classe *Elevage*.
- ***Superficie :*** mesurée en hectares ou en m^2, et se retrouve dans les classes *ZAA*, *Concession_mer*, *Concession_terre*.
- ***Fournisseur :*** Coordonnées du fournisseur pour chaque classe (Semence, Aliment, Infrastructure).

3.4.2 Migration vers un modèle dénormalisé NoSQL orienté document

Vu que l'architecture cible est stockée sur un référentiel Data Lake, il est clair que nous ne pouvons plus continuer à adopter un formalisme destiné pour les données structurées (modèle orienté objet).

Dans ce qui suit, nous présentons la démarche adoptée pour la migration d'un schéma orienté objet vers un schéma NoSQL orienté document.

Démarche

Dans un contexte orienté document, chaque classe du schéma 3.3 représente un document ou un objet.

- La classe *Exploitation* représente la classe mère, qui agit comme le document principal. Les attributs tels que *Raison_sociale*, *Type_elevage*, *Type_activité*, etc., sont représentés en tant que champs dans ce document.
- L'objet *Wilaya* est imbriqué dans l'objet *ZAA*.
- L'objet *Promoteur* est incorporé dans le document *Exploitation* en tant que sous-objet. Cela permet de regrouper les coordonnées du *promoteur* avec les informations générales de l'*exploitation*.
- Un objet *Coordonnées* est créé avec comme clés : les attributs *a_lat*, *a_lon* jusqu'à *d_lon*, et les coordonnées géographiques en degrés décimaux comme valeurs. Puis, celui-ci est intruduit dans chacun des objets *ZAA*, *concession_mer*, *concession_terre*.
- En somme, les objets *ZAA*, *Concession_mer*, *Concession_terre*, et *Infrastructure* sont imbriqués dans le document principal *Exploitation*.
- Un objet *Produit* est créé contenant une liste de deux objets (*Semence* et *Elevage*),

l'objet *Aliment* est imbriqué dans chacun d'eux car l'alimentation diffèrent entre la phase de semencement et la phase d'élevage.
Cependant, il est possible de considérer les objets *Semence* et *Elevage* comme deux documents distincts imbriquant l'objet *Aliment* dans ces derniers. Ainsi, deux variantes du modèle orienté document se présentent (schéma (a) 3.4 et schéma (b) 3.5).

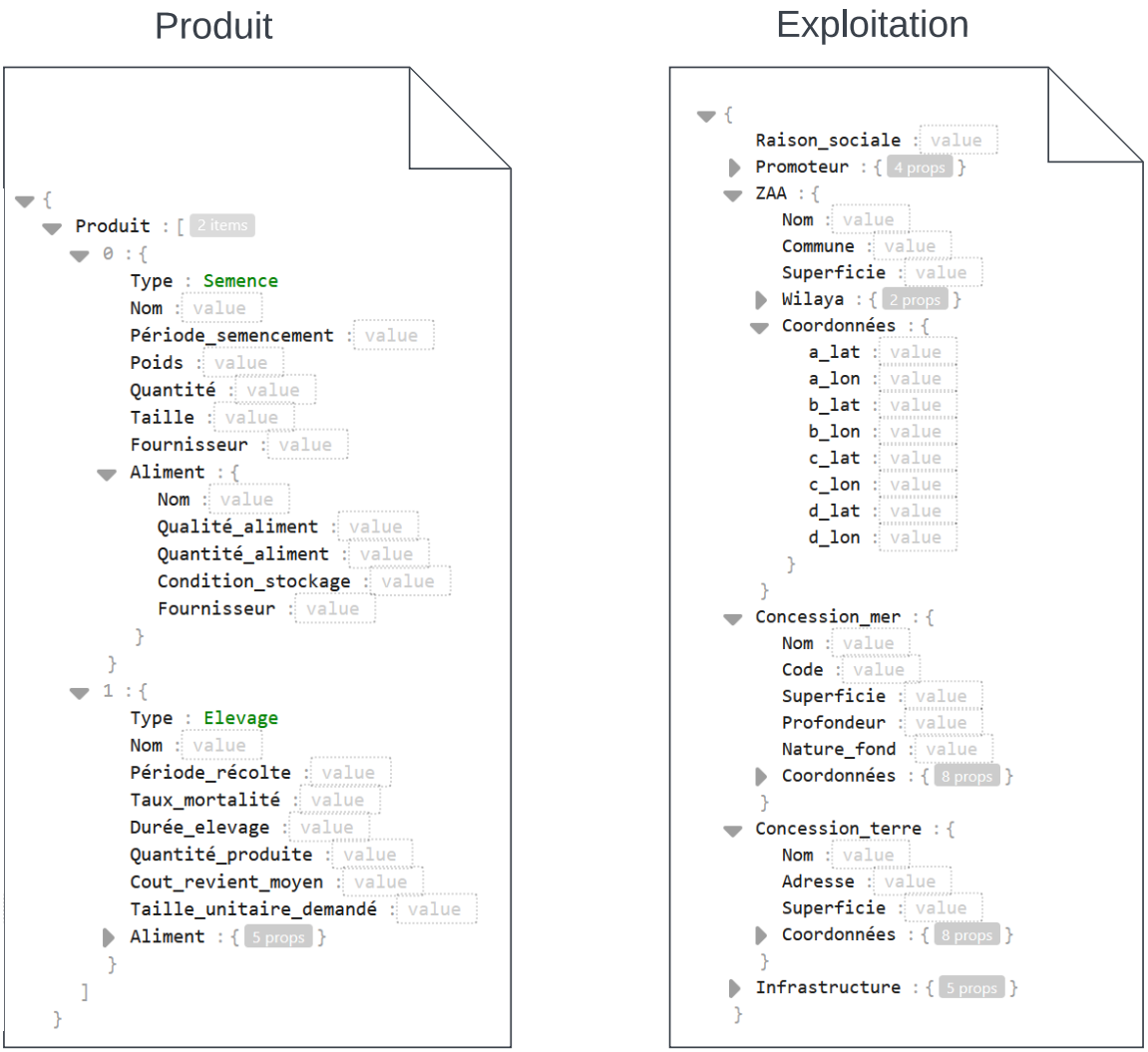


Figure 3.4: Modèle de données dénormalisé orienté document (a)

Le schéma (a) permet de regrouper toutes les informations liées à la production aquacole dans un seul document. Quant au schéma (b), il permet de gérer séparément les informations spécifiques à chaque phase (semencement ou élevage). Dans la suite de notre travail, nous avons opté pour le schéma (b) 3.5.

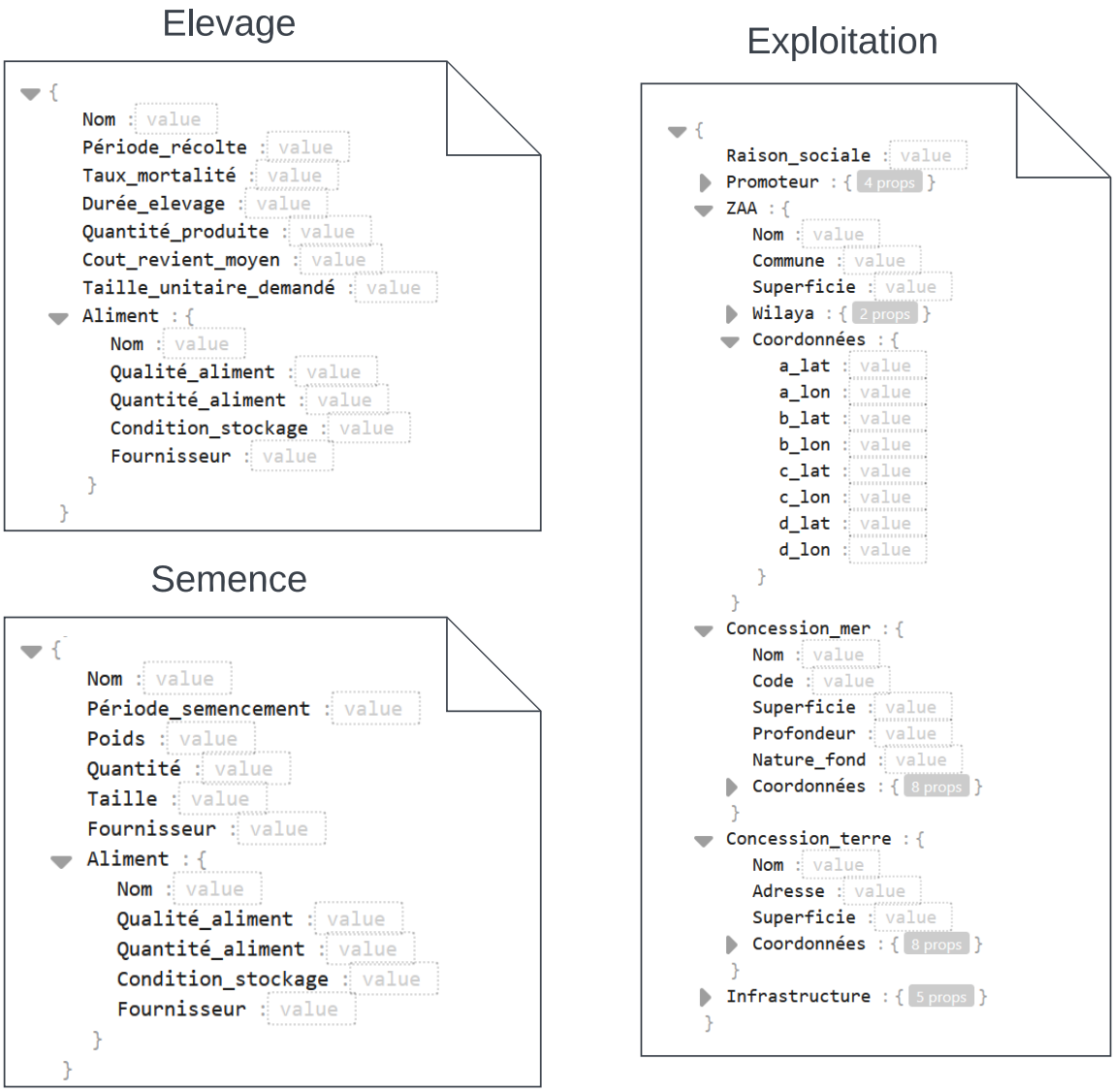


Figure 3.5: Modèle de données dénormalisé orienté document (b)

Avantages et inconvénients

Nous résumons dans ce qui suit, quelques avantages du modèle orienté document :

— Les documents peuvent contenir des données hétérogènes et imbriquées en raison de la nature "sans schéma" (schema-less) de ce modèle ;

— Ce modèle permet l'accès à toutes les informations nécessaires en une seule requête ;

— Performant en termes de requêtes et accès aux données car il évite l'utilisation de jointures coûteuse en temps.

Cependant, la redondance est considérée comme l'un des points faibles de ce modèle.

Les informations peuvent être répétées dans différents documents. De plus, vu la flexibilité du modèle, la maintenance de celui-ci peut s'avérer difficile. [23].

3.5 L'ingestion de données

3.5.1 Nouveau schéma d'ingestion de données : ECLT

Dans le domaine de l'intégration et de la transformation des données, deux processus sont couramment utilisés :

— l'ETL (Extract, Transform, Load) : généralement utilisé dans le contexte des entrepôts de données. Ce processus est illustré par le modèle (a) dans la figure 3.6.

— l'ELT (Extract, Load, Transform) : typiquement utilisé dans un environnement Data Lake (voir modèle (b)).

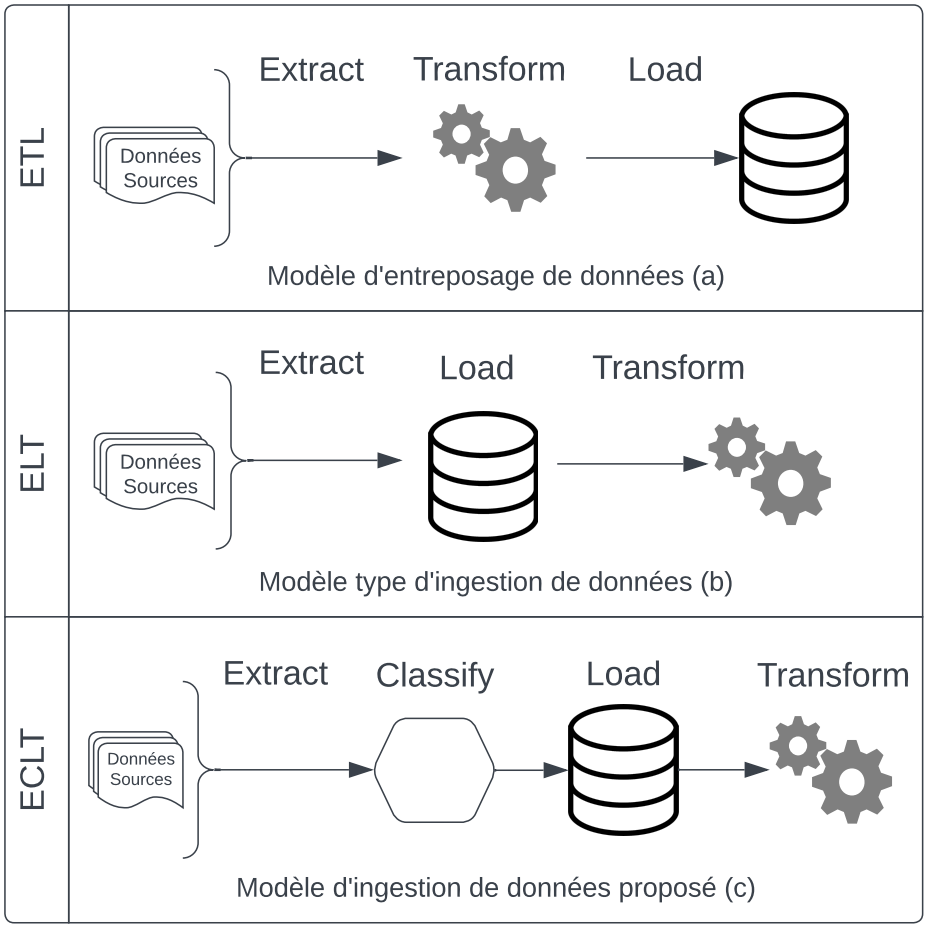


Figure 3.6: Modèle d'entreposage de donnée (a), modèle type d'ingestion de données (b), modèle d'ingestion de données proposé (c)

Dans le cas de notre présent travail, nous proposons un nouveau modèle d'ingestion de données, celui de l'ECLT (Extract Classify Load Transform). Nous avons introduit une nouvelle phase, « Classify » qui impose une classification aux données brutes dans le lieu de stockage cible comme le montre la figure 3.6 (c). ECLT organise les données par type et par format. En effet, les données brutes sont tout d'abord classées selon leur degré de structuration ("structurées", "semi- structurées" et "non structurées"), et par la suite, selon leur format natif (Word, PDF, Excel, etc.).

L'apport de cette phase de classification a pour but, non seulement, de simplifier les différents processus de traitement des données brutes, mais aussi, d'éviter à ce que la zone brute ne se transforme en un stockage sans stratégie souvent appelé "Data Swamp" (marécage de données).

3.5.2 L'ingestion en temps réel

Dans le domaine de l'aquaculture, les fermes aquacoles utilisent fréquemment des capteurs qui génèrent en continu des données précieuses, telles que la température, le taux de pH et le niveau d'oxygène. Ces mesures sont primordiales pour la prise des décisions et la surveillance de l'environnement aquacole. Si ces données ne sont collectées au moment de leur diffusion, elles seront perdues. C'est la raison pour laquelle il est nécessaire d'avoir un système de capture permanent.

Les capteurs IoT utilisent le protocole MQTT (Message Queue Telemetry Transport) pour transmettre de l'information. Ce dernier est un protocole de communication qui fonctionne sur le principe de publication/abonnement : d'un côté, les clients éditeurs publient des messages sur des sujets spécifiques, et d'un autre côté, des clients abonnés reçoivent ces messages aux sujets auxquels ils se sont abonnés. Un serveur MQTT agit en tant que courtier (Broker) ou intermédiaire entre ces clients [24].

Dans notre contexte, un outil d'ingestion en temps réel joue le rôle d'un abonné qui charge instantanément les données reçues dans la zone de diffusion.

3.6 Capture et transformation des données

Le schéma 3.7 suivant illustre le processus de capture et de transformation des données.

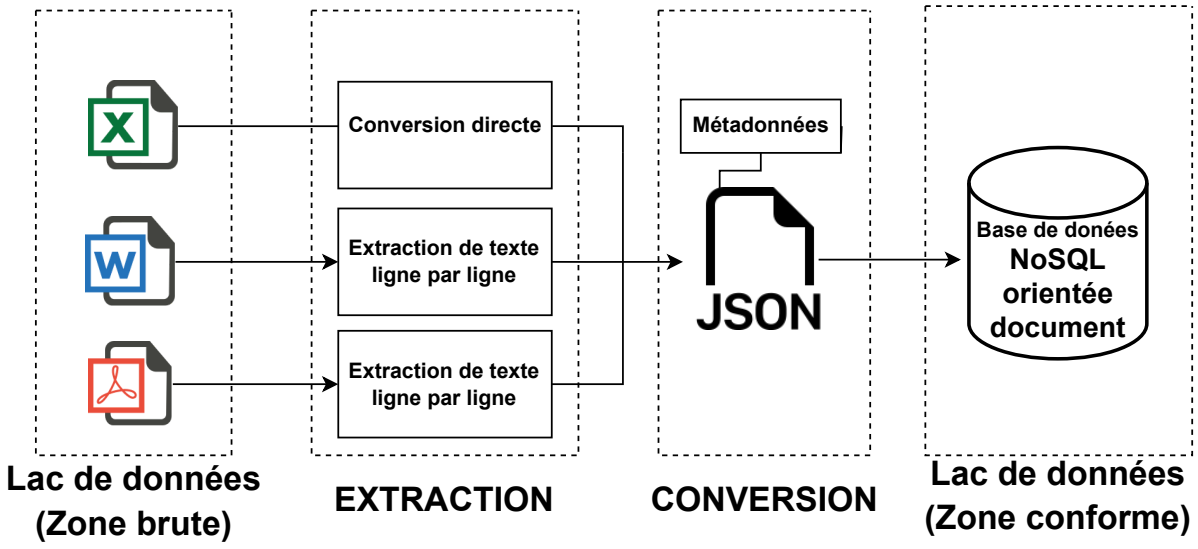


Figure 3.7: Processus de capture et de transformation des données

Dans la zone brute du lac de données, les fichiers dans leur état initial sont inexploitables. C'est la raison pour laquelle un processus de capture et de transformation des données brutes est introduit. Nous nous focalisons sur les fichiers Word, PDF, et Excel.

Ces documents suivent un modèle qui consiste en une liste contenant les informations d'une exploitation (la raison sociale, le type d'activité, le nom du promoteur, etc.), et l'extraction de ces données diffère selon le type de fichier.

Notre processus respecte la logique suivante :

— Chaque ligne extraite des fichiers Word et PDF est associée à un entête et transformée en un objet JSON.

— Puisque les fichiers Excel sont de type structuré, ils peuvent être convertis directement en JSON, un format flexible et adapté à notre approche orientée document.

— Ces objets sont ensuite stockés dans une collection d'une base de données NoSQL orientée document. Celle-ci représente la zone conforme de notre lac de données.

— Ainsi, chaque ligne de chaque fichier correspond a un document dans la zone conforme.

Cette approche permet d'homogénéiser les données brutes et de les exploiter à des fins décisionnelles.

3.7 Visualisation et exploration des données

Après la collecte et la transformation des données, il est important de les visualiser de manière claire et compréhensible. Suite à l'analyse des données de la zone conforme, nous avons remarqué la présence de valeurs géo-spatiales. Ces valeurs représentent les points

géographiques délimitant les fermes aquacoles. Il est nécessaire de cartographier celles-ci pour avoir une vue d'ensemble sur toutes les concessions existantes sur le territoire Algérien.

En plus de la présentation cartographique, la visualisation en temps réel est également un aspect important dans notre approche. Comme nous l'avons déjà mentionné, le processus d'ingestion récupère les paramètres provenant des capteurs IoT (température, pH, oxygène, etc.) puis, les chargent dans la zone de diffusion. Le système prend ces données et les affichent en temps réel, offrant une vue instantanée des conditions aquacoles.

3.8 Conclusion

Notre approche vise à concevoir une architecture innovante et robuste capable d'exploiter des données brutes, massives et diffusées en en temps réel. L'objectif est de faciliter la visualisation de ces données afin d'améliorer la prise de décision dans le domaine de l'aquaculture.

Chapitre 4

Implémentation, tests et évaluation

4.1 Introduction

Ce dernier chapitre est consacré à l'implémentation de l'architecture du système. Nous abordons l'architecture technique et les couches qui le constituent, les technologies utilisées, les différents modes d'ingestions et les tests effectués. Enfin, nous présentons l'interface utilisateur web développée pour le système.

4.2 Architecture technique du système

Après avoir exposé l'architecture fonctionnelle du système, nous présentons dans ce qui suit l'architecture technique en mettant en évidence les différents outils technologiques utilisés (illustré dans la figure 4.1).

a. Ingestion

— **Python :** Pour l'ingestion par lots, nous avons opté pour le langage de programmation orienté objet Python. Il a une syntaxe simple et facile à apprendre. Python bénéficie d'un ensemble de bibliothèques et outils dédiés à l'intégration et au traitement de données [25]. La partie ECL du Processus proposé ECLT est écrite avec ce langage.

— **Apache Nifi :** Cet outil est utilisé pour le processus d'ingestion en temps réel. Il se connecte à différentes sources de données, y compris les capteurs IoT. Apache Nifi propose une interface utilisateur graphique basée sur le web, qui facilite la gestion et la modification des flux de données [26].

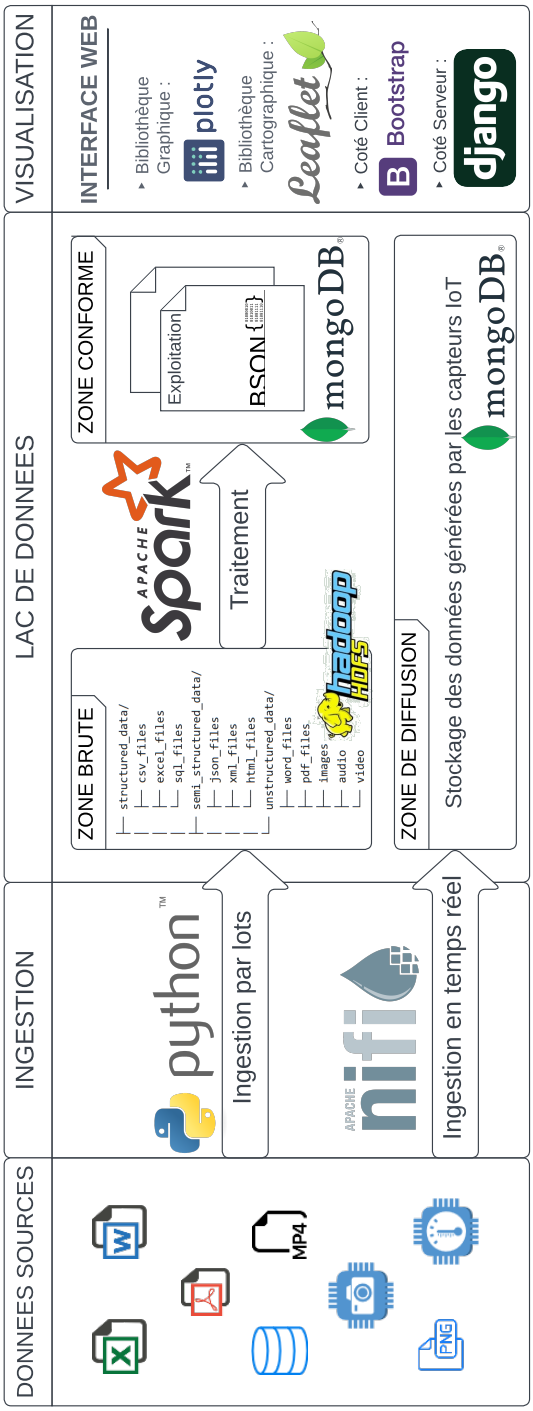


Figure 4.1: Architecture technique du système

b. Lac de données

La Zone Brute

Notre choix s'est fixé sur Hadoop HDFS, car il permet de gérer une grande quantité de données. Ses principales qualités sont sa haute disponibilité et sa tolérance aux pannes. HDFS permet d'ajuster l'évolutivité du système en ajoutant de nouveaux nœuds (machines physiques) au cluster (ensemble de nœuds interconnectés) [27]. Enfin, avoir un lac de données basé sur Hadoop nous permet de bénéficier de son écosystème qui offre une multitude de technologies telles que Apache Airflow, Apache Hive, etc.

Trois répertoires sont créés dans HDFS : (structured_data, semi_structured_data, unstructured_data), avec chacun des sous-répértoires. Par exemple, le répertoire semi_structrued_data regroupent les sous-répertoires json_files, xml_files, html_files. Un aperçu de cette arborescence est illustré dans la figure 4.2 suivante.

```
.
├── structured_data/
│   ├── csv_files
│   ├── excel_files
│   └── sql_files
├── semi_structured_data/
│   ├── json_files
│   ├── xml_files
│   └── html_files
└── unstructured_data/
    ├── word_files
    ├── pdf_files
    ├── images
    ├── audio
    └── video
```

Figure 4.2: Arborescence de la zone brute

La Zone Conforme

— **Apache Spark :** Ce framework de traitement des données est distribué et permet la parallélisation des tâches. Contrairement au framework MapReduce, il effectue les traitements en mémoire, le rendant beaucoup plus rapide. De plus, Spark prend en charge plusieurs langages, dont Python [28].

— **MongoDB :** Est une base de données NoSQL distribuée et orientée document. Elle stocke les documents sous forme de BSON (Binary JSON), une variante de JSON. Elle regroupe ces documents dans des collections. MongoDB offre des performances élevées pour les opérations de lecture et d'écriture le rendant ainsi adapté dans un environnement Big Data [29].

La Zone de Diffusion

Cette zone est représentée par une collection dans MongoDB. Elle intercepte les données ingérées par Apache Nifi.

c. Visualisation

Le produit final du système est une interface utilisateur web interactive, implémentée à l'aide des outils suivants :

— **Plotly :** Est la bibliothèque la plus puissante dans le domaine de la visualisation de données. Plotly possède le plus de fonctionnalités comparées à d'autres outils [30]. Elle offre la possibilité de créer des graphiques interactifs et dynamiques. Elle est adéquate pour les données diffusées en temps réel.

— **Leaflet :** Une bibliothèque JavaScript cartographique très populaire pour la visualisation des données géographiques.

— **Django :** Est un framework de développement web basé sur Python. Il est full stack, c'est-à-dire qu'il dispose de tous les outils nécessaires pour développer une application complète.

— **Bootstrap :** Un framework CSS, HTML et JavaScript. Il a des composants HTML pré construits, ce qui facilite et accélère le développement d'une interface utilisateur esthétique.

Ces technologies fonctionnent en harmonie pour la création d'une architecture Big Data. Il est important de mentionner que toutes les technologies utilisées sont open source.

4.3 Environnement de développement

Le développement du système a été réalisé dans une machine virtuelle VirtualBox (logiciel de virtualisation open source). Le tableau 4.1 suivant présente les spécifications du matériel utilisé.

Composant	Description
Type de PC	Ordinateur portable
Marque et modèle	Dell Inspiron 7400
Processeur	Intel Core i7-1165G7
Mémoire RAM	16 Go
Stockage	SSD 1 To
Carte graphique	NVIDIA GeForce MX350
Système d'exploitation	Windows 10 Home
Logiciel installé	VirtualBox
Périphérique	Écran externe Asus 24 pouces
Configuration de la machine virtuelle avec VirtualBox	
Système d'exploitation	Ubuntu 22.04 LTS
Mémoire RAM allouée	6 Go
Espace disque alloué	50 Go
Nombre de cœurs CPU	3

Tableau 4.1: Spécifications du matériel utilisé

4.4 Jeu de données

Pour évaluer notre approche, nous avons généré deux types de jeux de données distincts :

1. Un ensemble de fichiers Word représentant notre donnée brute ;
2. Un ensemble d'objets JSON générés chaque seconde.

Le premier jeu de données est traité par le processus d'ingestion ECLT. Le contenu de chaque fichier Word est sous forme de liste où chaque ligne représente une exploitation comme le montre la figure 4.3. La taille totale de ce jeu de données est 1.3 Go, ce qui correspond à 110,100 fichiers Word.

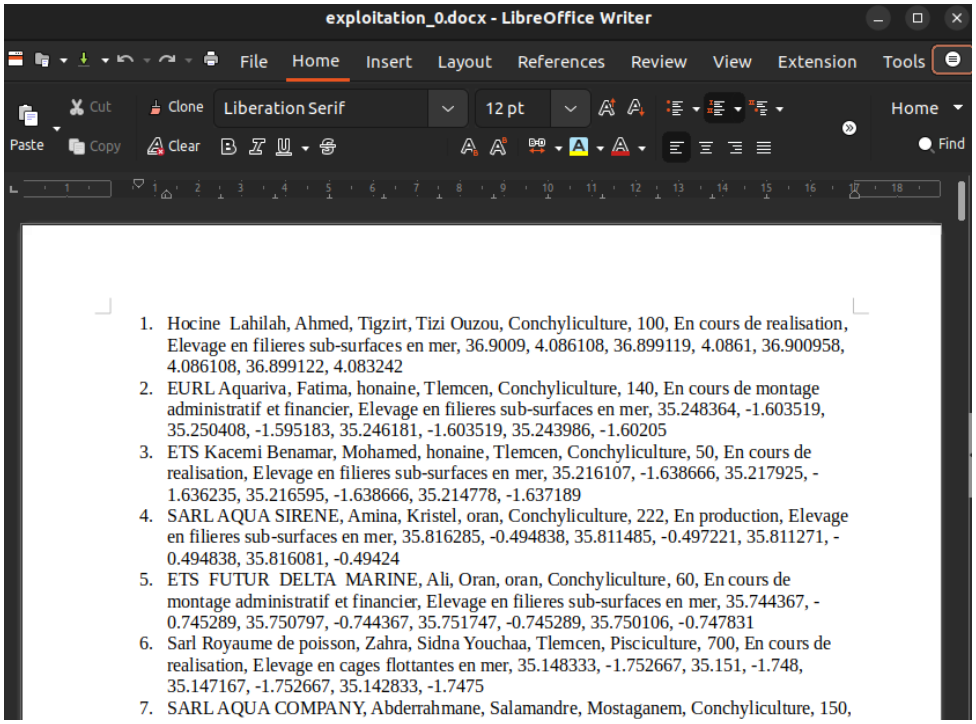


Figure 4.3: Aperçu sur le contenu d'un fichier Word

Quant au deuxième jeu de données, il s'agit d'un programme qui génère indéfiniment des objets JSON contenant des valeurs aléatoires sur les mesures de température, le taux d'oxygène et le taux de pH. Ainsi, ce jeu de données simule le comportement d'un capteur IoT (voir figure 4.4).

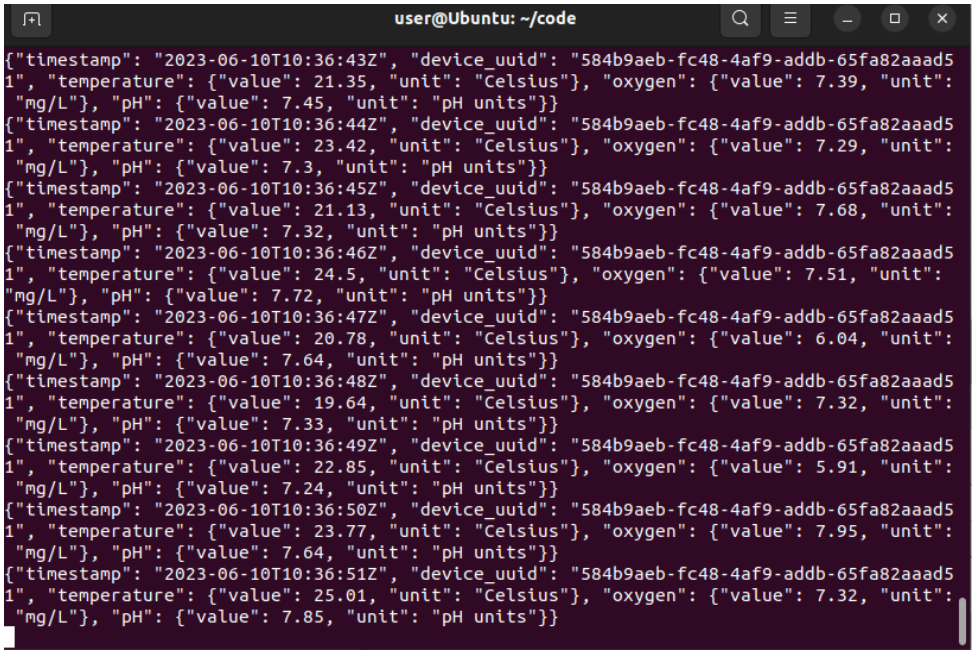


Figure 4.4: Aperçu sur les données générées de la simulation des capteurs IoT

4.5 Description des modules de traitement

4.5.1 Ingestion en temps réel avec Apache Nifi

Cette partie consiste à implémenter le processus d'ingestion en temps réel. Nous avons utilisé HiveMQ (une plateforme de messagerie évolutive et fiable pour l'IoT) comme serveur MQTT. Les différents capteurs publient des messages sur les sujets "*temp*" et "*pH*" vers le courtier MQTT (HiveMQ) (voir le schéma 4.5). Ces données sont ensuite interceptées par Apache Nifi qui s'abonne aux sujets précédents. Enfin, les données collectées sont ensuite stockées dans MongoDB.

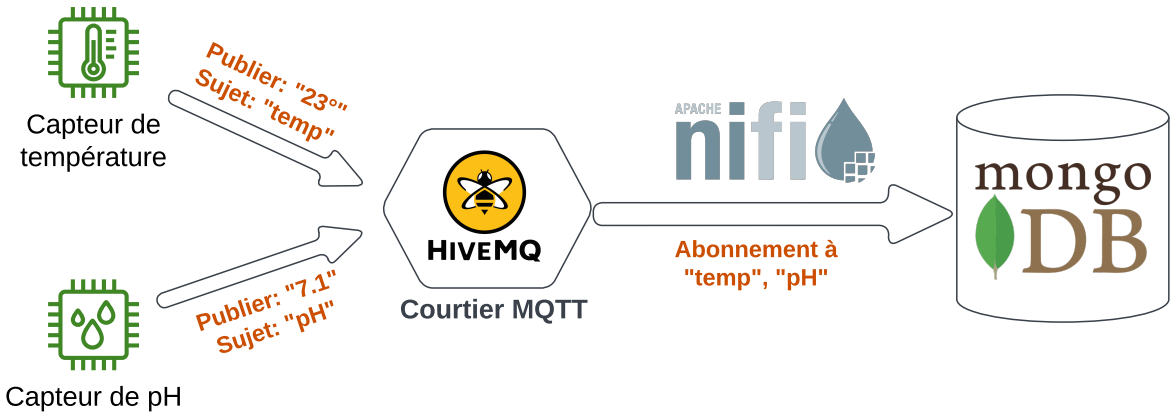


Figure 4.5: Flux de données en temps réel depuis les capteurs vers MongoDB via MQTT et Apache NiFi

Apache Nifi possède des unités de traitements configurables appelées processeurs. Ces derniers effectuent des opérations sur les flux de données tels que la lecture, l'écriture, le filtrage ou la transformation des données.

Afin de collecter et stocker les données en continu, deux processeurs sont utilisés :

— **"ConsumeMQTT"** : Il se connecte au broker MQTT (HiveMQ dans notre cas) et il récupère les messages publiés par les capteurs.

— **"PutMongoRecord"** : Il insère les données collectées dans une collection MongoDB 'la zone de diffusion).

La figure 4.6 illustre ces processeurs.

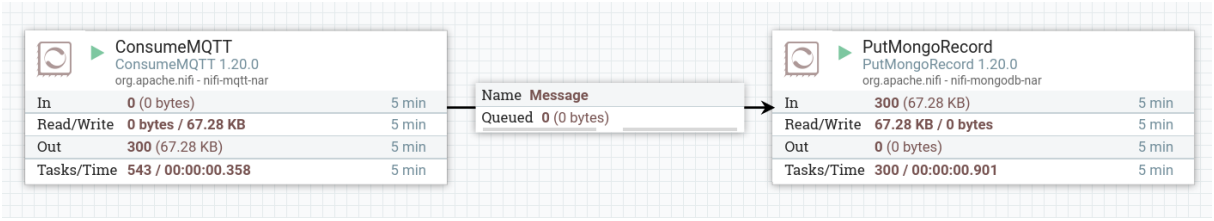


Figure 4.6: Processus d'ingestion en temps réel avec Apache Nifi

4.5.2 Parallélisation avec Apache Spark

Les étapes suivantes expliquent la parallélisation du processus d'extraction de données :

- Le répertoire contenant les fichiers Word est chargé dans un DataFrame Spark, qui est une structure de données tabulaire distribuée.
- Le DataFrame est divisé en partitions, où chaque partition contient une partie des données à traiter. Chaque partition peut être traitée indépendamment par un nœud du cluster.
- Une fonction *extract_ data* est définie pour extraire les données de chaque document Word.
- En parcourant chaque paragraphe d'un document, la fonction extrait le texte du paragraphe et supprime les espaces vides : la virgule est utilisée comme délimiteur et chaque valeur est nettoyée des espaces supplémentaires.
- Si le nombre de valeurs extraites correspond au nombre de colonnes attendues, cela signifie que les données sont valides.
- Pour chaque fichier, la fonction *extract_ data* est exécutée pour extraire les données en parallèle.
- Les données extraites sont converties en format JSON, puis enregistrées dans une collection MongoDB.

La parallélisation est le mécanisme clé utilisé par Apache Spark pour accélérer le traitement des données. Il divise automatiquement les tâches en plusieurs partitions et les exécute sur différents cœurs ou nœuds, permettant ainsi un traitement simultané des données.

4.6 Présentation de l'interface utilisateur

La figure 4.7 qui suit illustre une page web dédiée au processus ECLT.

Figure 4.7: Interface utilisateur dédiée au processus ECLT

Le bouton « Browse » permet de naviguer le système de fichier local, puis sélectionner les fichiers à charger dans le lac de données. En appuyant sur le bouton « Charger les données », les fichiers sélectionnés seront classés et organisés par type et par format dans la zone brute du lac de données, représentant ainsi les phases ECL « Extract Classify Load ».

La prochaine étape consiste à transformer les données chargées. Le bouton « Transformer les données » permet d'exécuter le processus d'extraction de données depuis les fichiers bruts et de les transformer en documents BSON stockés dans MongoDB, représentant alors la phase « Transform » de ECLT.

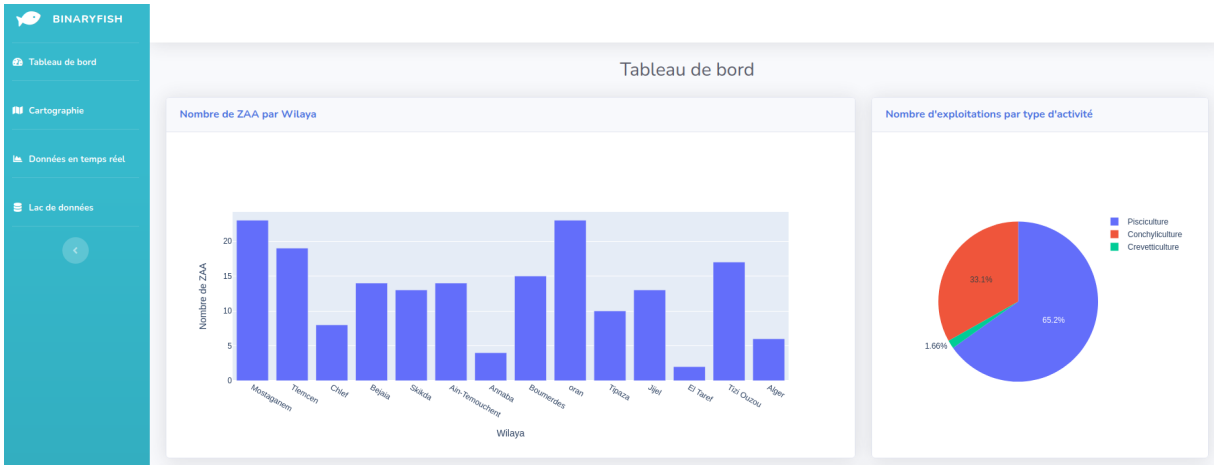


Figure 4.8: Tableau de bord de l'interface utilisateur

Une fois les données chargées dans la zone conforme, elles seront visualisées sous forme de graphiques tels que des diagrammes à barres, des diagrammes circulaires et des graphiques à bulles. Celles-ci illustrent des mesures comme le nombre de ZAA par Wilaya, nombre d'exploitations par type d'activité, capacité de production par Wilaya et par type d'élevage, etc., comme présenté dans la figure 4.8.

La page web suivante reflète une carte interactive comme le montre la figure 4.9. Chaque point représente une exploitation où nous avons la possibilité de filtrer l'affichage des fermes aquacoles par type d'activité (Pisciculture, Conchyliculture et Crevetticulture).

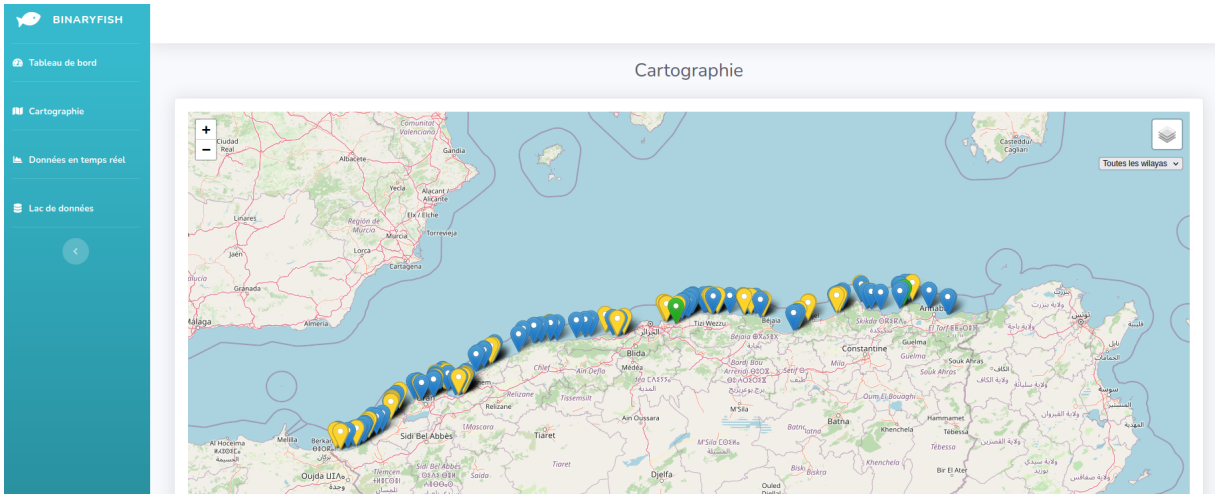


Figure 4.9: Carte géographique des fermes aquacoles

L'interface permet aussi de filtrer les concessions par Wilaya. Par défaut, toutes les Wilayas sont affichées, mais il est possible de sélectionner une Wilaya spécifique. En outre, lorsque nous passons la souris sur les points des fermes aquacoles, les informations de celles-ci surgissent sous la forme d'une fenêtre contextuelle ("pop-up").

La dernière page de l'interface traite la partie d'ingestion, de stockage et de restitution des données en temps réel (consulter la figure 4.10).

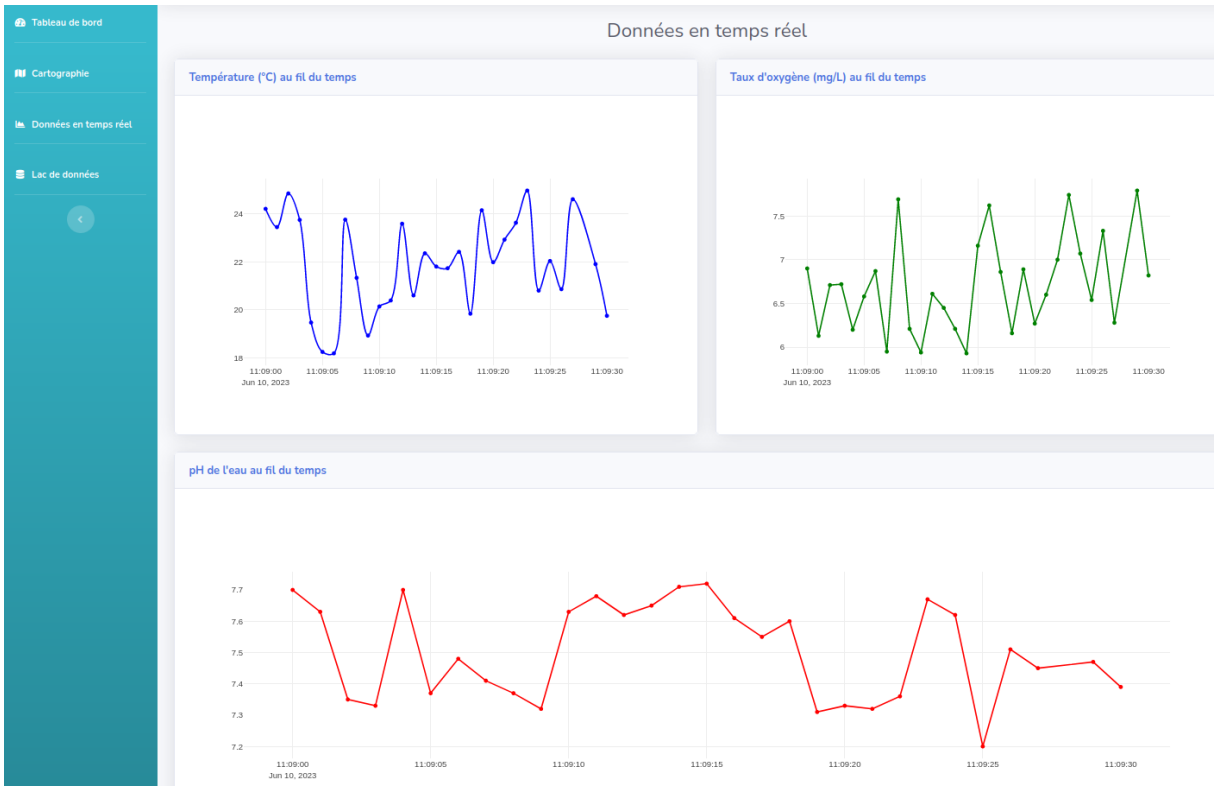
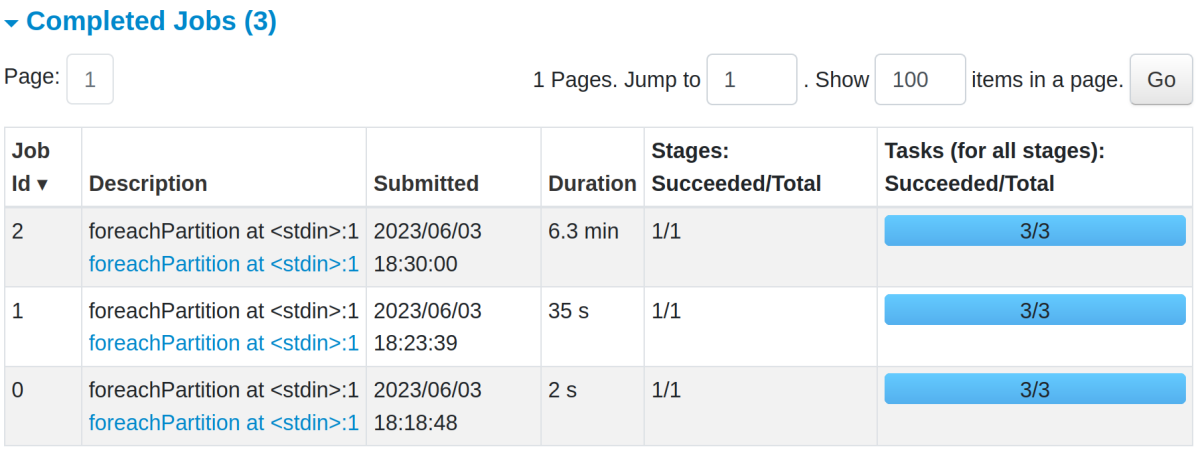


Figure 4.10: Visualisation des données générées par les capteurs en temps réel

Des graphiques en ligne sont utilisés pour illustrer le changement progressif et au fil du temps de la température (en degré Celsius), du taux d'oxygène (en mg/L) et du taux de pH, permettant ainsi une analyse dynamique des conditions aquacoles.

4.7 Tests et évaluation

La figure 4.11 illustre les tâches Spark (jobs) avec leur durée d'exécution.



▾ Completed Jobs (3)

Page: 1 1 Pages. Jump to 1 . Show 100 items in a page. Go

Job Id ▾	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
2	foreachPartition at <stdin>:1 foreachPartition at <stdin>:1	2023/06/03 18:30:00	6.3 min	1/1	3/3
1	foreachPartition at <stdin>:1 foreachPartition at <stdin>:1	2023/06/03 18:23:39	35 s	1/1	3/3
0	foreachPartition at <stdin>:1 foreachPartition at <stdin>:1	2023/06/03 18:18:48	2 s	1/1	3/3

Figure 4.11: Aperçu des performances des tâches Spark en fonction de leur durée d'exécution

Pour la première tâche Spark, nous avons utilisé 100 fichiers Word de 100 lignes chacun. Chaque ligne est capturée, transformée en document, puis stockée dans MongoDB formant un total de 10 000 documents. La durée de la tâche est de 2 secondes.

Avec une durée d'exécution de 35 secondes, la deuxième tâche a traité 10 000 fichiers Word, équivalent à 1 000 000 (un million) de documents.

Pour 100 000 fichiers Word (comme dans les cas précédents, chaque fichier contient 100 lignes) équivalent à 10 000 000 (dix millions) de documents stockés, la troisième tâche a duré 378 secondes (6.3 minutes).

Le tableau 4.2 suivant résume ces résultats.

Tâche N	Nombre de fichiers Word	Nombre de lignes par fichiers	Nombre de documents	Temps d'exécution (secondes)	Documents / seconde
1	100	100	10,000	2	5000
2	10,00	100	1,000,000	35	28751.4286
3	100,000	100	10,000,000	378	26455.0265

Tableau 4.2: Performance des tâches Spark selon le nombre de fichiers Word et de documents traités

La figure 4.12 ci-dessous présente un graphique à barres illustrant les résultats des trois tâches par rapport au nombre de documents traités par seconde.

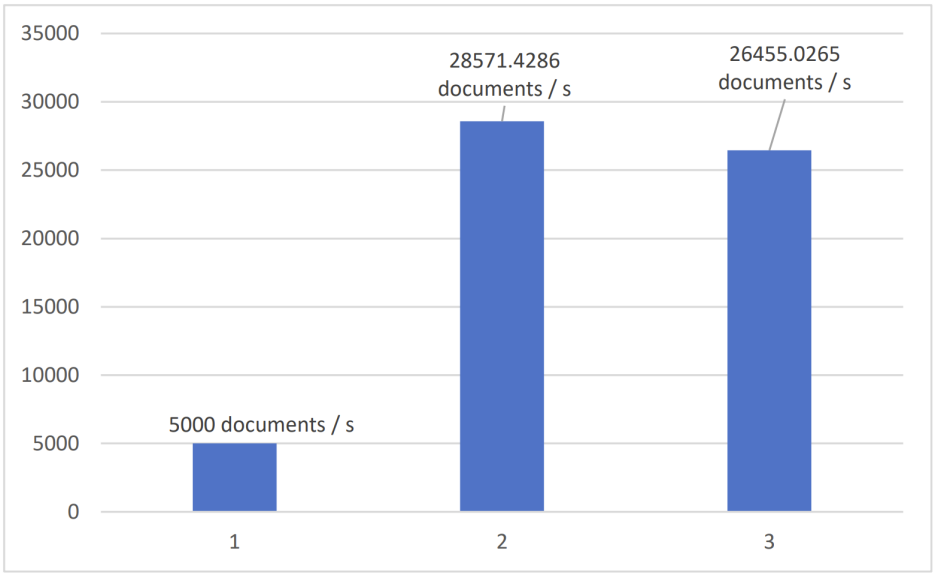


Figure 4.12: Résultats des tests de parallélisation de Spark

En considérons la deuxième tâche comme point de référence, avec un temps d'exécution de 35 secondes, équivalent à 28 571.4 documents par seconde, nous remarquons que les performances de la première tâche (5 000 documents / s) sont clairement inférieures à celles de la deuxième. Cela peut se justifier par la nature de Spark qui alloue automatiquement les ressources conformément à la taille du jeu de données : 10 000 documents sont relativement petits par rapport à 1 000 000 de documents.

Théoriquement, avec toujours la deuxième tâche comme valeur de référence, la troisième tâche qui a duré 378 secondes devrait prendre 350 secondes. Nous remarquons donc un déclin de performance de 8% (différence de 28 secondes). Mais, étant donné l'écart de taille entre les deux jeux de données, ces résultats restent intéressants.

4.8 Conclusion

Pour conclure, notre approche a été pleinement concrétisée dans ce chapitre.

Nos réalisations consistent en :

— Mise en œuvre des différentes zones de notre lac de données (zone brute, zone conforme, zone de diffusion) ;
— Application du processus ECLT ;
— Génération de deux jeux de données afin d'évaluer notre approche ;
— Application de la parallélisation du processus d'extraction et de transformation de données avec Apache Spark ;
— Ingestion, stockage et restitution du flux de données en temps réel ;
— Réalisation des tests et évaluation des résultats ;
— Mise en œuvre d'une interface utilisateur web interactive qui donne accès aux différentes fonctionnalités du système.

L'ensemble de ces réalisations répond pleinement aux besoins du CNRDPA.

Conclusion Générale

Conclusion

Notre projet de fin d'études nous a emmené à travailler sur la mise en place d'un système d'aide à la décision basé sur un Data Lake. Notre présente contribution a pour but de faciliter l'exploitation, l'analyse et la visualisation des données de l'aquaculture marine. Ceci, permettra une meilleure prise de décision dans le domaine de l'investissement de l'aquaculture marine à l'échelle nationale.

Avant d'entamer la conception de notre système, nous avons mené une étude comparative entre les entrepôts de données (DW) et les lacs de données (DL). La flexibilité et l'évolutivité de ce dernier constituent les principaux critères pour le choix de cette solution de stockage. L'étude des travaux connexes, dans le domaine de l'informatique décisionnels, a été un enseignement précieux qui nous a éclairé dans la conception et l'implémentation de notre système.

Après une étude de l'existant, nous avons conçu une architecture fonctionnelle et proposé un nouveau schéma d'ingestion de données : ECLT (Extract Classify Load Transform). Nous avons travaillé sur deux types d'ingestion de données : par lots et en temps réel.

Afin de mieux comprendre les données brutes, nous avons adopté une modélisation orientée objet, et nous avons ensuite procédé à la migration de ces données vers un modèle orienté document.

Concernant l'architecture technique, nous avons mis en œuvre les différentes zones de notre lac de données (zone brute, zone conforme et zone de diffusion). Afin d'évaluer notre approche, nous avons généré deux jeux de données (données de capteurs IoT et fichiers Word). Nous avons réussi à simuler le comportement des capteurs IoT, ainsi que la capture et le stockage de leurs données en temps réel.

Le produit final de ce projet de fin d'études est un système basé sur un Data Lake qui présente plusieurs fonctionnalités clés, notamment la capacité de collecter, stocker et présenter les données diffusées en continu par les capteurs IoT. Ce système comprend également une interface utilisateur web interactive, donnant accès à différentes fonctionnalités

telles qu'une carte géographique, un tableau de bord et une page pour la visualisation des données en temps réel. Enfin, pour évaluer les performances de la parallélisation, nous avons réalisé des tests sur le processus de capture et transformation avec Apache Spark. Les résultats obtenus se sont avérés intéressants.

Cependant, nous avons rencontré de nombreuses difficultés à la réalisation de ce projet, principalement le facteur temps. Savoir maîtriser plusieurs outils technologiques dans un délai limité s'est avéré difficile. Mais, grâce à notre persévérance, nous avons pu surmonter ces difficultés.

En conclusion, à travers ce projet, nous espérons développer le domaine de l'aquaculture marine et ainsi que l'aquaculture continentale lors de la disponibilité de ces données. Notre travail sur la mise en place d'un Data Lake permettra de stocker et d'analyser des données massives et hétérogènes, offrant ainsi une prise de décision pertinente.

De plus, l'implémentation réussie de l'ingestion en temps réel démontre la faisabilité de notre approche. Nous espérons que ce système servira comme un outil de surveillance des conditions aquacoles en temps réel ; chose qui n'existe pas encore en Algérie.

La surveillance des données environnementales permettra une prévention des risques d'accidents et une gestion efficace dans le domaine aquacole, favorisant ainsi son développement économique.

Travaux futurs

Malgré avoir atteint les objectifs fixés, nous considérons que notre travail présente encore des possibilités d'amélioration. Voici quelques propositions :

- Application d'un algorithme de Traitement Automatique du Langage (TAL) sur des documents déstructurés pour extraire des informations plus pertinentes ;
- Parallélisation du nouveau processus d'intégration proposé ECLT ;
- Chargement des données sources sous format Parquet ou Avro dans la zone brute (HDFS), pour un stockage efficace ;
- Mise en œuvre d'une algèbre OLAP (Online Analytical Processing) adaptée aux structures NoSQL de la zone conforme pour amplifier les capacités d'analyses.

Bibliographie

[1] Sarathkumar Rangarajan, Huai Liu, Hua Wang, and Chuan-Long Wang. Scalable Architecture for Personalized Healthcare Service Recommendation Using Big Data Lake. pages 65–79, 2018.

[2] Amr A Munshi and Yasser Abdel-Rady I Mohamed. Data lake lambda architecture for smart grids big data analytics. *IEEE Access*, 6 :40463–40471, 2018.

[3] David Sarramia, Alexandre Claude, Francis Ogereau, Jérémy Mezhoud, and Gilles Mailhot. CEBA : A data lake for data sharing and environmental monitoring. *Sensors (Basel)*, 22(7) :2733, April 2022.

[4] El Mehdi Ouafiq, Rachid Saadane, Abdellah Chehri, and M Wahbi. Data lake conception for smart farming : A data migration strategy for big data analytics. pages 191–201, 2022.

[5] Sarah Benjelloun, Mohamed El Mehdi El Aissi, Younes Lakhrissi, and Safae El Haj Ben Ali. Data lake architecture for smart fish farming data-driven strategy. *Applied System Innovation*, 6(1) :8, Jan 2023.

[6] Pegdwendé Sawadogo and Jérôme Darmont. On data lake architectures and metadata management. *Journal of Intelligent Information Systems*, 56 :97–120, 2021.

[7] Cédrine Madera. *L'évolution des systèmes et architectures d'information sous l'influence des données massives : les lacs de données*. Theses, Université Montpellier, November 2018. [Departement_IRSTEA]Territoires [TR1_IRSTEA]SYNERGIE [Axe_IRSTEA]TETIS-SISO [Encadrant_IRSTEA]Miralles, A.

[8] Jean-Fabrice Lebraty. Les systèmes décisionnels. In Akoka, A, Comyn-Wattiau, and I., editors, *Encyclopédie de l'informatique et des systèmes d'information*, pages 1338–1349. Vuibert, 2006.

[9] Pravin Chandra and Manoj Gupta. Comprehensive survey on data warehousing research. *International Journal of Information Technology*, 10, 12 2017.

[10] W. Inmon. *Building the Data Warehouse*. Wiley Publishing, Indianapolis, USA, 4th edition, 2005.

[11] Mahfoud Bala, Omar Boussaid, and Zaia Alimazighi. A fine-grained distribution approach for ETL processes in big data environments. *Data Knowl. Eng.*, 111 :114–136, September 2017.

[12] Arsia Amir-Aslani and Ricky Bhajun. Les 7 «v» piliers du big data. 11 2016.

[13] Marilex Rea Llave. Data lakes in business intelligence : reporting from the trenches. *Procedia Computer Science*, 138 :516–524, 2018. CENTERIS 2018 - International Conference on ENTERprise Information Systems / ProjMAN 2018 - International Conference on Project MANagement / HCist 2018 - International Conference on Health and Social Care Information Systems and Technologies, CENTERIS/ProjMAN/HCist 2018.

[14] Hassan Alrehamy and Coral Walker. Personal data lake with data gravity pull. 08 2015.

[15] D. P. Acharjya and Kauser Ahmed P. A survey on big data analytics : Challenges, open research issues and tools. *International Journal of Advanced Computer Science and Applications*, 7(2), 2016.

[16] B. Devlin. Thirty years of data warehousing. *Business Intelligence Journal*, 23(11) :12–24, 2018.

[17] B. Sharma and an O'Reilly Media Company Safari. *Architecting Data Lakes, 2nd Edition*. O'Reilly Media, Incorporated, 2018.

[18] Mahfoud Bala, Oussama Mokeddem, Omar Boussaid, and Zaia Alimazighi. Une plateforme etl parallèle et distribuée pour l'intégration de données massives. *Revue des Nouvelles Technologies de l'Information*, RNTI-E-28 :455–460, 01 2015.

[19] Dipti M Tayade. Comparative study of etl and e-lt in data warehousing. *Int. Res. J. Eng. Technol*, 6 :2803–2807, 2019.

[20] David Taniar and Wenny Rahayu. Data lake architecture. In *Advances in Internet, Data and Web Technologies*, pages 344–357. Springer International Publishing, Cham, 2021.

[21] Mohamed El Mehdi El Aissi, Sarah Benjelloun, Yassine Loukili, Younes Lakhrissi, Abdessamad El Boushaki, Hiba Chougrad, and Safae Elhaj Ben Ali. Data lake versus data warehouse architecture : A comparative study. In *Lecture Notes in Electrical Engineering*, Lecture notes in electrical engineering, pages 201–210. Springer Singapore, Singapore, 2022.

[22] Adnan Masood, Adnan Hashmi, Adnan Masood, and Adnan Hashmi. Text analytics : The dark data frontier. *Cognitive Computing Recipes : Artificial Intelligence Solutions Using Microsoft Cognitive Services and TensorFlow*, pages 189–224, 2019.

[23] Kosovare Sahatqija, Jaumin Ajdari, Xhemal Zenuni, Bujar Raufi, and Florije Ismaili. Comparison between relational and nosql databases. In *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pages 0216–0221, 2018.

[24] Pallavi Sethi and Smruti R. Sarangi. Internet of things : Architectures, protocols, and applications. *Journal of Electrical and Computer Engineering*, 2017 :1–25, 2017.

[25] MF Sanner. Python : a programming language for software integration and development. *Journal of molecular graphics and modelling*, 17(1) :57—61, February 1999.

[26] Karol Wnęk and Piotr Boryło. A data processing and distribution system based on apache nifi. *Photonics*, 10(2) :210, Feb 2023.

[27] Dhruba Borthakur. Hdfs architecture. *Document on Hadoop Wiki. URL http ://hadoop. apache. org/common/docs/r0*, 20, 2010.

[28] Salman Salloum, Ruslan Dautov, Xiaojun Chen, Patrick Xiaogang Peng, and Joshua Zhexue Huang. Big data analytics on apache spark. *International Journal of Data Science and Analytics*, 1 :145–164, 2016.

[29] Veronika Abramova and Jorge Bernardino. Nosql databases : Mongodb vs cassandra. In *Proceedings of the international C* conference on computer science and software engineering*, pages 14–22, 2013.

[30] Igor Stančin and Alan Jović. An overview and comparison of free python libraries for data mining and big data analysis. In *2019 42nd International convention on information and communication technology, electronics and microelectronics (MIPRO)*, pages 977–982. IEEE, 2019.