REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE AMAR TELIDJI LAGHOUAT

Optimisation des méthodes d'analyse et de conception, pour l'extraction de connaissance et la visualisation de Big Data pour la prise de décision

Thése présentée

par

Mohammed el Habib MAICHA

En vue d'obtenir le titre de

Docteur en Informatique

Membres du jury

Mr. Mohamed Bachir YAGOUBI	Prof	UATL	Président
Mr. Younes GUELLOUMA	M.C.A	UATL	Examinateur
Mr. Slimane BELLAOUAR	M.C.A	UG	Examinateur
Mr. Amine KHALDI	M.C.A	UKMO	Examinateur
Mr. OUINTEN Youcef	Prof	UATL	Directeur de thèse
Mr. ZIANI Benameur	M.C.A	UATL	Co-directeur de thèse

Dédicace

إهـــداء

هذا العمل مهدى الى عائلتي و أصدقائي

Je dédie ce travail de thèse à ma famille et à mes amis.

I dedicate this thesis work to my family and friends.

Dedico este trabajo de tesis a mi familia y amigos.

Dedico questo lavoro di tesi alla mia famiglia e ai miei amici.

Bu tez çalışmasını aileme ve arkadaşlarıma ithaf ediyorum.

我将这篇论文献给我的家人和朋友。

Remerciements

تشكـــــرات

Je tiens à exprimer ma sincère gratitude au jury. En commençant par mes directeurs de thèse, j'aimerais remercier le président et les examinateurs d'avoir pris le temps de lire et d'évaluer mon travail.

Dear reader,
thank you for including this document in your literature search.
I hope you find it beneficial.

عزيزي القارئ
شكرا لتضمين هذه الرسالة في البحث الخاص بك. أتمنى أن تجدها مفيدة

Cher lecteur,
je vous remercie d'avoir inclus ce document dans votre recherche bibliographique. J'espère que vous le trouverez bénéfique.

Abstract

The adoption of Big Data systems by the companies is relatively new, although the data modeling and system design are ages old. Despite the fact that traditional databases are built on solid foundations, they cannot handle the swift and massive flow of data coming from multiple different sources. Herein, NoSQL databases are an inevitable alternative. However, these systems are schemaless compared to traditional databases. It is important to emphasize that schemaless does not mean no-schema which would mean that NoSQL databases do not need modeling. Hence, there is a need for conceptual models to define the data structure in these databases. This thesis sheds a light on the importance of the UML in showing how to store Big Data described throw Meta-models within NoSQL databases.

We propose a novel Big Data modeling methodology for NoSQL databases called UML4NoSQL, which is independent of the target system, and supports the four Big Data characteristics : Variety, Volume, Velocity, and Veracity (4V's). The approach relies on the UML blocks with a data-up technique ; it starts with a Use-Case and the Class diagram resulting from the understanding of the data at hand and the definition of the developer's strategies while focusing on the user's needs. To illustrate our approach, we take a case study from health care domain. We show that our approach produces designs that can be implemented on NoSQL Document-oriented system with respect to Big Data 4V's.

Keywords : Big data, UML, Database modelling, NoSQL, Document-store, UML4NoSQL.

Résumé

L'adoption des systèmes Big Data par les entreprises est une tendance assez récente, bien que la modélisation des données et la conception des systèmes datent de plusieurs années. Malgré le fait que les bases de données traditionnelles reposent sur des fondations solides, elles ne peuvent pas gérer le flux rapide et massif de données provenant de multiples sources différentes. C'est pourquoi les bases de données NoSQL constituent une alternative inévitable. Cependant, ces systèmes sont dépourvus de schémas par rapport aux bases de données traditionnelles. Il est important de souligner que l'absence de schéma ne signifie pas la disparition du schéma, ce qui signifierait que les bases de données NoSQL n'ont pas besoin d'être modélisées. Il existe donc un besoin de modèles conceptuels pour définir la structure des données dans ce type de bases de données. Cette thèse a pour but de mettre en évidence l'importance de l'UML pour montrer comment stocker les Big Data décrites par des méta-modèles dans des bases de données NoSQL.

Nous proposons une nouvelle méthodologie de conception de Big Data pour les bases de données NoSQL, appelée UML4NoSQL, qui est indépendante du système cible, et qui prend en charge les quatre caractéristiques du Big Data : Variété, Volume, Vélocité et Véracité (4V's). L'approche s'appuie sur les blocs UML avec une technique de data-up ; elle commence par un Use-Case et le diagramme de classe résultant de la compréhension des données à disposition et de la définition des stratégies du développeur tout en se concentrant sur les besoins de l'utilisateur. Pour illustrer notre approche, nous prenons un cas d'étude dans le domaine de santé. Nous montrons que notre approche produit des schémas qui peuvent être mis en œuvre sur un système NoSQL orienté document en respectant les 4V du Big Data.

Mots clés : Big data, UML, modélisation de bases de données, NoSQL, Document-store, UML4NoSQL.

Table des matières

Table des figures

Liste des tableaux

Liste des abbréviations

NoSQL **N**ot **o**nly SQL

UML **U**nified **M**odeling **L**anguage

XML **eX**tencible **M**arkup **L**anguage

UML4NoSQL **UML 4**pour **NoSQL**

4V's Variété, Volume, Vélocité et Véracité

SGBD **S**ystème de **G**estion de **B**ase **D**onnées

SI **S**ystème d'**I**information

RSL **R**evue **S**ystématique de **L**ittérature

CDM **C**onceptual **D**ata **Model**

NoREL **N**ot **REL**ationnel

CAP **C**onsistency **A**vailability **P**artition

BLOB **B**inary **L**arge **OB**ject

ACID **A**tomicité **C**ohérence **I**solation **D**urabilité

SGBDR **S**ystème de **G**estion de **B**ase **D**onnées **R**elationnelle

OOAD **O**bject **O**riented **A**nalyse **D**esign

OMT **O**bject **M**odeling **T**echnique

OOSE **O**bject **O**riented **S**oftware **E**ngineering

OOA **O**bject **O**riented **A**nalysis

QR **Q**uestion de **Recherche**

ER **E**ntité **R**elationnelle

BI **B**usiness **I**ntelligence

MCD **M**odèle **C**oceptuel des **D**onnées

MLOD **M**odèle **L**ogique **O**rienté **D**ocuments

DME **D**ossiers **M**édicaux **E**lectroniques

Vol **Vol**ume

Vel **Vel**ocité

Var **Var**iété

Ver **Ver**acité

Chapitre 1

Introduction

"Data is not information, information is not knowledge, knowledge is not understanding, understanding is not wisdom"

(Clifford Stoll)

1.1 Contexte et motivations

Contrôlées par des valeurs électriques de "off" ou "on", deux bits "zéro" et "un", sont à la base de tout dans notre monde. Nous appelons ces petits incréments les "données". Ces dernières n'ont cessé de croître, en particulier au cours de la dernière décennie. La perception générale est que nous sommes submergés de données, ce qui fait de la capacité à modéliser, stocker, traiter, analyser, interpréter, consommer et agir sur des données massives appelées communément "Big data" une préoccupation majeure. Dans cette nouvelle ère, le monde de recherche et le secteur industriel s'intéressent de plus en plus au traitement des Big Data.

Le terme "Big Data", désigne généralement les volumes extrêmement importants de données, structurées ou non structurées, que les entreprises peuvent aujourd'hui recueillir et tenter d'analyser de manière significative afin d'améliorer les activités de prise de décision et de découverte de connaissances. Cela a nécessité le développement de nouveaux

outils et processus d'analyse des données, ainsi que d'importantes applications en science et en gestion.

Le big data est considéré comme une nébuleuse d'informations nourris par une grande variété de sources de données. Toute requête sur internet via un moteur de recherche, tout achat sur un site e-commerce, chaque intervention dans un réseau social, et chaque usage d'un objet connecté alimente cette nébuleuse imparable. Sans négliger les informations fournies par les appareils mobiles dont nous sommes devenus progressivement dépendants tels les smartphones, les cartes bancaires, etc [1].

Dans la seconde moitié du vingtième siècle, les systèmes de gestion de bases de données relationnelles (SGBD relationnels) étaient considérés comme des outils "one-size-fits-all" d'analyse de données[1]. Toutes les données dans leur modèle relationnel sont représentées en termes de tuples, qui sont regroupées dans un ensemble de tables liées les unes aux autres. Chaque table possède un schéma prédéfini auquel tous les tuples de la table doivent adhérer. Au fil des années, les SGBD relationnels ont supporté avec succès une large gamme d'applications centrées sur les données, avec des caractéristiques et des exigences variables.

Cependant, à l'aube du XXIe siècle, les SGBD traditionnels deviennent inadéquats dans de nombreux scénarios d'application. La croissance exponentielle des données est l'une des principales raisons pour lesquelles les SGBD traditionnels sont dépassés. En tant qu'entreprise axée sur les données, Google traite plus de 3,5 milliards de requêtes par jour et stocke plus de 10 exaoctets de données. Facebook collecte quotidiennement 600 téraoctets de données, dont 4,3 milliards de contenus, 5,75 milliards de "j'aime" et 350 millions de photos. Selon certaines estimations, 64,2 zettaoctets de données existent actuellement [2].

1. https ://medialude.com/article-big-data/
2. https ://waterfordtechnologies.com/big-data-interesting-facts/

Les SGBD relationnelles présentaient deux limitations majeures en termes d'évolution des bases de données. La première était l'incapacité de s'étendre horizontalement, et la seconde la difficulté à gérer les données que l'industrie génère en permanence. L'émergence des bases de données NoSQL a résolu ces problèmes en offrant au monde la flexibilité des schémas. L'une des principales raisons de leur popularité contrairement aux bases de données relationnelles, les bases de données NoSQL utilisent des structures de données orientées document, orienté colonnes, graphe ou des paires clé-valeur. En faisant une projection du big data sur la définition d'un système d'information (SI), "un ensemble organisé de ressources qui permet de collecter, stocker, traiter et distribuer de l'information"[2], on observe que le big data peut être considérer comme un SI avec des différence au niveau de : la variété des sources de données, la volumétrie des données et du non structuration de ces dernières et l'utilisation des données qui sont plus des données décisionnelles.

Le big data est un système informatique et comme tout autre système il nécessite une modélisation pour permettre de cerner ses frontières et ses utilisations. D'une part, nous savons que dans un environnement Big data, les données sont présentes en permanence. La question qui se pose, est de savoir ce que nous prévoyons de faire et de tirer profit de ces données. D'autre part, l'UML (Unified Modeling Language) a fait ses preuves en tant que langage de modélisation des systèmes informatiques traditionnels, web et même des systèmes plus complexes résultant de l'évolution des besoins de plus en plus complexes des entreprises contemporaines.

1.2 Problèmes et objectifs

En admettant que les systèmes big data nécessitent une modélisation pour aider à identifier leurs limites et leurs utilisations. Et, le fait qu'UML peut être utilisé pour concevoir des bases de données NoSQL, alors de nombreux défis sont à relever :

— Le manque d'expérience des développeurs, en raison de la nouveauté, de la variété et de la complexité des technologies existantes. Il n'existe pas de méthodologie claire et complète pour guider le développement du projet big data.

— L'absence d'approches qui facilitent l'intégration de données provenant de différentes sources dans les projets big data.

— Il est crucial d'identifier la possibilité de transformer les diagrammes UML de manière à ce qu'ils puissent être utilisés dans le processus de conception NoSQL.

— Le processus doit fournir des techniques et des directives complètes pour une méthodologie de modélisation de données efficace pour les bases de données NoSQL.

— Le processus doit finalement conduire à un modèle de données physique complet et valide à mettre en œuvre dans les systèmes NoSQL.

L'hypothèse de cette thèse est que le développement de projets big data échoue principalement en raison du manque de directives pour aborder leur conception de manière systématique. De ce fait, le développement d'une méthodologie systématique et formelle pour guider les développeurs de Big data, contribuera à améliorer le succès de tels projets. Étant donné l'ampleur des projets big data, et en tenant compte des aspects discutés ci-dessus, nous concentrerons notre travail sur les objectifs suivants :

— Conception d'une architecture de données big data. En systématisant et en formalisant le processus de conception, guidé par l'analyse des besoins des applications cibles en considérant les 4V du big data.

— Une étude systématique de la litérature sur les approches consacrées au développement de méthodologies et d'outils destinés à la conception de bases de données NoSQL.

— Un benchmark standard pour l'évaluation de la performance des méthodes de conception de bases de données NoSQL.

1.3 Contributions

Sur la base de la définition de la problématique de recherche et de ses défis, cette thèse apporte les contributions clés suivantes.
Commençons par la première contribution qui consiste à élaborer une revue systématique de la littérature (RSL) selon la démarche de Kitchenham [3]. Cette RSL regroupe les méthodes de conception des bases de données NoSQL, du point de vue de la méta-modélisation. En effet, les directives de Kitchenham résument les étapes de ce processus itératif en trois principales phases : *Planification de la revue, Conduite de la revue, Compte-rendu de la revue.*
Les étapes associées à *la planification de la revue* sont les suivantes :

— Identification du besoin d'une revue.

— Élaboration d'un protocole d'examen.

Les étapes associées à la conduite de la revue sont :

— Identification de la recherche.

— Sélection des études primaires.

— Évaluation de la qualité des études.

— Extraction et contrôle des données.

— Synthèse des données.

Le compte-rendu de la revue est une phase à une seule étape.

Suivant la démarche citée ci-dessus, nous avons regroupé les étapes les plus importantes entreprises (comme le montre la figure 1.1) :

— Élaborer un protocole.

— Définir les questions de recherche.

— Définir la stratégie de recherche.

— Définir les données à extraire de chaque étude primaire.

— Tenir à jour les listes des études incluses.

— Utiliser les lignes directrices pour la synthèse des données.

— Utiliser les directives pour l'établissement des rapports.

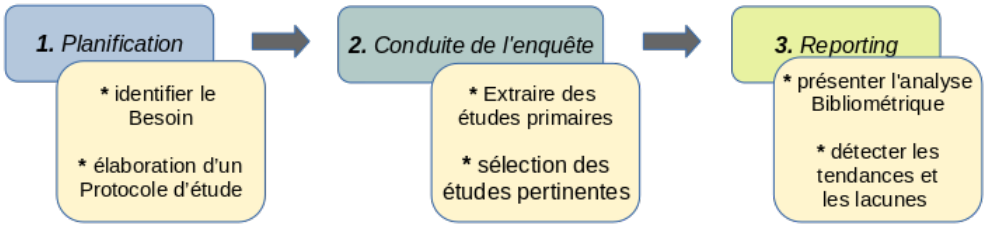


FIGURE 1.1 – Les phases d'une analyse systématique de la littérature

Comme seconde contribution, nous avons proposé une nouvelle approche de modélisation de données pour les bases de données NoSQL basée sur l'UML appelée **UML4NoSQL**. Dans notre approche, nous visons à fournir une bonne représentation des données de l'application dans une base de données NoSQL cible (Orientée document), qui prend en considération les principales caractéristiques du Big Data, connues sous le nom de 4V. L'approche s'appuie sur les blocs UML avec une technique de data-up ; elle commence par le diagramme Use-Case plus un diagramme de classe résultant de la compréhension des données à disposition et de la définition des stratégies du développeur tout en se focalisant sur les besoins de l'utilisateur.

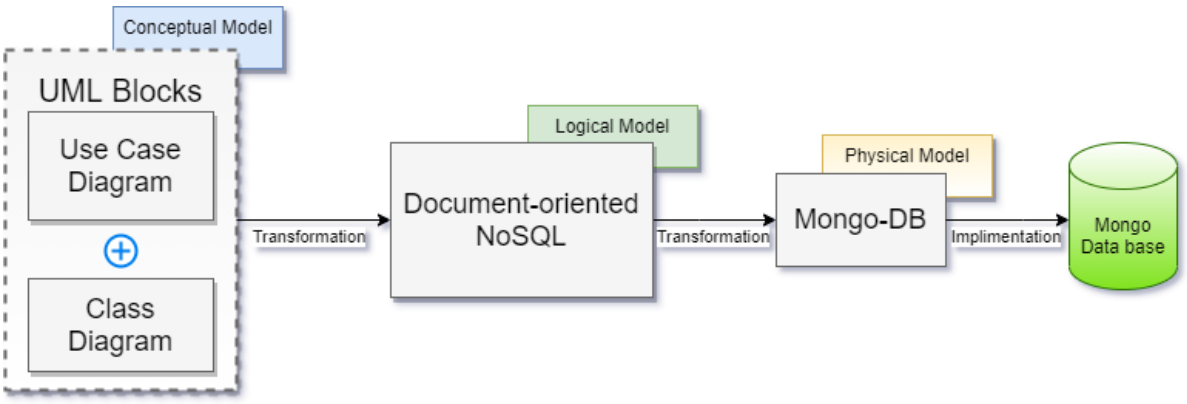


FIGURE 1.2 – Aperçu de la méthodologie proposée

Comme le montre la figure 1.2, notre approche s'articule autour de quatre axes majeurs. Le premier axe consiste en la compréhension des données à la disposition de l'utilisateur. Quant au second axe, l'objectif est de s'assurer que les stratégies du développeur et les besoins de l'utilisateur soient compréhensibles. L'élaboration d'un méta-modèle NoSQL constitue le troisième axe. Afin de maximiser la scalabilité, le résultat du quatrième axe, et un design qui évolue facilement.

1.4 Organisation du manuscrit

Le reste de ce manuscrit est organisé en six chapitres.

Le deuxième chapitre est consacré à la présentation des notions de base de la modélisation des bases de données dans le monde du big data. Nous présentons quelques fondements théoriques des systèmes de bases de données NoSQL. dans un deuxième lieu, nous présentons quelques concepts sur l'UML et son application dans la production du modèle conceptuel d'un système NoSQL.

Le chapitre 3 présente notre première contribution. Il s'agit d'une revue des méthodes de modélisation conceptuelle, logique et physique de base de données NoSQL basées sur l'UML. Nous mettons l'accent sur celles destinées à la modélisation des données dans les base de données NoSQL orienté document, car nous fonderons nos contributions sur leurs limites.

Notre deuxième contribution est introduite dans **le quatrième chapitre** sur deux parties. En commençant par donner des détails, sur comment les diagrammes UML peuvent être adaptés à la modélisation des bases de données NoSQL, en utilisant notre méthode UML4NoSQL. Dans un deuxième lieu, nous présentons les lignes directrices pour un passage réussi d'un diagramme de classe (CDM) à un modèle orienté document (MLOD) NOSQL.

Dans **le chapitre 5**, un exemple illustratif est fournis pour illuster l'application du UML4NoSQL et l'ensemble de schémas produits, dans le but de stocker correctement des données massives dans un système NoSQL de type document.

Enfin, **le chapitre 6** conclut cette thèse en deux sections. Dans une première partie, la conclusion générale qui rappelle les objectifs de la thèse et souligne les contributions réalisées et dans une deuxième partie, les perspectives du travail présenté selon une vision à court et à moyen terme.

Chapitre 2

Le cadre théorique

"Information is the oil of the 21st century, and analytics is the combustion engine" (Peter Sondergaard)

Dans ce chapitre, nous abordons le contexte théorique et les concepts clés liés à la modélisation des bases de données dans le monde du big data. Nous présentons d'abord quelques fondements théoriques de base sur les systèmes big data et les bases de données NoSQL. De plus, nous présentons quelques concepts sur l'UML et son application dans la production de modèle conceptuel d'un système NoSQL. Enfin, nous explorons également les effets du big data sur le processus de prise de décision.

2.1 Big data : un bref aperçu historique

Le concept de big data est apparu pour la première fois en 1997 dans un article rédigé par des chercheurs du Groupe Intel et de la NASA [4]. Ce travail a mis l'accent sur la taille croissante des données accessible et a expliqué comment les préparer pour le processus de visualisation. Quelques années plus tard, en 2001, le modèle "3Vs" a été utilisé pour fournir la première définition scientifique du terme big data [5], qui définissait le big data en plus du volume, par leur rythme rapide de génération de données et leur large éventail de types et d'attributs. Cependant, le big data n'est pas seulement une

question de données de très grande volume, ni une question d'échantillonnage dans de grands flux de données. Le point crucial du big data est qu'il change la façon d'aborder l'analyse des données, nécessitant de nouveaux modèles de traitement et de nouvelles représentations des connaissances.

De plus, les big data peuvent être divisées en deux catégories, celle des données structurées et celle des données non structurées. Au contraire des données non structurées, les données structurées sont filtrées, ont un format prévisible et sont définies par un ensemble de règles. Les sources de données non structurées peuvent être des réseaux sociaux, du texte, des photos ou des vidéos [6, 7]. Cette variété de sources rend les données ingérables pour un traitement traditionnel. Aussi, l'analyse de flux de données de grande taille augmente la probabilité de détecter des motifs et des anomalies intéressants comparativement à des flux de données plus modestes. Environ 80 à 90 % du big data étant des données non structurées, il est vital pour les entreprises de les donner un sens afin d'exploiter et de tirer profit des informations qu'elles recueillent [8].

2.1.1 Caractéristiques du big data

Bien que de nombreuses caractéristiques existent dans la littérature, les quatre caractéristiques fondamentales (voir figure 2.1) qui contribuent aux défis que posent le traitement et l'analyse des big data sont :

— ***Volume :***"*Big data is a matter of size*"- Des quantités massives de données qui dépassent les capacités des approches traditionnelles d'analyse des données.

Selon les statistiques, les internautes créent en moyenne 2,5 quintillions (1^{18}) d'octets de données par jour en 2022 [9]. La figure 2.2 montre le volume de données et d'informations créées, capturées, copiées et consommées dans le monde entre 2010 et 2025 (en zettaoctets), il s'agit toutefois d'un chiffre agrégé et très impressionnant. Cependant, aucun seuil n'a été fixé pour déterminer si le volume d'un ensemble de données particulier est "suffisamment important" pour être consi-

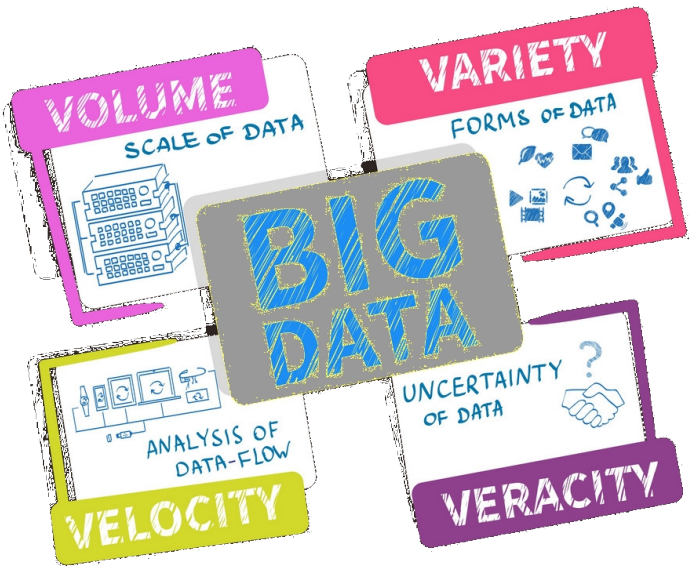


FIGURE 2.1 – Les 4 caractéristiques (4Vs) du big data

déré comme du big data. Le concept de big data implique une complexité et une hétérogénéité qui rendent difficile l'établissement d'un seuil, étant donné que sa définition peut varier en fonction du type de données, du moment (les données volumineuse d'aujourd'hui ne seront pas les mêmes que dans 5 ou 10 ans) ou de la manière la plus appropriée de les stocker, car selon leur type, des bases de données plus ou moins grandes seraient nécessaires [10].

— ***Variété :***"*Big data is extremely diverse in nature*"- Il s'agit du large éventail de types et de structures de données existant dans les datasets, qui proviennent d'une variété de sources. En ce qui concerne les types ou les formats, les données peuvent être, par exemple, des données cartographiques, des données d'imagerie, des données textuelles géolocalisées, des données structurées et non structurées, des données matricielles et vectorielles. Des centaines, voire des milliers, d'attributs différents dans de multiples dimensions, génèrent trop de variétés et énormément de combinaisons (Ceci est dû à l'absence d'une clé primaire pour relier ces ensembles de données).Les outils traditionnels de gestion des bases de données sont donc incapables de les gérer.

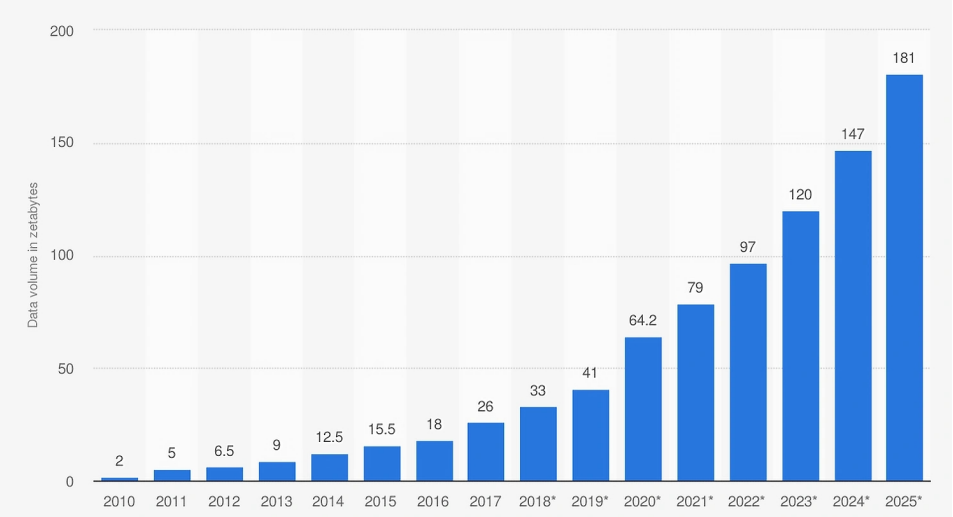


FIGURE 2.2 – Volume de données produites et consommées dans le monde entre 2010 et 2025 (en zettaoctets) [11]

Par exemple, lors de la finale de la Coupe du monde de football 2014 entre l'Allemagne et l'Argentine, 88 millions d'utilisateurs de Facebook ont envoyés collectivement 280 millions d'interactions Facebook, notamment des messages, des commentaires ou des "likes" [12]. Des milliers de commentaires ont été postés et reçus dans des milliers de villes du monde entier, et ces interactions ont été communiquées par des centaines de fournisseurs de données différents utilisant des centaines de types d'appareils différents. Par conséquent, il existe un nombre infini de combinaisons et de variations possibles dans l'analyse des big data.

— ***Vélocité :****"Big data is generated at a high rate of speed"*- Dans un contexte où les dispositifs numériques (tels que les ordinateurs, les smartphones, les terminaux de paiement ou les machines industrielles équipées de capteurs) sont couramment utilisés dans un large spectre d'activités sociales et professionnelles, les données sont générées à grande vitesse et en temps réel. Par exemple, selon les chiffres de l'étude Internet 2021 de smartinsights (voir figure 2.3) [13], chaque minute, des centaines de vidéos sont téléchargées, 125 406 vidéos sont visionnées sur YouTube, 510 miles commentaires sont publiés avec 293 000 statuts sont mis à jour et 136 000 photos sont partagées sur Facebook. Ces ensembles de données en croissance

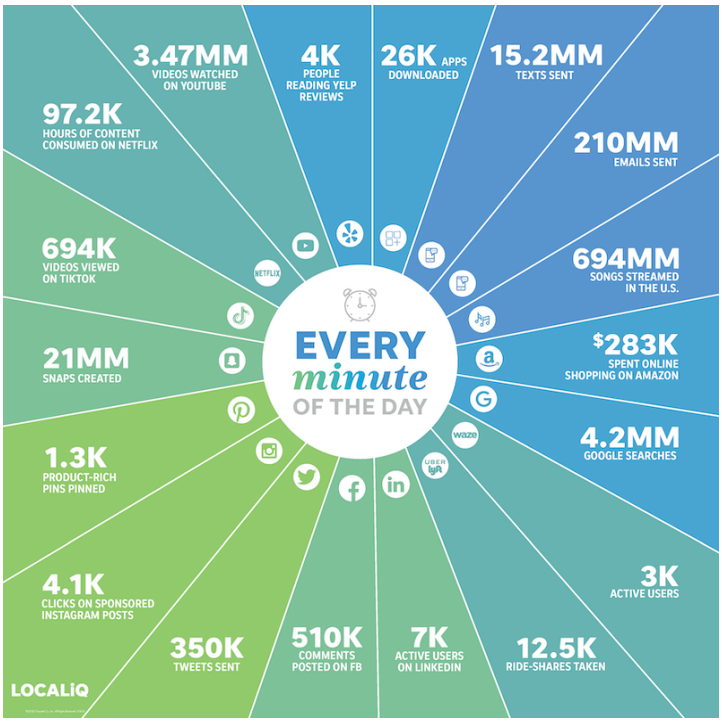


FIGURE 2.3 – Ce qui se passe en ligne en 60 secondes en 2021[13]

rapide augmentent la difficulté de stocker, traiter, analyser et visualiser les données en temps réel pour les rendre utiles au processus de prise de décision.

— ***Véracité :***"*Refers to the precision and correctness of big data.*"- Les données recueillies peuvent être désordonnés, avec du bruit et des erreurs. Les raisons en sont principalement des erreurs humaines ou de machine dans l'extraction, le stockage et/ou le traitement des données, ainsi que l'inexactitude ou l'incertitude des données elles-mêmes (par exemple, les données provenant des réseaux sociaux dans lesquels les individus expriment leurs opinions personnelles : ils peuvent mentir, exprimer quelque chose de faux ou faire des erreurs de frappe, entre autres). On peut accroître la véracité en collectant les données auprès de sources fiables, en les traitant sur des serveurs de confiance et en les stockant dans des espaces sécurisés. Par consequence, les décisions sont prises avec intégrité. [14].

2.2 Les bases de données NoSQL

Carlo Strozzi [15] a inventé l'acronyme NoSQL en 1998 pour désigner sa base de données "relationnelle" légère et open-source qui n'utilisait pas SQL.
La définition exacte de NoSQL est fréquemment débattue ;

— Certains prétendent que cela signifie "No SQL" (c'est-à-dire que le système n'utilise pas SQL mais un autre langage d'interrogation).

— D'autres élargissent la définition pour inclure d'autres technologies ou langages d'interrogation (pas seulement SQL).

— Mais, beaucoup affirment que la seule chose que toutes les bases de données NoSQL ont en commun est qu'elles sont non relationnelles, et que le nom "NoREL" serait plus approprié. On trouve cette dernière, la plus adéquate.

2.2.1 Les caractéristiques des bases de données NoSQL

Le terme "NoSQL" fait désormais référence à un groupe de systèmes de gestion de bases de données (SGBD) qui partagent certaines caractéristiques, telles que celles énumérées ci-dessous.

1. **Le théorème de CAP :** Le théorème de CAP, également connu sous le nom de théorème de Brewer, du nom de son fondateur, Eric Brewer [16], est un concept fondamental des bases de données non relationnelles. Ce théorème affirme qu'un entrepôt de données distribué ne peut pas fournir plus de deux sur trois des garanties suivantes :
   — *Cohérence (Consistency).* Même après l'achèvement d'une opération, les données de la base de données restent cohérentes. Par exemple, après la mise à jour d'un système, tous les clients visualiseront les mêmes données.
   — *Disponibilité (Availability).* La base de données doit être accessible et opérationnelle à tout moment. Il ne doit pas y avoir de temps d'arrêt (pause).
   — *Tolérance au partition (Partition Tolerance).* Si la communication entre les serveurs n'est pas stable, le réseau doit continuer à fonctionner. Les serveurs,

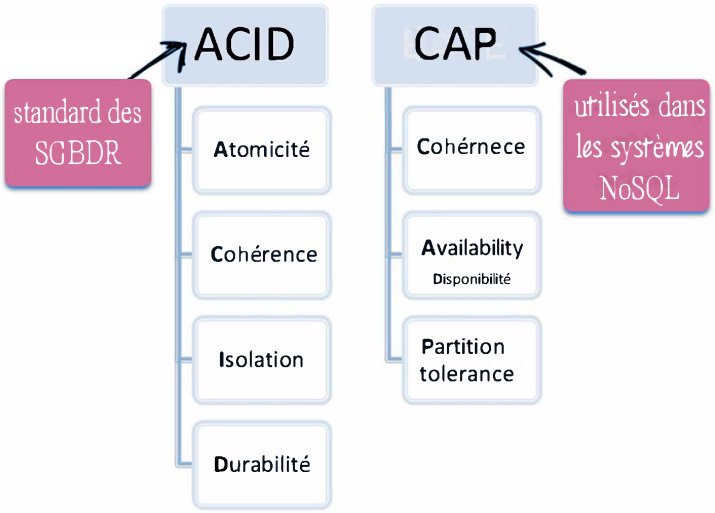


FIGURE 2.4 – ACID vs CAP

par exemple, peuvent être divisés en plusieurs groupes qui peuvent ou non communiquer entre eux. Si une partie (des serveurs hébergeant des bdd) de la base de données tombe en panne, les autres parties (serveurs) restent toujours opérationnelles.

La figure 2.4 montre le concept traditionnel d'ACID côte à côte avec le théoreme CAP.

2. **Non-Relationnel :** Par non relationnel, nous comprenons qu'il n'est pas basé sur le modèle relationnel d'E. F. Codd, proposé en 1970 [17]. La limite des bases de données relationnelles est que chaque élément ne peut avoir qu'un seul attribut. Les bases de données non relationnelles, en revanche, sont très différentes, où plusieurs éléments liés peuvent être enregistrés dans une "ligne" d'une même table à l'aide de paires clé-valeur (Key-Value Stores).

Quelques spécificités liées à l'aspect non relationnelles des bases de données NoSQL :

— Le modèle relationnel n'est jamais utilisé dans les bases de données NoSQL.

— N'utilise jamais des structures tabulaires à colonnes fixes (comme nous pouvons le voir sur la Figure 2.5).

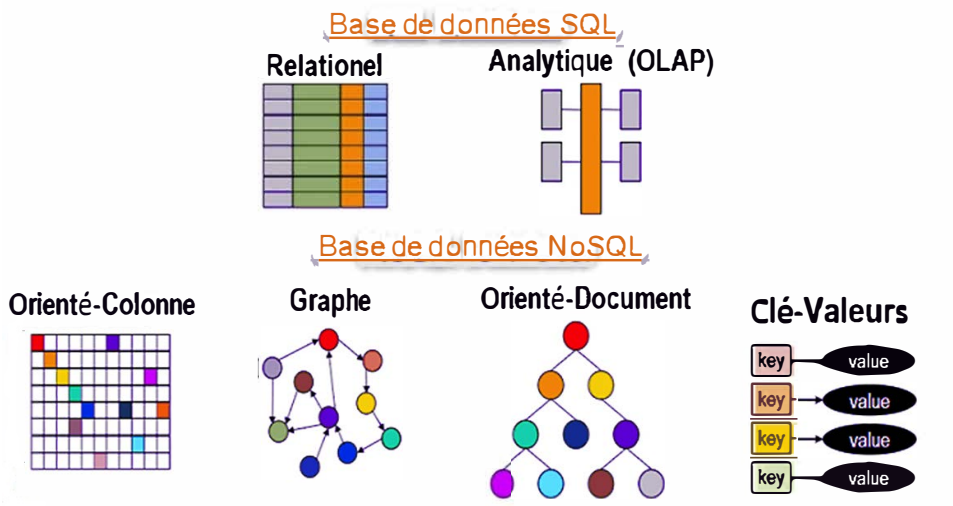


FIGURE 2.5 – Aspects non relationnelles du bases de données NoSQL

— Travaille avec des BLOBs ou des agrégats autonomes.

— La normalisation des données et le mappage objet-relationnel ne sont pas nécessaires.

— Il n'y a pas de fonctionnalités avancées telles que les langages de requête, les planificateurs de requêtes, les jointures d'intégrité référentielle ou ACID.

Les bases de données non relationnelles, ont leurs propres forces et faiblesses. À titre d'exemple, les bases de données relationnelles favorisent la recherche, tandis que les bases de données non relationnelles favorisent la précision et la redondance. Il semble raisonnable de se rappeler la philosophie du "bon outil pour le bon travail" ; c'est-à-dire que les bases de données relationnelles et non relationnelles continueront à fusionner de manière eclectique, en ajoutant des forces et en minimisant les faiblesses [18].

3. **Schema-less :** Au contraire des SGBD relationnels, la plupart des bases de données NoSQL sont sans schéma ou ont des schémas détendus, et elles ne nécessitent aucun type de définition de schéma de données (On peut le voir dans la figure 2.6). Les bases de données NoSQL fournissent également des structures de données hétérogènes au sein d'un même domaine.

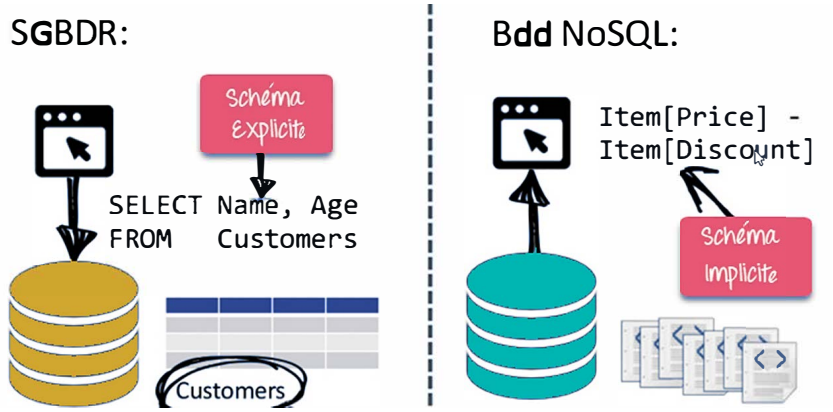


FIGURE 2.6 – Les systèmes NoSQL ont moins de schéma

4. **Évolutivité horizontale :** Bien que les bases de données relationnelles puissent être mises en place dans un cluster, elles sont plus difficiles à mettre en place que les bases de données NoSQL en raison de l'approche SGBDR. Lorsque l'on met à l'échelle une base de données relationnelle de cette manière, les performances peuvent en souffrir. La mise à l'échelle (illustré dans la figure 2.7), c'est-à-dire l'ajout des ressources à une seule machine ou la passage vers une machine plus performante, est plus facile avec les bases de données relationnelles.

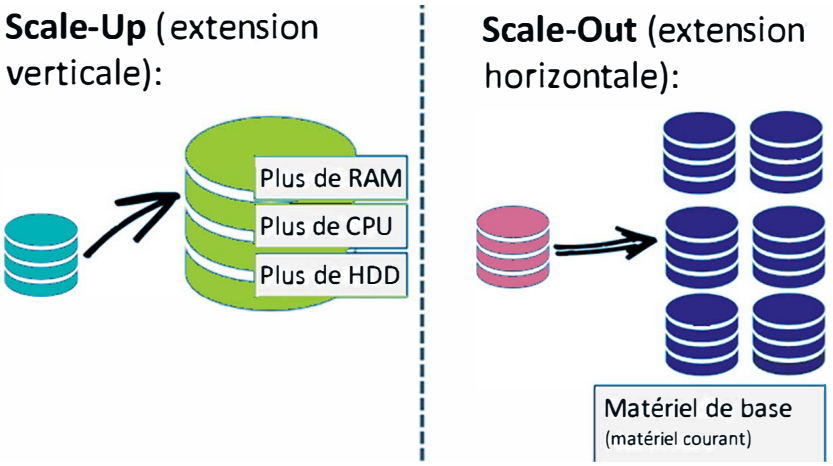


FIGURE 2.7 – L'évolutivité horizontale

Dans les environnements en cluster, la plupart des bases de données NoSQL fonctionnent bien. Les données sont réparties sur plusieurs ordinateurs afin que chacun

d'eux puisse effectuer une tâche spécifique sans affecter les autres. Chaque processeur peut accomplir sa tâche sans partager la mémoire ou l'espace disque avec les autres processeurs. C'est ce que l'on appelle une architecture de type "shared nothing", comme le montre la figure 2.8.

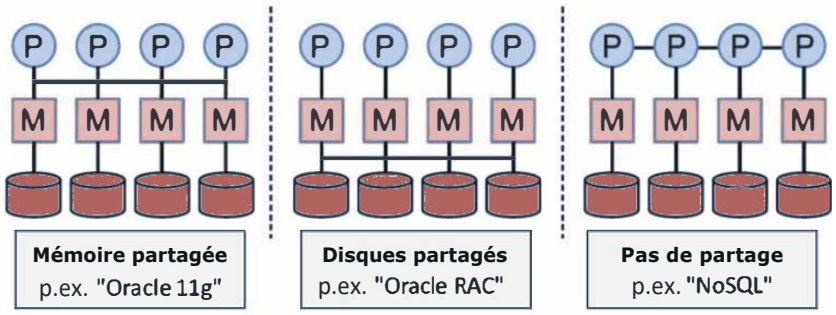


FIGURE 2.8 – L'architecture de type "shared nothing"

5. **Open source :** La majorité des SGBD NoSQL sont gratuits et open source. Bien qu'il existe de nombreux SGBD relationnels open source, le mouvement NoSQL favorise les projets open source, avec de nombreuses organisations contribuant aux efforts de développement d'une solution unique. Facebook, par exemple, a rendu le projet Cassandra open source en 2008.

2.2.2 Types de bases de données NoSQL :

Les bases de données à paires clé-valeur, orientées colonnes, basées sur des graphes et orientées documents sont les quatre principaux types de bases de données NoSQL. Chaque catégorie possède ses propres caractéristiques et ses propres limitations. Aucune des bases de données énumérées précédemment n'est meilleure pour résoudre tous les problèmes. Les utilisateurs doivent choisir une base de données qui répond aux exigences de leur produit. Certaines bases de données, cependant, sont un mélange de différents types, mais elles entrent généralement dans l'une des quatre catégories représentées sur la figure 2.9.

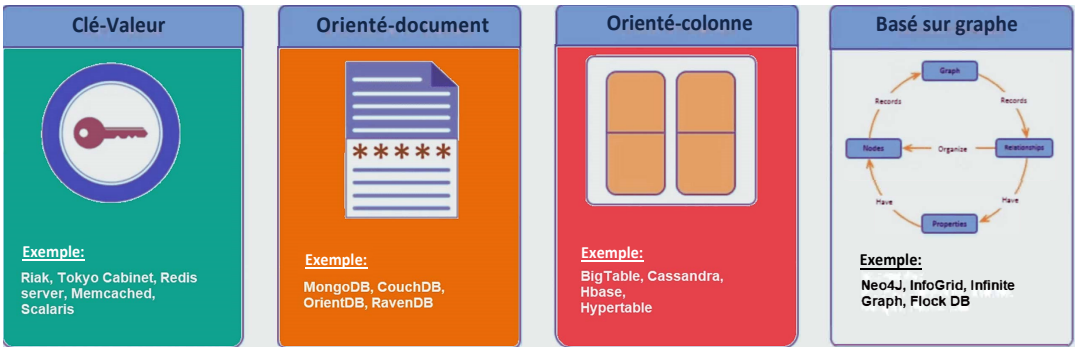


FIGURE 2.9 – Les quatre types de bases de données NoSQL

1. ***Bdd Orienté clé-valeur :***. Le terme "base de données clé-valeur" (Key-vlaue store), désigne une base de données qui stocke des données sous la forme d'un ensemble de paires clé/valeur (voir figure 2.10). L'implémentation la plus simple et basique d'une base de données NOSQL est une base de données orientée clé-valeur, dans laquelle les clés sont associées aux valeurs de manière similaire à un dictionnaire ou à un hachage.

   Puisque il n'y a ni relation ni structure dans les bases de données orientées clé-valeur. Elles stockent une paire de clés et de valeurs, qui peuvent être utilisées pour récupérer une valeur lorsqu'une clé est connue. Les bases de données clés-valeurs fonctionnent différemment des bases de données relationnelles, car ces dernières utilisent une structure de données prédéfinie sous la forme d'une série de tableaux contenant des champs avec des types de données bien définis. En revanche, les systèmes de stockage clés-valeurs traitent les données comme une collection unique et non transparente qui peut avoir des champs différents pour chaque enregistrement. L'évolutivité est l'une des principales caractéristiques des magasins de valeurs clés, car ils nécessitent peu ou pas de remaniement et peuvent être rapides dans la plupart des cas [19]. Parmi les bases de données clés-valeurs les plus populaires, citons DynamoDB, Riak et Redis.

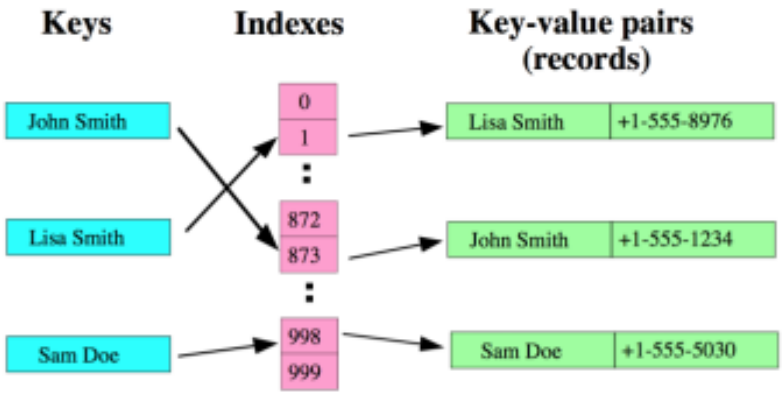


FIGURE 2.10 – Base de données orientée clé-valeur [20]

2. ***Bdd Orienté colonne :*** Une base de données "Orienté colonne" (également connue sous le nom de base de données Column, Wide column store ou Columnar) est basée sur le principe de fonctionnement du BigTable de Google. Elle travaille avec des colonnes, où Chaque colonne est traitée séparément, tandis que les valeurs unique sont conservées ensemble. Les bases de données orientées colonnes sont essentiellement un tableau bidimensionnel où chaque clé, un enregistrement/une rangée a une ou plusieurs paires clé/valeur qui lui sont attachées [21].

   En d'autres termes, un enregistrement peut se trouver dans une ou plusieurs colonnes. Les colonnes peuvent également être imbriquées dans d'autres colonnes, appelées super colonnes. Les colonnes peuvent être regroupées en une ou plusieurs familles de colonnes. L'extraction des données se fait en utilisant une clé primaire par famille de colonnes. À première vue, les bases de données orientées colonnes peuvent ressembler à une base de données relationnelle (voir figure 2.11), mais elles sont différentes car elles n'ont pas de table pré-structurée pour travailler avec les données. Cette propriété de la base de données orientée colonne rend la récupération de grandes quantités d'un attribut particulier plus rapide. HBase, Cassandra, HBase et Hypertable, sont des exemples de bases de données orienté colonnes.

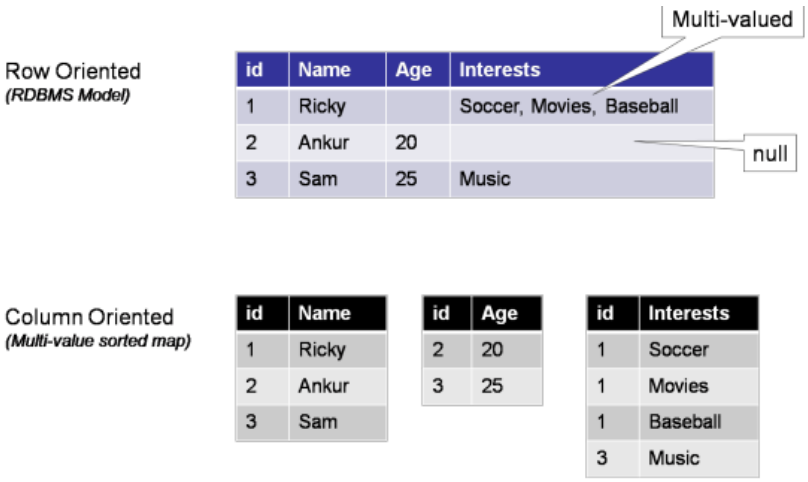


FIGURE 2.11 – comparaison d'une base de données orientée colonnes avec une base de données relationnelle [20]

3. ***Bdd Orienté graphe :*** Une base de données de type graphe, est une collection de branchements entre des noeuds. Les nœuds représentent des entités distinctes (une entreprise, une personne ou un objet). Chaque mémoire/nœud est lié au nœud suivant. Une "arête" est une connexion qui représente une relation entre deux nœuds. Les bases de données orientées graphe reposent sur une structure de graphe explicite arborescente comme illustré dans la figure 2.12. Chaque nœud connaît ses nœuds adjacents et il existe un index pour les recherches. Les nœuds stockent des données sur chaque entité de la base de données, les arêtes décrivent la relation entre les nœuds et la propriété du nœud opposé de la relation [22]. Les bases de données orientées graphe les plus populaires sont OrientDB, MarkLogic et Neo4j.

4. ***Orienté documents :*** Une base de données de type documents (également connue sous le nom de base de données orientée documents, base de données agrégée, ou simplement base de données documents), est une base de données qui stocke et récupère des données sous la forme d'une paire clé-valeur, la partie valeur étant stockée sous forme de document. Les base orientées documents enregistrent toutes les informations relatives à un élément unique sous la forme d'une seule instance (voir figure 2.13) au lieu de les répartir sur plusieurs tables, comme dans le cas des

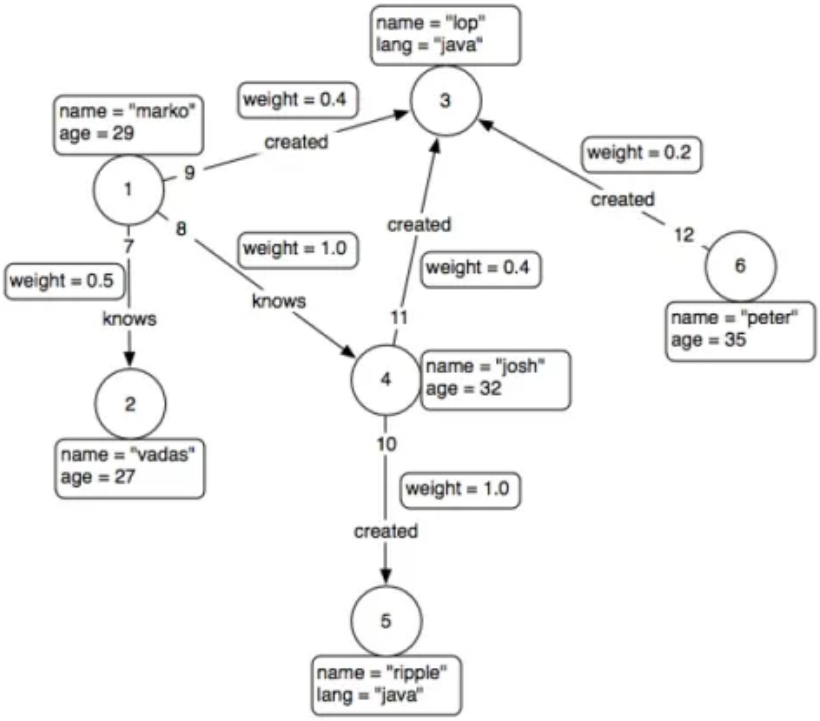


FIGURE 2.12 – Base de données orienté graphe [20]

bases de données relationnelles. Cela facilite un peu le mappage des éléments dans la base de données.

Dans ces bases de données, chaque enregistrement et ses données associées sont traités comme un document (tout ce qui est lié à un objet est encapsulé ensemble). Parmi les formats utilisés pour encapsuler et coder les données dans les documents, nous citons : JSON (JavaScript Object Notation), XML (Extensible Markup Language) et BSON (Binary JSON) [21]. Parmi les bases de données orientées documents les plus populaires, citons MongoDB, CouchDB et Couchbase.

2.3 Le Langage de Modélisation Unifié (UML) :

Omit de nos propres définitions, la plupart des concepts et des définitions présentés dans cette section sont principalement compilés à partir des références suivantes. [23, 24][25, 26]

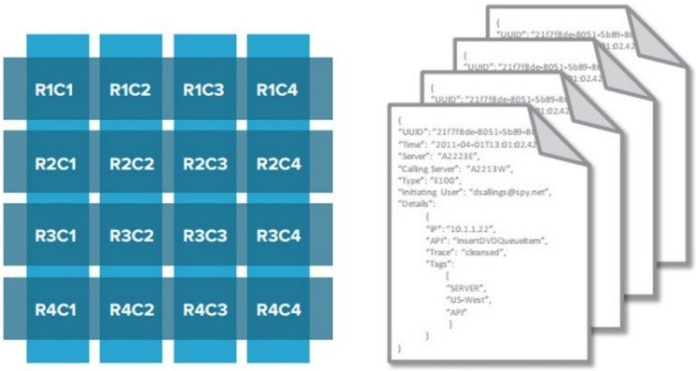


FIGURE 2.13 – comparaison d'une base de données orientée document à une base de données relationnelle [20]

Le Langage de Modélisation Unifié, en anglais Unified Modeling Language (UML), est un langage de modélisation graphique à base de pictogrammes qui est utilisé pour comprendre et décrire les besoins des utilisateurs, spécifier et documenter les systèmes, esquisser des architectures logicielles, concevoir des solutions et communiquer des points de vue.

Il a été créé dans les années 1990 comme une combinaison de plusieurs techniques, notamment la technique OOAD de Grady Booch, l'OMT de James Rumbaugh et l'OOSE d'Ivar Jacobson. Le projet UML visait à normaliser les modèles sémantiques, les notations syntaxiques et les diagrammes de OOA. La version actuelle, UML 2.5 comprend 14 types de diagrammes, sept structurels et sept comportementaux. En comparaison, UML 1.3 comptait 25 types de diagrammes différents.

2.3.1 Systèmes et modèles en UML :

- **Système :** Un ensemble d'éléments organisés pour atteindre les objectifs visés. Les systèmes sont souvent divisés en sous-systèmes et décrits par un ensemble de modèles.
- **Modèle :** Le modèle est une abstraction simplifiée, complète et cohérente d'un système, créée pour une meilleure compréhension du système.

- **Vue :** Une vue est une projection du modèle d'un système selon une perspective spécifique.

2.3.2 Modèle coceptuel d'UML :

La production d'un modèle conceptuel pour tout système donné, en suivant le langage UML, nécessite trois éléments majeurs. Les blocs de construction de base de l'UML, les règles qui dictent la façon dont ces blocs de construction peuvent être assemblés, et certains mécanismes communs qui s'appliquent à l'ensemble de l'UML.

□ **Les blocs de construction de base :**

(a) **Les Objets :** Il y a quatre types d'objets dans UML, à savoir :

- **Les objets structurels :** Ce sont les substantifs des modèles UML représentant les éléments statiques qui peuvent être physiques ou conceptuels. Les éléments structurels sont les suivants : classe, interface, collaboration, cas d'utilisation, classe active, composants et noeuds.
- **Les objets de comportements :** Ce sont les verbes des modèles UML qui représentent le comportement dynamique dans le temps et l'espace. Les deux types d'objets comportementaux sont l'interaction et l'état transitions.
- **Les objets de regroupement :** Ils constituent les parties organisationnelles des modèles UML. Il n'existe qu'un seul type de regroupement, à savoir le paquetage.
- **Les objets annotationnelles :** Ce sont les explications dans les modèles UML qui représentent les commentaires appliqués pour décrire les éléments.

(b) **Les relations :** Les relations sont le lien entre les objets. Les quatre types de relations qui peuvent être représentées en UML sont les suivants :

- **Dépendance :** Il s'agit d'une relation sémantique entre deux objets telle qu'un changement dans l'un entraîne un changement dans l'autre. Le premier est l'objet indépendant, tandis que le second est l'objet dépendant.
- **Association :** Il s'agit d'une relation structurelle qui représente un groupe de liens ayant une structure et un comportement communs.
- **Généralisation :** Cela représente une relation de généralisation/spécialisation dans laquelle les sous-classes héritent de la structure et du comportement des super-classes.
- **Réalisation :** Il s'agit d'une relation sémantique entre deux ou plusieurs classifieurs telle qu'un classifieur établit un contrat que les autres classifieurs s'engagent à respecter.

(c) **Les Diagrammes :** Un diagramme est une représentation graphique d'un système. Il comprend un groupe d'éléments généralement sous la forme d'un graphe. Les diagrammes UML les plus utilisés parmi les 14 diagrammes sont :

- **Diagramme de Classes**
- **Diagramme d'Objets**
- **Diagramme de Cas d'utilisations**
- **Diagramme de Séquence**
- **Diagramme de Collaboration**
- **Diagramme d'Etats transitions**
- **Diagramme d'Activités**
- **Diagramme de Composants**
- **Diagramme de Déploiement**

- **Les règles :** UML comporte un certain nombre de règles afin que les modèles soient sémantiquement autoconsistants et reliés aux autres modèles du système de manière harmonieuse. UML possède des règles sémantiques pour les éléments suivants :
  - **Les noms**
  - **Le domaine d'application**
  - **La visibilité**
  - **L'intégrité**
  - **L'exécution**
- **Les Mécanismes communs :** UML a quatre mécanismes communs :
  - **Les spécifications**
  - **Les Garnitures**
  - **Les divisions communes**
  - **Les mécanismes d'extensibilité**

Ce n'est pas le sujet de cette thèse de donner des explications sur tous les mécanismes et diagrammes UML. Néanmoins, nous mettrons des explications sur les deux diagrammes concernés par la méthode que nous proposons, à savoir le diagramme de cas d'utilisation et le diagramme de classe.

2.3.3 Le diagramme de Cas d'utilisations :

Un diagramme de cas d'utilisation est une représentation visuelle de la manière dont un utilisateur pourrait interagir avec un système. Un diagramme de cas d'utilisation décrit les différents cas d'utilisation du système et les différents types d'utilisateurs, et est souvent accompagné avec d'autres diagrammes. Des cercles ou des ellipses sont utilisés pour représenter les cas d'utilisation. Les acteurs sont souvent représentés sous forme de personnages en bâton. La figure 2.14, montre un exemple de diagramme de cas d'utilisation.

Un diagramme de cas d'utilisation efficace peut aider l'équipe de développement à discuter et à représenter :

— Les scénarios dans lesquels le système ou l'application interagit avec des personnes, des organisations ou des systèmes externes.

— les objectifs que le système ou l'application aide ces entités (appelées acteurs) à atteindre

— La portée du système

Composants d'un diagramme de cas d'utilisation :

Les composants et éléments communs d'un diagramme de cas d'utilisation sont les suivants :

— **Acteurs :** Les utilisateurs qui interagissent avec un système. Un acteur peut être une personne, une organisation ou un système externe qui interagit avec le système. Il doit s'agir d'objets externes qui produisent ou consomment des données.

— **Système :** Une séquence spécifique d'actions et d'interactions entre les acteurs et le système. Un système peut également être appelé un scénario.

— **Les objectifs :** Le résultat final de la plupart des cas d'utilisation. Un diagramme réussi doit décrire les activités et les variantes utilisées pour atteindre l'objectif.

2.3.4 Le diagramme de Classe :

En UML, un diagramme de classes est un diagramme de structure statique qui décrit la structure d'un système en affichant les classes, les attributs, les opérations (ou méthodes) et les relations entre les objets du système. La figure 2.15 présente un exemple d'un diagramme de classes.
Les diagrammes de classes offrent un certain nombre d'avantages pour toute organisation. Les diagrammes de classes UML sont utilisés pour :

— Illustrer les modèles de données des systèmes d'information, qu'ils soient simples ou complexes.

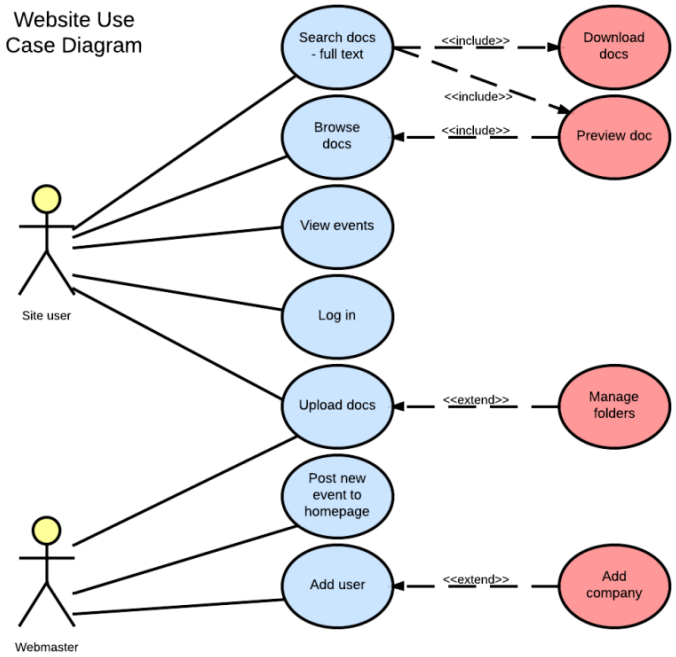


FIGURE 2.14 – Un exemple d'un diagramme de cas d'utilisation [26]

— Mieux comprendre l'aperçu général des schémas d'une application.

— Exprimer visuellement tout besoin spécifique d'un système et diffuser cette information dans l'entreprise.

— Fournir une description indépendante de l'implémentation des types utilisés dans un système qui sont ensuite transmis entre ses composants.

Composants de base d'un diagramme de classes : Le diagramme de classes standard est composé de trois sections :

— **Section supérieure :** Contient le nom de la classe. Cette section est toujours requise, que l'on parle du classificateur ou d'un objet.

— **Section centrale :** Contient les attributs de la classe. Cette section est utilisée pour décrire les qualités de la classe, et elle n'est requise que pour décrire une instance spécifique d'une classe.

— **Section inférieure :** Comprend les opérations (méthodes) de la classe. Présentée sous forme de liste, chaque opération occupe sa propre ligne. Les opérations décrivent comment une classe interagit avec les données.

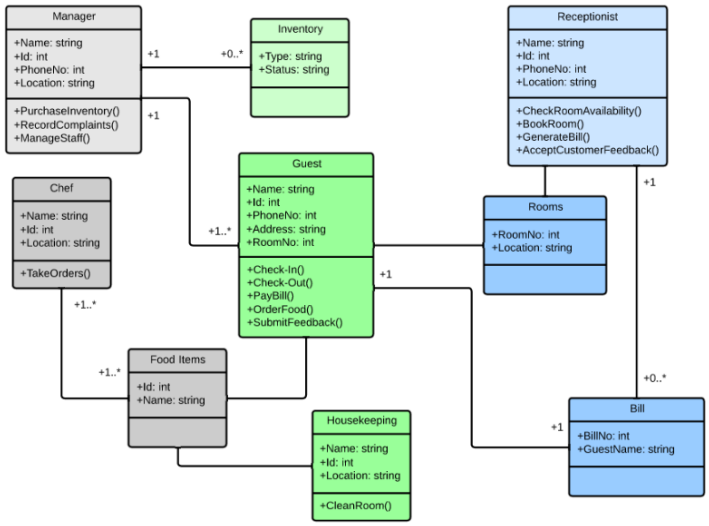


FIGURE 2.15 – Un exemple d'un diagramme de classes [26]

2.4 Conclusion :

Dans ce chapitre, nous avons abordé les concepts clés liés à la modélisation des bases de données dans le monde du big data dans un cadre théorique. Nous avons abordé certaines bases théoriques fondamentales pour les systèmes de big data et les bases de données NoSQL. En outre, nous avons abordé quelques concepts UML et la manière dont ils peuvent être utilisés pour créer un modèle conceptuel d'un système NoSQL.

Chapitre 3

À la recherche d'une méthodologie de conception pour les bases de données NoSQL

"Torture the data, and it will confess to anything" (Ronald Coase)

Ce chapitre présente le processus de modélisation des bases de données NoSQL à l'ère du big data. Tout d'abord, nous recensons les publications disponibles dans la littérature qui ont étudié la question de la conception des bases de données NoSQL, du point de vue de la méta-modélisation. Ensuite dans le même contexte, nous proposons comme premième contribution une revue systématique de la littérature sur la modélisation des données dans les bases de données NoSQL basée sur UML.

3.1 Enquêtes et études pertinentes existantes :

La littérature sur la modélisation des big data montre une variété d'approches consacrées au développement de méthodologies et d'outils supportant la conception de bases de données NoSQL [27, 28][29]. Cependant, les publications qui ont étudié cette question du point de vue de la méta-modélisation sont très rares. En plus, la plupart de ces travaux proposent une solution de transformation des bases de données traditionnelles vers un modèle NoSQL spécifique pour un cas d'étude précis.

L'étude de synthèse	Année	Contexte
[30]	2011	Les bases de données NoSQL
[31]	2016	la modélisation et la gestion des bases de données NoSQL
[32]	2018	Les bases de données NoSQL
[33]	2018	Méthodologie de conception des bases de données NoSQL
[34]	2020	Méthodes de conception des bases de données à l'ère du big data
[35]	2020	Modélisation et gestion des big data dans les bases de données.
[36]	2021	Modélisation des données et des bases de données NoSQL

TABLE 3.1 – Enquêtes et revues sur la modélisation des données dans les bases de données NoSQL

Le tableau 3.1, présente des travaux universitaires qui ont été produits sous forme d'enquêtes ou d'évaluations d'experts, mais aucun d'entre eux n'a fait l'objet d'une analyse de la littérature systématique sur des approches ou des méthodes spécifiques. Bien que la modélisation des données soit mentionnée dans [30], ce concept (modélisation des données) fait référence à la structure, à l'organisation et à la méthode de stockage des données dans les systèmes NoSQL, et non pas à la manière de modéliser les données à l'aide de niveaux de représentation et de modèles spécifiques. De même, en utilisant des techniques telles que l'ontologie, la sémantique, les schémas RDF et le langage SPARQL, Gil et Song [31] ont introduit le terme "modélisation conceptuelle" pour le big data. Les points forts et les points faibles des diverses approches de conception de benchmark ont été discutés dans cet article.

Deux travaux académiques ont été publiés en 2018 [32, 33]. Les deux donnent un aperçu de quatre bases de modèles non orthogonales pour les systèmes de bases de données distribuées : le partitionnement des données, les modèles de données, le théorème CAP et le modèle de cohérence. Encore une fois, le modèle de données dans ces travaux académiques est simplement une compréhension de la façon dont les données sont organi-

sées, structurées et stockées. Ce n'est qu'en 2020, que les deux travaux [34, 35] ont mené une enquête spécifiquement axée sur le modèle de données des bases de données NoSQL, ont abouti à des découvertes importantes malgré l'absence d'une analyse systématique.

Enfin, dans [36] les auteurs se sont concentrés sur les méthodologies utilisées par diverses études. Ils ont répertorié les travaux connexes visant à unifier la plupart des bases de données NoSQL sous une méthodologie de conception uniforme avant de proposer une classification méthodologique.

3.2 Méthodes de conception spécifiques aux bdd NoSQL basées sur UML :

Dans cette section, nous définissons les questions de recherche et le protocole de l'analyse, pour réaliser une synthèse systématique.
Les directives proposées par Kitchenham [3] ont été utilisées pour réaliser cette revue systématique de la littérature (RSL), qui est une méthode permettant d'identifier, d'évaluer et d'interpréter toutes les recherches possibles concernant un sujet spécifique. Notre étude RSL réalisée en trois phases, est décrite dans la figure 3.1.

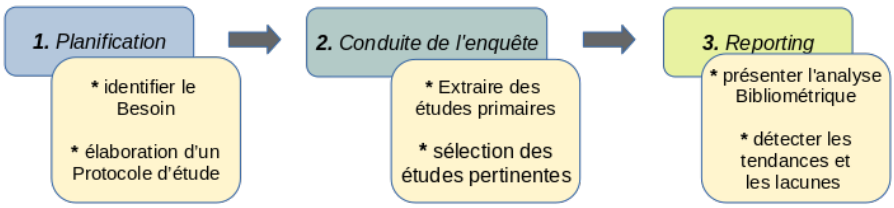


FIGURE 3.1 – Les phases d'une analyse systématique de la littérature

Les activités de la première phase consistaient à vérifier l'existence d'autres études RSL sur le sujet d'intérêt, à définir le protocole de révision à utiliser afin d'atténuer les biais potentiels des chercheurs impliqués, à élaborer les objectifs, les justifications et les questions de recherche et enfin, à définir la stratégie à suivre.

Les activités de la deuxième phase consistaient à collecter les études primaires à partir des sources définies dans l'étape de planification. Puis, à appliquer les critères d'inclusion pour ne retenir que les études relatives à la modélisation des big data basée sur UML.

Dans la troisième phase, les activités correspondantes sont liées à une analyse complète pour fournir des informations sur les modèles employés à chaque niveaux de représentation utilisés pour la modélisation des bases de données NoSQL.
Chaque phase est expliquée plus en détail dans les sections qui suivent.

3.3 La planification de l'étude RSL :

Les principaux objectifs de cette phase consistent à déterminer la nécessité d'une étude de RSL et d'élaborer un protocole de révision.

a) **Le besoin d'une étude de RSL :** Nous avons déjà montré dans le chapitre précédent, pourquoi il est important de faire une phase de modélisation avant de développer des applications basées sur les big data.
Néanmoins, il n'existe pas de méthodologies de modélisation des données largement adoptés par le secteur économique, et admises par la communauté scientifique pour les systèmes NoSQL. Malgré l'hétérogénéité, le nombre inconnu de niveaux de représentation et le manque de modèles utilisés à chaque niveau, de nouvelles propositions de modélisation des données apparaissent pour les bases de données NoSQL. Notre intérêt est de mener des recherches dans ce domaine et de fournir un RSL sur les travaux qui utilisent une norme ou un standard pour modéliser les bases de données NoSQL.

b) **Le protocole du RSL :** Avant de réaliser le RSL, un protocole de recherche doit être défini afin de prévenir tout conflit d'intérêt potentiel entre les chercheurs [3]. En conséquence, notre protocole est formé comme suit : D'abord, nous avons proposé des objectifs de développement spécifiques ainsi que des arguments pour ex-

pliquer nos contributions. Ensuite, afin de résumer les preuves existantes sur la modélisation du big data à l'aide d'UML, nous avons formulé cinq questions de recherche.

- ***Objectifs et justifications :*** Dans une analyse complète de la littérature, le premier objectif est de présenter des informations sur les recherches les plus pertinentes en matière de modélisation des bases de données NoSQL. Sur la base de ces résultats, le deuxième objectif était d'identifier les diverses approches de modélisation du big data utilisées dans les différentes études afin de déterminer les tendances et les lacunes liées aux trois concepts clés du big data, le modèle source, le modèle cible et les 4V. L'étude RSL menée dans le cadre de cette recherche a permis de rassembler toutes les recherches sur la modélisation des bases de données NoSQL basées sur UML en un seul document, ce qui profitera à l'industrie et au monde académique.
- ***Les questions de recherche :*** L'objectif de cette recherche de littérature, est de découvrir comment les différents modèles appliqués dans les bases de données traditionnelles (en particulier UML), sont appliqués aux bases de données NoSQL. Pour cette proposition, cinq questions de recherche (QR) ont été définies :

  QR1. *Quels types de bases de données NoSQL sont abordés par les chercheurs ?*

  QR2. *Quels niveaux de représentation sont présentés dans les méthodologies de modélisation des bases de données NoSQL ?*

  QR3. *Quelles considérations sont prises en compte pour les 4V avec ces méthodes ?*

  QR4. *Comment ces méthodes ont-elles été évaluées ?*

  QR5. *Quelles sont les tendances et les lacunes en matière de modélisation et de gestion du big data ?*

3.4 La Conduite de l'enquête :

Les étapes suivantes ont été nécessaires afin de compiler une liste exhaustive d'études sur le sujet d'intérêt : Trouver des études primaires, sélectionner des études potentiellement pertinentes, appliquer des critères d'inclusion aux études primaires, extraire et synthétiser les données.

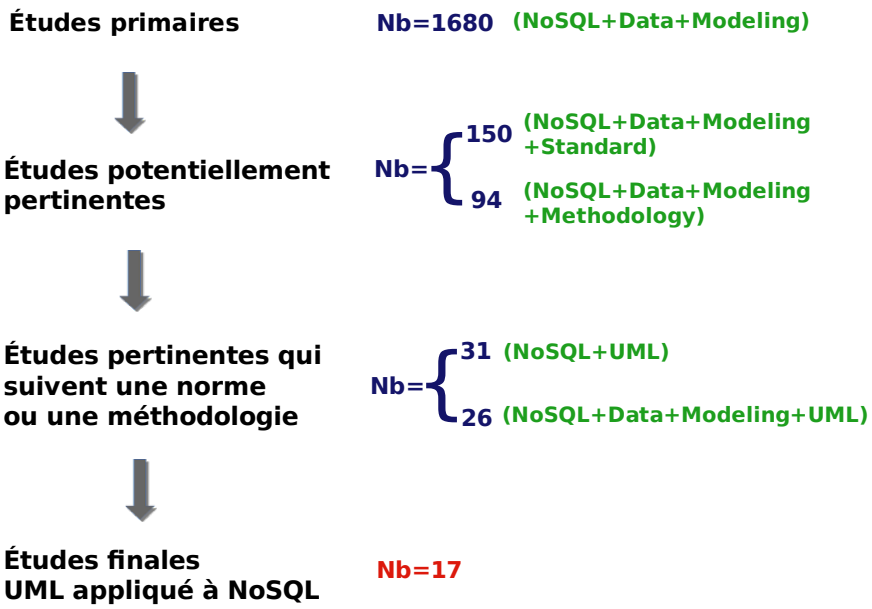


FIGURE 3.2 – La procédure de sélection

La figure 3.2 donne une représentation du processus de sélection appliqué aux études. Les études en double ont été écartées de l'étape des études potentiellement pertinentes.

3.4.1 Identification des études primaires :

Pour identifier les candidats aux études primaires, nous avons d'abord appliqué la chaîne de recherche générique (NoSQL+Data+Modeling) sur les deux moteurs de recherches "Google Scholar" et "Semantic Scholar". Nous avons ainsi obtenu 1680 documents au total à partir de ces deux sources. En outre, une recherche manuelle (sondage cumulatif) a été effectuée dans "Connected papers" dans le but de trouver plus de travaux de recherches liés à la modélisation des données pour les bases de données NoSQL.

3.4.2 La sélection des potentiellement pertinentes :

Afin de sélectionner les études liées à la modélisation et à la conceptualisation du big data, nous avons effectué une recherche par un processus de correspondance de termes spécifiques dans les titres, les résumés et les mots-clés des articles. Sur la base de nos questions de recherche, trois principaux termes de recherche ont été dérivés : NoSQL, UML et Modélisation. Ces termes ont été sélectionnés après avoir combiné plusieurs options, afin d'obtenir un nombre significatif d'articles et ces termes couvraient la majorité des études qui abordaient notre sujet de recherche.

3.4.3 Application des critères d'inclusion :

A ce stade, un examen rapide du contenu complet de chaque étude nous a permis de sélectionner uniquement les études relatives à la modélisation du big data basée sur UML. Cela a abouti à l'acceptation de 17 études, au rejet de 25 articles et au filtrage de 5 travaux dupliqués. A cette phase, nous avons éliminé les travaux dupliqués, et les 17 études restantes ont constitué notre corpus final pour rapporter l'étude SLR.

3.4.4 Extraction des données et synthèse :

L'objectif de cette étape est de faire une analyse plus approfondie des études primaires afin de répondre aux questions de recherche soulevées dans le protocole de review. Pour ce faire, nous lisons les travaux académiques sélectionnés en examinant les attributs communs comme le titre, le résumé, les mots-clés et, si nécessaire, leurs introductions et leurs conclusions. En plus de ces attributs, nous avons défini les niveaux de représentation des attributs, le type de base de données NoSQL, le contexte de la modélisation des données, et la prise en charge des caractéristiques 4V's du big data.

3.5 Résultats de l'analyse :

Maintenant, nous répondons aux questions de recherche par le biais des activités suivantes :

1. Une analyse bibliométrique, pour recueillir des informations sur les auteurs et les données de publication, l'impact des études sélectionnées (H-index et sjr) et l'année de publication, ainsi que les revues et conférences où les études ont été publiées ;
2. Une revue de la littérature pour classer les études en fonction de trois concepts clés : les entrées, les sorties et les caractéristiques du big data, dans une table de concepts. Dans les entrées de la source, nous analysons le modèle source, les blocs UML et les types de données. Dans les sorties cibles, nous analysons les niveaux d'abstraction des données, les modèles de données proposés aux niveaux conceptuel, logique et physique au système NoSQL visé. Au niveau des caractéristiques du big data, nous analysons la prise en charge de chacune des 4V's.
3. Une discussion pour identifier les tendances et les lacunes dans le domaine de la modélisation et de la gestion des big data.

3.5.1 L'analyse bibliométrique :

L'objectif de cette analyse est de consolider, par le biais d'une analyse bibliométrique, tous les efforts de recherche sur le sujet, en fournissant aux chercheurs la possibilité de connaître toutes les informations sur les auteurs et les données de publication dans un seul document. Ainsi, le lecteur peut savoir comment les études, dans notre sujet d'intérêt, se sont développées au fil du temps, quels sont les travaux qui ont apporté des contributions significatives au sujet et quelles sont les études les plus citées. Dans le tableau 3.3, la pertinence des revues est illustrée par les métriques H-Index et SJR. Dans ce tableau, nous résumons les résultats de l'étape d'inclusion et mettons en évidence les conclusions suivantes :

Papier	Année	Journal / Proceeding	Sjr 2020	H-Index
[37]	2016	International Conference on Conceptual Modeling, LNCS subseries Lecture Notes in Bioinformatics	0.249	400
[38]	2018	International Conference on Model and Data Engineering, LNCS subseries LNBI	0.249	400
[29]	2017	IEEE Transactions on Knowledge and Data Engineering	1.36	174
[27]	2020	International Conference on Computer Systems and Technologies	0.56	63
[39]	2014	Frontiers in Artificial Intelligence and Applications	0.16	54
[40]	2018	International International Conference on Business Process Modeling, Development and Support	0.21	49
[41]	2017	International Journal of Applied Engineering Research	-	40
[42]	2013	IEEE International Conference on big data	-	26
[43]	2021	International Journal of Data Warehousing and Mining	0.16	23
[44]	2016	International Journal of Web Information Systems	0.19	18
[45]	2018	International Journal of Advanced Computer Science and Applications	0.19	18
[46]	2020	International Conference on Database and Expert Systems Applications - DEXA	0.17	15
[47]	2015	IEEE International Conference on Smart City	-	10
[48]	2014	IEEE Workshop on Advanced Research and Technology in Industry Applications	-	10
[49]	2015	International Conference on Information Integration and Web-Based Applications	-	8
[50]	2018	Proceedings ofthe 14th International Conference on Web Information Systems and Technologies	0.12	3
[51]	2018	International Conference on BigData Innovations and Applications, Innovate-Data	0.12	2

TABLE 3.3 – Journaux et conférences où les études pertinentes ont été présentées

— La croissance du nombre d'études sur la modélisation des bases de données NoSQL basée sur UML, est constante au fil des années.

— Avant 2013, aucune étude pertinente n'a été trouvée.

— 2018 a été l'année au cours de laquelle nous avons découvert le plus grand nombre d'études sur la modélisation du big data.

Le tableau 3.2 montre les revues les plus pertinentes. Les 17 travaux universitaires sélectionnés ont été produits par 54 auteurs, issus de 22 institutions de recherche publiques et privées, situées dans 14 pays. Ces chiffres montrent à quel point la recherche sur la modélisation des données pour les bases de données NoSQL est diffuse.

3.5.2 Revue systématique de la littérature :

L'objectif de cette section est de répondre aux trois questions de recherche QR1, QR2 et QR3, vues dans la section 3.3.

QR1. *Quels types de bases de données NoSQL sont abordés par les chercheurs ?*

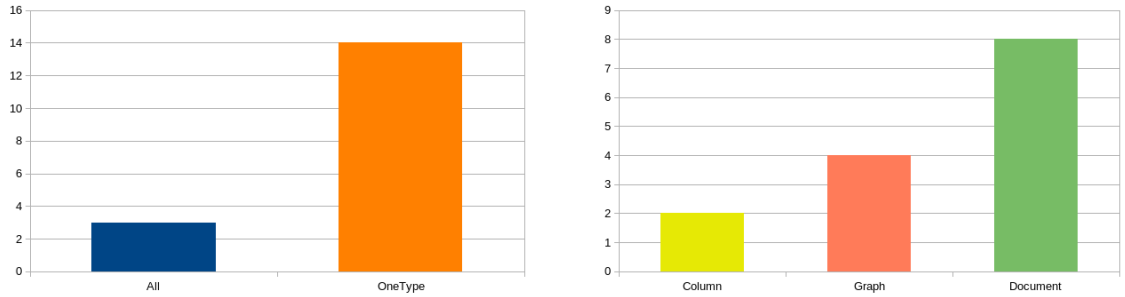
Certaines méthodes peuvent s'adresser à des types de bases de données spécifiques, voire à des fournisseurs de bases de données spécifiques. Ces méthodes sont moins inclusives et plus limitées, tandis que d'autres ont tendance à être plus générales. Pour répondre à cette question, nous avons vérifié la compatibilité des études avec les différentes bases de données. Notre analyse a mis en évidence deux catégories différentes. La première catégorie est constituée d'études qui n'abordent aucune base de données spécifique (catégorisées comme "All" - non adaptées à une base de données spécifique), c'est-à-dire des méthodes qui considèrent tous les types de bases de données NoSQL. La deuxième catégorie contient les études qui traitent un type spécifique de base de données NoSQL, que nous avons catégorisé comme "OneType"- adapté à une base de données spécifique.

La figure 3a montre la répartition des trois catégories, tandis que la figure 3b, se concentre sur les différents types. Trois des 17 méthodes identifiées (18%) abor-

Travaux	Nombre d'auteurs	Pays
[37]	3	France, Espagne
[38]	5	Espagne
[29]	3	Canada, Qatar
[27]	4	Italie
[39]	2	Allemagne
[40]	4	Israël
[41]	3	Corée du Sud
[42]	2	Inde
[43]	3	France
[44]	2	Brésil
[45]	6	Malaisie, Nigeria, Espagne
[46]	3	France, Colombie
[47]	4	Chine
[48]	3	Chine
[49]	2	Brésil
[50]	3	Espagne
[51]	2	Malaisie

TABLE 3.2 – Répartition des études primaires par nombre d'auteurs et par pays

daient le vaste problème de la conception de bases de données NoSQL suivant une norme. Bien qu'elles visent à concevoir une méthode applicable à plusieurs types de bases de données, ces méthodes peuvent être plus compliquées et nécessitent un processus d'apprentissage plus long. Quatre études ont porté sur les bases de données de type graphe ; deux études ont porté sur les bases de données de type colonne. En examinant de plus près les différents types, aucune étude n'a abordé les bases de données à clé-valeur. Cela peut être dû à leur structure simple[52]. Les bases de données orienté-document ont attiré le plus d'attention, avec huit études.



(a) Spécifique à la base de données (b) Type de NoSQL

FIGURE 3.3 – (a) nombre de travaux par traitement de base de données ; (b) nombre de travaux par type de NoSQL

QR2. *Quels niveaux de représentation sont présentés dans les méthodologies de modélisation des bases de données NoSQL ?*

L'objectif de cette question est d'identifier quels et combien de niveaux de représentation (conceptuel, logique et physique) sont appliqués dans le processus de modélisation des données dans les bases de données NoSQL. Le niveau de représentation utilisé pour modéliser une base de données NoSQL influencera les propriétés ou les spécifications à prendre en compte dans le processus de modélisation.

Nous avons répondu à cette question de recherche pour chaque type de base de données NoSQL.

La figure 3.4 montre les niveaux de représentation de chaque catégorie. Dans le cas des bases de données NoSQL orientées documents, les travaux se concentrent sur la modélisation dans le niveau logique, tandis que dans les travaux à deux niveaux de représentation, ils se concentrent sur les niveaux logique-physique de représentation. Trois travaux avec trois niveaux de représentation ont été trouvés. Pour les bases de données NoSQL en colonnes, la littérature se concentre sur les niveaux de représentation conceptuel-physique et conceptuel-logique. Aucun travail avec trois niveaux de représentation n'a été trouvé. Enfin, il est à noter que les travaux qui se concentrent sur la modélisation des bases de données de type graphe utilisent un seul niveau de représentation ; dans ce cas, le niveau conceptuel de représentation.

Le deuxième niveau qui a été utilisé le plus fréquemment dans cette catégorie est le niveau logique.

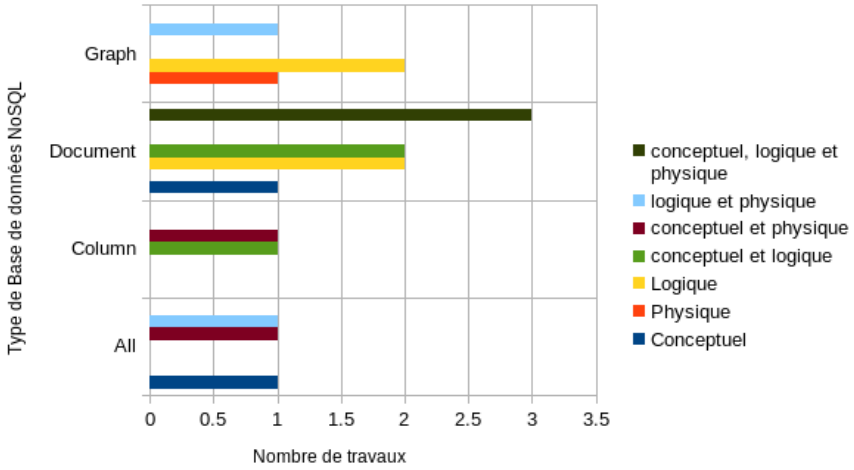


FIGURE 3.4 – Niveaux de représentation par type de bases de données NoSQL.

QR3. *Quelles considérations sont prises en compte pour les 4V avec ces méthodes ?*

Pour répondre à cette question, nous devons d'abord définir ce que sont les concepts d'exigences non-fonctionnelles.

Les exigences non-fonctionnelles font référence à la capacité d'exprimer des méta-propriétés comme l'évolutivité et la cohérence, qui peuvent avoir un impact sur la base de données et sa conception. Nous nous attendons également à ce que les méthodes de conception aient la capacité de prendre en charge les 4V du big data. Les critères des exigences non-fonctionnelles sont moins abordés que ceux des exigences fonctionnelles, avec seulement six études abordant les exigences non-fonctionnelles dans une certaine mesure. Le tableau 3.4 montre le niveau de support des différents critères des 4V par les méthodes étudiées. Le concept le plus abordé est le volume, avec plus de 3 études l'utilisant dans leur méthode de conception, par exemple, [43], abordait le volume et les autres V du big data. Certaines méthodes ont utilisé des exigences non fonctionnelles qui n'étaient pas incluses dans nos critères ; Par exemple, dans la méthode [47], le concept de "performance" (représenté par le processus de conversion basé sur un workload) est utilisé dans le

Articles	Année	Entrées (eg : UML blocks)	Sorties (NoSQL systems)	Caractéristiques du big data
[42]	2013	Orienté modèle - ER	Orienté documents et graphes	————
[39]	2014	Notation UML	Bdd Graphe	Valeur
[48]	2014	Diagramme de class de l'UML	modèle indépendant de la plate-forme	————-
[49]	2015	ER étendue, Notation UML	Orienté documents	Cohérence
[47]	2015	Diagramme de class de l'UML	Orienté documents	Variabilité
[37]	2016	Méta-modèle du diagramme de classe UML	Orienté colonne	—
[44]	2016	ER étendue	Orienté documents	Volume et Valeur
[29]	2017	Schéma conceptuel et statistiques	Column-store	Volume et Vélocité
[41]	2017	Modèle conceptuel de données UML - Peter Chen	Orienté document	—
[50]	2018	Modèle entité-association	Orienté colonne	————-
[51]	2018	Modèle ER	Orienté documents	————-
[40]	2018	Diagramme de class UML	Bdd de type Graphe	————-
[38]	2018	Modèle de données générique	Bdd de type document et colonne	————-
[45]	2018	Notation UML	Orienté documents	Cohérence
[46]	2020	Diagramme de class UML et Contraintes OCL	Base de données de graphes	—
[27]	2020	Vue de données orientée vers l'agrégat	Modèle de données indépendant du système	Évolutivité et cohérence
[43]	2021	Diagramme de class de UML	Bdd de type colonnes, documents et graphes	Volume, variété et vélocité

TABLE 3.4 – Une étude comparative des approches de modélisation NoSQL basée sur l'UML

cadre de la méthode de conception proposée ; et dans l'approche de [50], le concept d'"intégrité" (c'est-à-dire l'intégrité logique des données liée à la cohérence d'une ligne répliquée dans toutes les répliques du cluster Cassandra.) et le concept de performance ont été utilisés.

3.5.3 Discussion :

Dans cette section, nous répondons aux QR4 et QR5.

QR4. *Comment ces méthodes ont-elles été évaluées ?*

Nous avons été confrontés à ce problème lors de l'élaboration de notre méthode. Nous avons donc décidé de nous pencher sur les types d'évaluation adoptés par les différentes études.

Pour répondre à cette question, nous avons identifié plusieurs types d'évaluation : *(1) Expérimentation :* certaines méthodes offraient plusieurs options de conception. Des expériences ont été réalisées afin de comparer les différentes stratégies proposées par la méthode elle-même ; *(2) Exemple illustratif :* certains travaux présentent une étude de cas pour évaluer leur approche avec un exemple illustratif dans un domaine spécifique ; *(3) Comparaison :* certaines études ont tenté de comparer une méthode suggérée avec une méthode valide et ayant fait l'objet de nombreuses citations. Certains auteurs ont procédé à une évaluation hybride. Par exemple, dans [48], ils ont utilisé une application réelle utilisant Apache Cassandra pour stocker certaines données afin d'évaluer l'approche proposée, ainsi qu'une comparaison entre le schéma de données généré et le schéma de données réel de cette application.

La figure 3.5 montre la répartition des types d'évaluation. Comme on peut le voir dans cette figure, la plupart des méthodes n'ont pas été évaluées. Certaines études comprenaient des plans pour l'évaluation de la méthode en question à l'avenir ; d'autres présentaient un simple cas d'utilisation au lieu d'une évaluation complète.

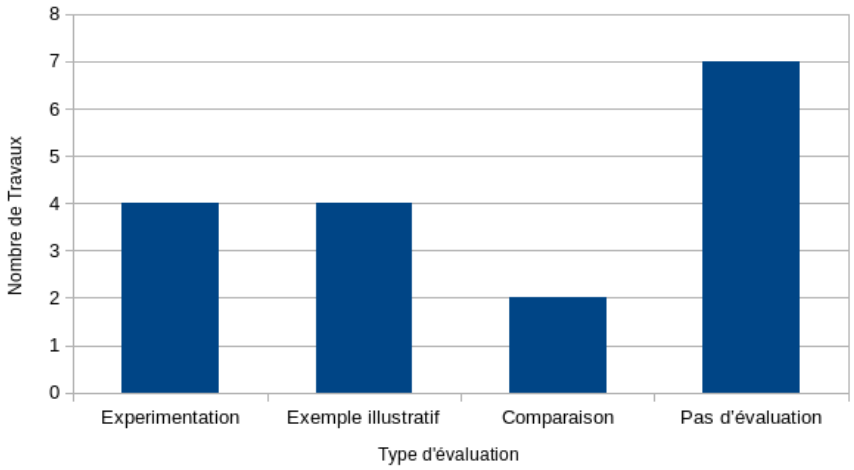


FIGURE 3.5 – Nombre d'études par type d'évaluation

QR5. *Quelles sont les tendances et les lacunes en matière de modélisation et de gestion du big data ?*

Pour répondre à la QR5, nous avons examiné les suggestions des articles concernant les futures lignes de recherche. Cinq des études n'ont rien dit sur les perspectives. Nous avons divisé les futures orientations de recherche proposées en trois catégories sur la base des 12 études restantes. La première est l'évaluation, qui comprend les travaux qui n'ont pas évalué leur méthode mais qui ont l'intention de le faire à l'avenir. La deuxième catégorie concerne l'implémentation automatique ou semi-automatique de la méthode proposée. La troisième catégorie considère l'expansion de la méthode suggérée de deux façons : l'expansion verticale considère l'expansion de la méthode pour inclure plus de fonctionnalités (telles que le support des requêtes, l'indexation, etc.) et l'expansion horizontale concerne l'expansion de la méthode pour inclure plus de types de bases de données. Certains articles ont discuté des futurs plans de recherche dans plusieurs de ces catégories.

Comme résultats supplémentaires, les informations suivantes ont été identifiées dans les articles examinés :

1. La majorité des études présentent leurs résultats à trois niveaux d'abstraction : conceptuel, logique et physique.
2. Au niveau conceptuel des données, le modèle le plus utilisé est le diagramme de classes, suivi par le modèle ER.
3. Le modèle orienté document est le plus étudié au niveau logique, suivi du modèle orienté graphe et du modèle orienté colonne.
4. La majorité des méthodes se concentrent sur un modèle de données NoSQL spécifique, chacune offrant sa propre approche de modélisation conceptuelle. Les concepteurs qui travaillent avec différents modèles de données NoSQL et qui doivent passer d'une méthode de conception à l'autre devront fournir un effort supplémentaire.
5. De nombreuses méthodes décrivent comment construire un schéma de base de données NoSQL en utilisant un ensemble d'heuristiques et de directives qui doivent être appliquées manuellement. Cela peut prendre beaucoup de temps et être source d'erreurs.

3.6 Conclusion

Dans ce chapitre, nous avons soulevé cinq questions de recherche auxquelles nous avons répondu par une revue systématique de la littérature. La contribution de cette RSL, en clarifiant les connaissances sur le sujet spécifique de la modélisation du big data basée sur UML, peut aider les chercheurs et les praticiens à améliorer leurs projets big data. Les résultats de cette étude révèlent des informations très intéressantes. Par exemple, plus de 70% des études ne vérifient pas leurs propositions avec des jeux de données du monde réel. La caractéristique de "Vélocité" du big data n'est pas justifiée par la plupart des études. Le modèle **ER** est le plus utilisé au niveau conceptuelle, et le modèle orienté document est le plus étudié au niveau logique. En outre, nous avons identifié comme principales lacunes le manque d'évaluations méthodes proposées.

Nous savons que, en raison de la nécessité d'analyser de grands volumes de données avec une variété de structures, qui arrivent à une fréquence élevée, la recherche sur les bases de données s'est orientée vers NoSQL. Cependant, les SGBD NoSQL n'ont pas été en mesure d'acquérir certains points forts déjà présents dans les bases de données relationnelles, tels que la capacité à supporter simultanément la cohérence et la disponibilité.

Nous espérons maintenir le RSL à jour et présenter des résultats pour d'autres concepts. De plus, compte tenu des lacunes et du besoin constant de nouvelles méthodes de modélisation des bases de données NoSQL basées sur un standard, nous présenterons dans le prochain chapitre notre nouvelle approche de la modélisation des bases de données NoSQL basée sur UML.

Chapitre 4

UML4NoSQL : Une nouvelle approche pour la modélisation des bases de données NoSQL Orienté-documents basée sur UML.

Vous reconnaissez peut-être l'expression "les données ne sont pas des informations" de *Clifford Stoll*, vue au début de ce document. Dans le monde de la BI et de l'analyse des données, les mots de *Stoll* signifient que les données seules ne sont pas particulièrement utiles. Ce n'est que lorsque nous y ajoutons du contexte qu'elles peuvent être traduites en informations exploitables. Comment passe-t-on des données brutes à la connaissance ? La réponse se trouve dans la modélisation et la visualisation des données, qui vont au-delà des tableaux de bord et des diagrammes à barres.

Dans ce chapitre, nous présentons notre principale contribution en deux parties. Tout d'abord, nous décrivons comment les diagrammes UML peuvent être utilisés pour modéliser les bases de données NoSQL en utilisant l'approche proposée connue sous le nom de "UML4NoSQL". Ensuite, nous présentons des lignes directrices pour réussir la transition d'un diagramme de classe (CDM) vers un modèle orienté document (LDOM) NoSQL. Le processus de conversion est ensuite appliqué au schéma orienté document dans la phase expérimentale. Pour formaliser notre approche, nous définissons quelques termes utilisés dans la construction du modèle orienté document.

4.1 La modélisation des données dans le contexte du big data :

Un modèle de données spécifie comment les entités du monde réel et leurs relations sont représentées et exploitées [53]. En conséquence, les systèmes NoSQL utilisent différents modèles de données à des fins différentes, y compris (entre autres) les modèles à colonnes étendues et les modèles de documents. Malgré leurs différences, ces modèles de données sont orientés vers les agrégats. Plus précisément, ils organisent les données sous forme d'unités de paires clé-valeur liées de manière imbriquée. Une telle unité de données imbriquée représente des données auxquelles on accède (lecture/écriture) en une seule opération [54].

Pour exploiter les avantages de l'imbrication ou de la dénormalisation des données lors de la conception des schémas des bases de données NoSQL, les concepteurs doivent non seulement tenir compte des données qui seront stockées dans la base de données, mais aussi de la manière dont ces données seront accessibles [55].

Cependant, les systèmes NoSQL manquaient initialement de méthodes bien définies pour la conception de schémas de bases de données. D'autre part, malgré l'existence de méthodes bien établies de conception de schémas de bases de données traditionnelles , elles ne conviennent pas aux bases de données NoSQL. Ceci est principalement dû au fait qu'elles visent à atteindre d'autres objectifs tels que les propriétés ACID. Pour combler cette lacune, de nombreuses méthodes ont été proposées pour la conception de schémas de bases de données NoSQL au cours des dernières années. Néanmoins, ces méthodes souffrent encore de certains inconvénients, qui sont récapitulés dans la section 3.5.3.

Plus spécifiquement, la disponibilité des données n'est pas un problème ; vu que la dénormalisation des données augmente leur redondance, les opérations d'écriture peuvent être entravées par les surcharges liées au maintien de la cohérence. Il est également possible que les coûts de stockage augmentent. La question la plus importante est de

savoir quoi faire avec les données ou ce qui peut être fait avec elles. Dans notre approche, nous visons à fournir une bonne représentation des données de l'application dans une base de données NoSQL (Document Oriented) cible, ainsi qu'à supporter les caractéristiques 4V des systèmes big data. Comme on peut le voir sur la figure 4.1, notre méthodologie s'articule autour de quatre axes majeurs :

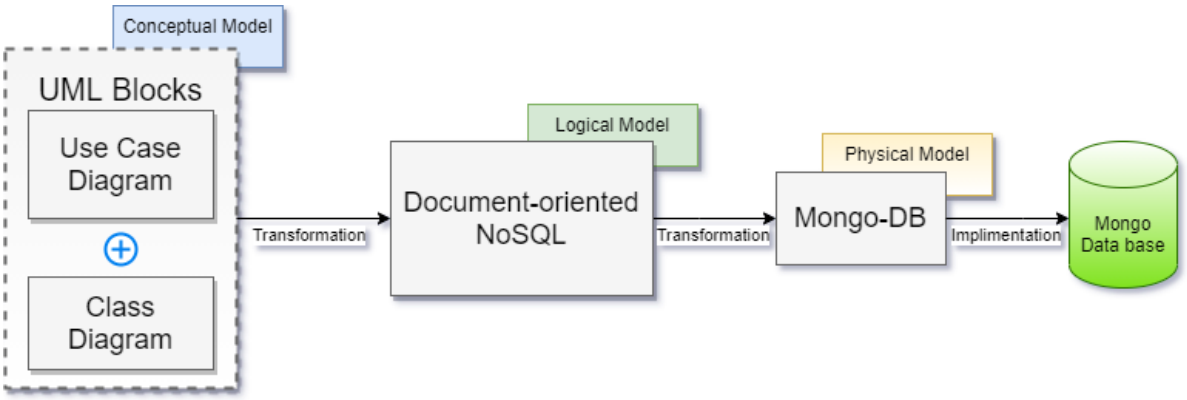


FIGURE 4.1 – Aperçu de la méthodologie proposée

4.1.1 Comprendre les données à la disposition de l'utilisateur :

Le premier principe d'une conception de base de données réussie est d'avoir une bonne connaissance des données pour lesquelles la conception est prévue en considérant les critères les plus importants : (i) le type de données à analyser, (ii) le volume de données à disposition et (iii) la rapidité avec laquelle nous avons besoin de ces données. Les auteurs dans [56] et dans [57] ont largement étudié les défis rencontrés lors de la manipulation des big data. Ces défis incluent les questions liées à la qualité des données, aux flux de données, à l'évolution dynamique des données, à l'hétérogénéité et à la modélisation des données, aux bases de données multi-modèles, aux interfaces client et requêtes, à la compression des données, au cryptage des données, au contrôle d'accès et à l'autorisation. Dans cette perspective, nous constatons que les utilisateurs finaux du big data deviennent de plus en plus non techniciens. Cela nous conduit à l'introduction d'un nouveau profil que nous appelons *Citizen developer* (un développeur citoyen).

4.1.2 Stratégies du développeur et besoins de l'utilisateur :

Alors que les diagrammes UML traditionnels restent centrés sur les besoins du système en termes de fonctions, avec l'arrivée du big data, les diagrammes devraient se concentrer sur les besoins de l'utilisateur (généralement les décideurs) en termes de traitements basés sur les données qu'ils ont en charge (analyse prescriptive, prédictive, diagnostique et descriptive). Il est important de savoir ce que l'on peut faire avec les données dont on dispose à court, moyen et long terme. Les réponses à cette question peuvent être modélisées par un diagramme de cas d'utilisation. Nous appliquons les diagrammes de cas d'utilisation pour visualiser le comportement d'un système, d'un sous-système ou d'une classe. Le diagramme de cas d'utilisation est un outil fondamental pour identifier les exigences [58]. Cela permettra aux utilisateurs de comprendre comment utiliser ces éléments, et aux développeurs comment les mettre en œuvre. Dans ce contexte, nous définissons deux nouveaux blocs comme suit :

Définition 1 : *Super Container* En français, *Super Conteneur* ; héberge toutes les qualités big data requises pour un diagramme de cas d'utilisation qui réduit la complexité du diagramme en évitant la création de multiples instances d'un service de données comme le montre la figure 4.2.

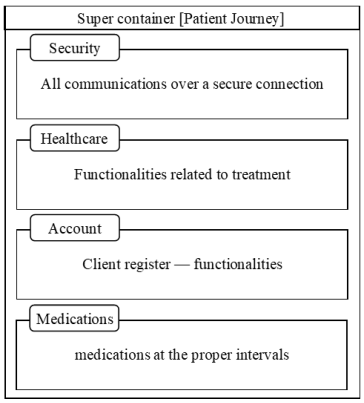


FIGURE 4.2 – Super Container

Définition 2 : *Actors' Container* En français, *Conteneur d'acteurs* ; Contient le groupe d'acteurs qui ont un comportement identique lorsqu'ils interrogent le système. Cela permet de contrôler la variabilité et la vélocité des requêtes des acteurs à l'intérieur du conteneur, comme le montre la figure 4.3.

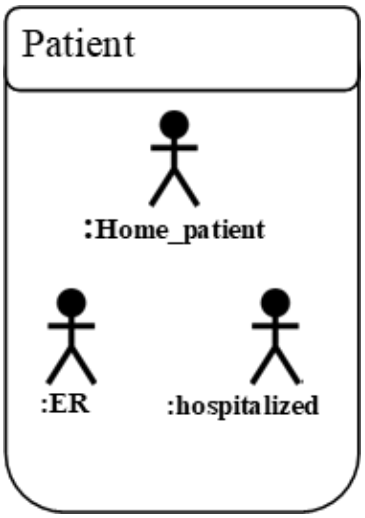


FIGURE 4.3 – Actors' Container

À partir du diagramme de cas d'utilisation, nous pouvons décider quelles données à conserver dans notre environnement big data. Soit nous conservons toutes les données, soit nous ne conservons que les données importantes et rejetons les données qui deviendront obsolètes à long terme, ce que nous appelons les données sales. Les données conservées seront modélisées par un diagramme de classe étendu. En parallèle, nous pouvons prévoir des diagrammes de séquence pour décrire les scénarios d'utilisation du big data, et des diagrammes d'activité pour plus de détails sur les algorithmes utilisés (ex. : algorithmes de data mining).

4.1.3 Le Méta-modèle NoSQL :

Plusieurs études ont indiqué que les systèmes qui supportent les big data sont sans schéma. Mais en réalité, nous pouvons dire qu'ils ont moins de schémas que les autres données. L'un des principaux objectifs de cette thèse est d'attirer l'attention sur l'importance des méta-modèles pour les systèmes big data, car lorsque nous accédons aux

données, nous avons besoin d'un schéma qui nous aide à les interpréter, donc s'il n'y a pas de modèle explicite à respecter, nous devons en déduire un. En fait, les modèles ne sont pas des artefacts statiques, fixes et complets, mais plutôt une vue partielle, dynamique et temporelle des données pour faciliter leur manipulation à un instant précis. Dans le développement traditionnel de logiciels, les développeurs suivent une approche descendante [59], dans laquelle le modèle définit les données à utiliser. Par analogie, nous voulons proposer une approche qui s'appuie sur une technique data-up, c'est-à-dire que le modèle utilisé est basé sur les données disponibles.

4.1.4 Créez un design qui évolue facilement :

Une des principales clés d'une bonne conception de la base de données, est la compréhension des requêtes réelles, les requêtes ont un effet important sur la conception du schéma. En plus d'assurer le support correct des requêtes, nous devrions prendre en compte le chemin d'accès de chaque requête pour organiser les données efficacement (dénormalisation ou utilisation d'un schéma relationnel). Les travaux de [47] et [28], présente en détail la normalisation et la dénormalisation dans les bases de données.

Plutôt que de préserver un schéma relationnel, il est préférable de dénormaliser les données afin de pouvoir tirer profit des documents imbriqués et répétés. En effet, les documents imbriqués et répétés peuvent maintenir des relations sans affecter les performances du schéma relationnel (normalisé) préservé. Nous convenons avec les auteurs [60, 61][32], que la dénormalisation des données améliore les performances des bases de données NoSQL.

4.2 Du modèle conceptuel de données au modèle orienté document :

Les composants du modèle conceptuel de données UML peuvent être transformés en composants du modèle de données des documents, comme le montre le tableau 4.1. La classe peut être transformée en collection (ensemble de documents). L'attribut peut être converti en colonne. L'association peut être mappée en référence ou incorporée par les modèles de requête de l'application.

	NoSQL Systems			
UML Class Diagram	**Key_Value**	**Column Store**	**Document Store**	**Graph Database**
Class	Associative array	Column	Collections	Graph
Attribute in Class	Key-value	name/value in column	Documents	Nodes
Association	Key-value pairs	Directory hierarchies	References / Embedded	Edges

TABLE 4.1 – Correspondance entre les composants du diagramme de classe UML et les modèles de données NoSQL

Deffinition3 : Soit D un ensemble de documents $D_j, j = 1...m$. Chaque document est défini par un ensemble de couples atomiques ou sous-documents (attribut, valeur) :

$$\mathcal{D}_j \quad = \quad \{(Att, Val)_i; i = 1...n\}, j = 1...m$$

où n est le nombre d'attributs, et m le nombre de documents. Les couples (attribut, valeur) des documents imbriqués sont triés dans une collection :

$$\mathcal{C}oll \quad = \quad \{(Att, Val)_i^j; j = 1...m; i = 1...n\}$$

Si $\mathcal{C}^A$ est un sous-ensemble de $\mathcal{C}oll$, représentant une classe $\mathcal{A}$, nous supposons que :

1. $\exists D_i, D_j \in D$ de telle sorte que $D_i \subset \mathcal{C}^A$ et $D_j \subset \mathcal{C}^B$
2. $\mathcal{C}^{AB} = \{C^A\{C^B\}\}$ représentent le fait que C^B et intégré dans C^A.

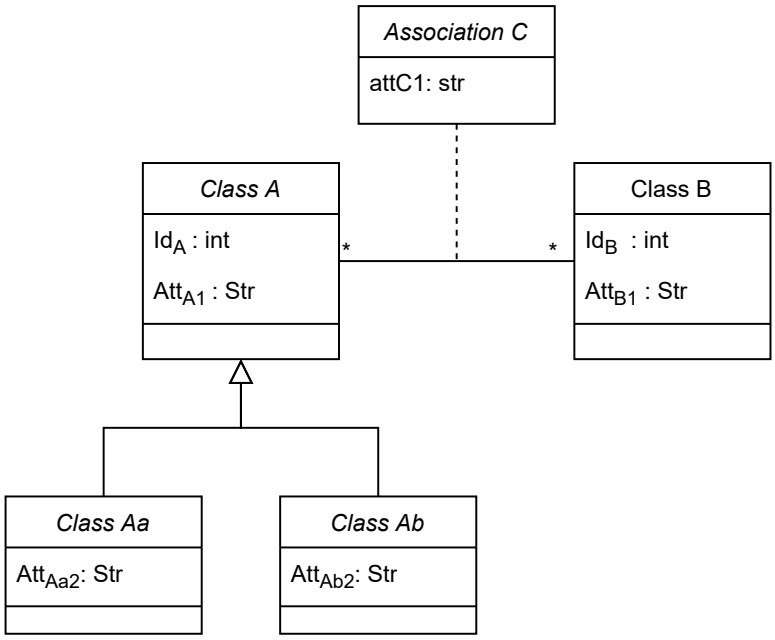


FIGURE 4.4 – Passage d'un MCD to MLOD

Le processus de passage d'un MCD à un MLOD, pour le cas illustré dans la figure 4.4, passe par les étapes suivantes :

— Chaque classe est transformée en une collection composée d'un ensemble de documents, où les attributs de la classe sont convertis en attributs dans chaque document :

$$\begin{aligned} C^{classA} &\subset \mathcal{C}oll \\ C^{classA} &= \{(Att, Val)^{A}_{id}, (Att, Val)^{A}_{Att_{A1}}\} \end{aligned}$$

— Une association basique de type (1..*) entre les deux collections, C^{classA} et C^{classB}, où la clé primaire de C^{classB} migre vers la collection C^{classA} comme clé étrangère, génère une collection imbriquée N^{classB} qui est C^{classB} :

$$C^{AB} = \{C^{ClassA}\{C^{ClassB}\}\} = \left\{ \begin{array}{l} (Att, Val)^{A}_{id}, (Att, Val)^{A}_{Att_{A1}}, (Att, Val)^{B}_{id}, \\ \\ N^{classB} : \{(Att, Val)^{B}_{id}, (Att, Val)^{B}_{Att_{B1}}\} \end{array} \right\}$$

— Une association plusieurs à plusieurs (*..*) entre deux collections, C^{classA} et C^{classB}, donne naissance à une nouvelle collection $Coll^{AB}$ contenant les attributs de la classe d'association, plus les clés primaires des deux classes participantes, plus les deux collections imbriquées N^{classA} et N^{classB} avec leurs propres attributs.

$$Coll^{AB} = \left\{ \begin{array}{c} (Att, Val)^{A}_{id}, (Att, Val)^{B}_{id}, (Att, Val)_{AttC1}, \\ N^{classA} : \{(Att, Val)^{A}_{id}, (Att, Val)^{A}_{Att_{A1}}, \\ N^{classB} : \{(Att, Val)^{B}_{id}, (Att, Val)^{B}_{Att_{B1}} \end{array} \right\}$$

— Le modèle orienté document, sera constitué des ensembles parents-enfants (association d'héritage). Dans chaque collection enfant imbriquée $N^{classAa}$ et $N^{classAb}$, on trouve les attributs non clés de la classe plus la clé primaire composée des clés de la classe parent-enfant. La valeur "Sort" indique quelle classe enfant doit être utilisée pour compléter les informations dans la collection générée C^{AaAb}.

$$C^{AaAb} = \left\{ \begin{array}{c} (Att, Val)^{A}_{id}, (Att, Val)^{A}_{A1}, (Sort, \\ N^{classAa} : \{(Att, Val)^{Aa}_{id}, (Att, Val)^{A}_{id}, (Att, Val)^{Aa}_{Att_{Aa2}}\} \\ N^{classAb} : \{(Att, Val)^{Ab}_{id}, (Att, Val)^{A}_{id}, (Att, Val)^{Ab}_{Att_{Ab2}}\}) \end{array} \right\}$$

4.3 Conclusion

Dans ce chapitre, nous avons présenté une nouvelle approche de conception de schémas pour les bases de données NoSQL orientées documents. Cette méthode est réalisée en quatre étapes : 1) en commençant par une compréhension des données à disposition, 2) la définition des stratégies de développement et des besoins des utilisateurs, 3) création du diagramme de cas d'utilisation et du diagramme de classe et 4) le passage des modèles de données conceptuels améliorés au modèle logique orienté document.
Afin de vérifier l'efficacité de notre proposition, le chapitre suivant illustre par le biais d'une étude de cas dans le domaine de la santé, le processus de modélisation des données en suivant les étapes de notre méthode.

Chapitre 5

Exemple illustratif

"Hiding within those mounds of data is knowledge that could change the life of a patient, or change the world"

(Atul Butte)

L'évaluation est une tâche difficile dans le cas de la modélisation des bases de données NoSQL. Dans leur article [34], Roy-hubara et al. ont abordé ce problème en présentant des techniques d'évaluation telles que l'évaluation à l'aide d'exemples illustratifs. Dans ce chapitre, nous utilisons un exemple illustratif pour démontrer l'applicabilité et la pertinence de notre approche.

5.1 Le cas d'étude :

Pour illustrer et motiver notre travail, nous avons choisi une étude de cas dans le domaine de la santé, où une base de données NoSQL a un rôle légitime à jouer. L'étude de cas porte sur un système de 51 hôpitaux comptant plus de 100 000 soignants qui fournissent des services de soins à des millions de patients chaque année. Les données des patients sont réparties dans de nombreux systèmes, notamment les dossiers médicaux électroniques (DME).

L'objectif principal de ce système de santé est (1) de collecter des données sur l'évolution de la maladie au fil du temps à partir des dossiers des patients, afin d'améliorer les soins aux patients, le coût des traitements et l'expérience des soins médicaux, (2) de proposer des tableaux de bord affichant des données détaillées sur la qualité et les coûts. Les tableaux de bord permettent aux praticiens et aux cliniciens de voir les analyses liées à chaque hôpital, à chaque clinicien et à chaque unité de soins, (3) d'analyser et de partager les données de manière plus organisée, ce qui permet aux médecins de comprendre plus facilement quels comportements ont un impact positif ou négatif sur les soins aux patients, ce qui entraîne des améliorations substantielles des mesures de qualité et des réductions importantes du coût des soins.
En outre, le système de santé peut intégrer les données des capteurs des systèmes de surveillance des patients pour améliorer la prévisibilité des alertes, il peut également utiliser les données météorologiques et saisonnières pour prévoir les besoins en personnel et en lits.

Cet exemple peut être considéré comme un problème de big data, Compte tenu de la définition des "4V" qui sont le volume, la variété, la vélocité et la véracité ; **Volume :** La grande quantité de données recueillies durant plusieurs années auprès des 51 hôpitaux peut facilement atteindre des dizaines de téraoctets. **Variété :** Chaque patient possède son propre dossier médical, qui provient de diverses sources et se présente donc sous différentes formes (structurées, non structurées, semi-structurées) et différents formats (résultats d'analyses de laboratoire, blocs de texte, rapports médicaux, photos, vidéos, images médicales, listes de médicaments, etc.) **Vélocité :** La large diffusion de l'IoT et des dispositifs médicaux intelligents conduit à la génération de données de santé en temps réel. Il existe de nombreux outils permettant d'analyser les données des patients et de conseiller aux professionnels de la santé de prendre les mesures appropriées. Les données des capteurs, par exemple, peuvent être extrêmement utiles pour surveiller la santé des patients, qu'il s'agisse du contrôle de la pression artérielle ou d'autres conditions recueillies directement à leur domicile. En revanche, essayer de répondre à chaque flux

de données sur la santé peut entraîner une allocation inefficace des ressources. Il est donc essentiel de déterminer quelles données nécessitent une action immédiate et lesquelles peuvent attendre. **Véracité :** La qualité des données comprend des données intégrées, fiables, complètes, sans biais et sans bruit. Les hôpitaux cherchent à réduire le nombre de visites aux urgences, ce qui augmente les coûts des soins de santé et qui ne permet pas nécessairement d'obtenir de meilleurs satisfactions chez les patients.

5.1.1 Description du cas expérimental :

Avant de procéder à la modélisation du cas d'utilisation, nous créons un modèle logique de données UML indépendant de tout modèle de base de données. Ce modèle est ensuite comparé avec le modèle résultant de notre approche.
L'utilisation du big data dans le domaine de la santé a déjà donné des résultats significatifs. Pour valider notre méthode, nous avons choisi une étude de cas dans ce domaine. La figure 5.1 montre un fragment de MLD pour le système analytique de santé décrit ci-dessus.

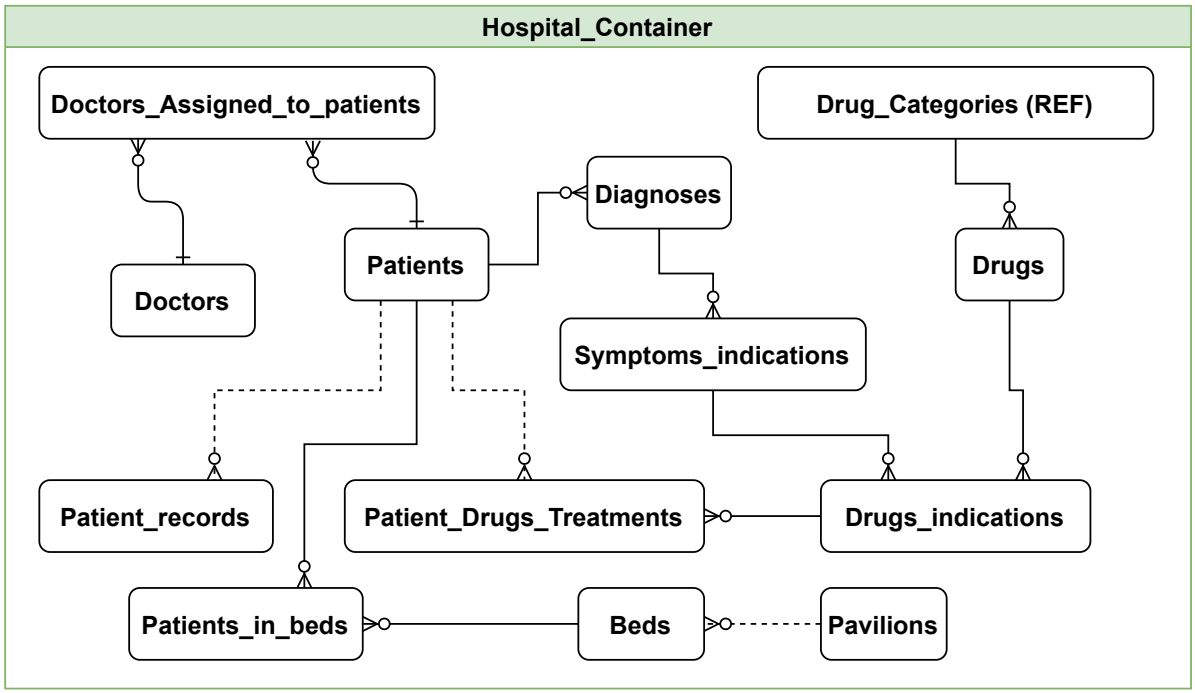


FIGURE 5.1 – Modèle logique de données pour un système de santé

5.2 Application de la méthode UML4NoSQL :

Dans cette section, nous appliquons notre approche sur l'exemple courant ci-dessus en décrivant les deux étapes suivantes : (i) Modélisation des stratégies des développeurs et des besoins des utilisateurs, (ii) La génération du modèle de données NoSQL.

5.2.1 Modélisation des stratégies des développeurs et des besoins des utilisateurs :

Afin d'établir le contexte du système, nous commençons par identifier les acteurs qui l'entourent et les intéractions entre eux. L'application du premier et du deuxième aspect de la modélisation des données UML4NoSQL conduit aux diagrammes représentés dans les figures suivantes 5.2, 5.3 et 5.4.

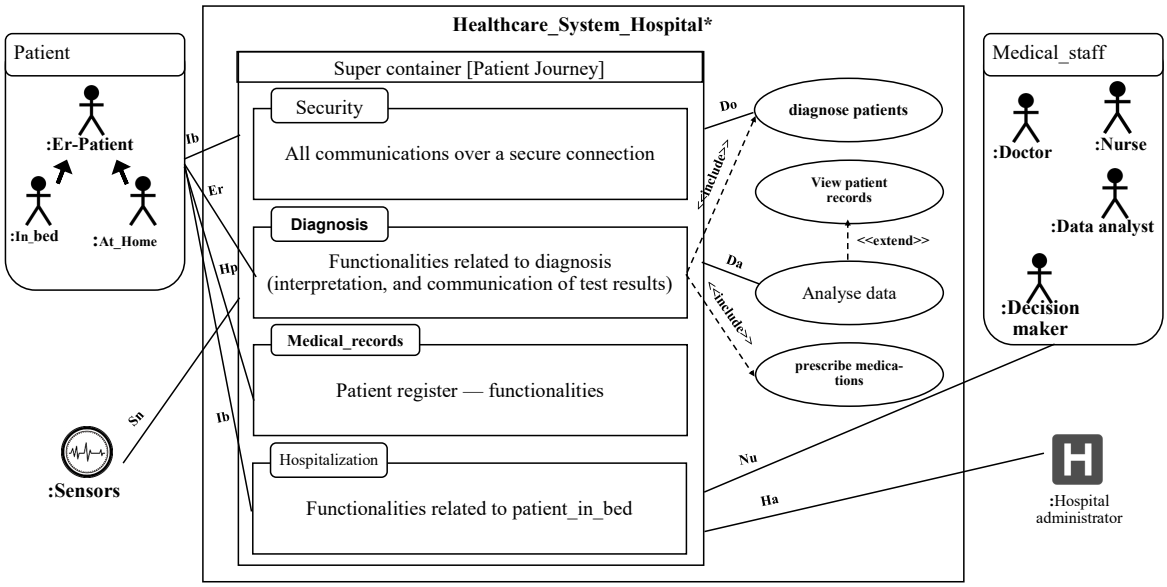


FIGURE 5.2 – Diagrammes de cas d'utilisation avancés pour le système de santé

Le diagramme de cas d'utilisation optimisé proposé introduit un *super conteneur* comme on peut le voir sur la figure 5.2. Le super conteneur héberge toutes les qualités big data requises pour un diagramme de cas d'utilisation et réduit la complexité

du diagramme en évitant la création de multiples instances d'un service de données. Par exemple, le super conteneur *Patient journey* permettra de surveiller le processus de diagnostique d'un patient accompli par un médecin.

Nous proposons également l'utilisation d'un conteneur pour un groupe d'acteurs qui ont le même comportement lorsqu'ils interrogent le système, de manière à contrôler la variabilité et la vélocité des requêtes de chaque acteur.
Nous décrivons désormais comment sont représentées les lignes de communication entre le super conteneur du parcours du patient et les conteneurs des différents acteurs afin de les connecter à d'autres sous-cas d'utilisation. Le scénario que nous décrivons ici est le suivant : Pour comprendre les changements de comportement des patients en matière de santé, l'analyste de données (acteur) analyse un échantillon de big data ciblé. Tous les contrôles au sein du super conteneur sont appliqués, car le super conteneur du parcours du patient se trouve sur la ligne de communication entre l'analyste de données et le sous-cas d'utilisation. Comme il y a plus d'un acteur, les lignes de communication doivent être étiquetées. En se référant au même scénario mentionné ci-dessus, la connexion entre l'analyste de données et le conteneur est étiquetée avec "**Da**".

Après avoir développé le cadre général du système par le biais d'un diagramme de cas d'utilisation, nous en arrivons à identifier deux étapes majeures à comprendre : Les données à gérer, et comment une application orientée données doit accéder à ces données. La première étant capturée via un modèle conceptuel de données présenté dans la figure 5.3.

Ce modèle de données est enrichi d'informations relatives aux 4V. Par exemple, le développeur doit anticiper le volume de données (Vol) en supposant des informations sur le nombre d'occurrences produites par les relations inter-entités, ainsi que sa vitesse de croissance (Vel). Selon le détail de la représentation des types de données, la variété (Var) est liée aux attributs. La véracité (Ver) est représentée en fonction de la confiance du développeur envers les sources de données, à deux niveaux : (i) au niveau de l'attribut,

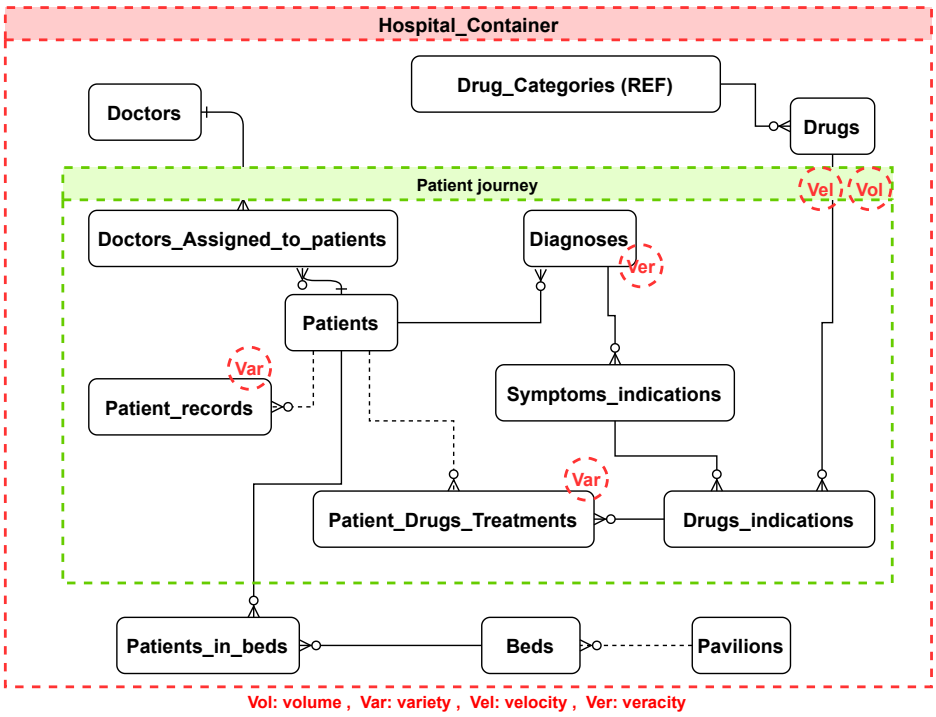


FIGURE 5.3 – Un modèle de données conceptuel enrichi pour le cas d'utilisation de santé

lorsqu'il s'agit de caractériser la confiance dans la qualité de la valeur, et (ii) au niveau de l'agrégat, lorsqu'il s'agit de caractériser la relation entre les entités. Chaque modèle d'accès spécifie quel(s) attribut(s) rechercher avec/sur, ordonner par, ou agréger.

5.2.2 Le modèle de données NoSQL généré :

Les principes de modélisation des données présentés dans la section 4.2, nous permettent de représenter une transition pilotée par requête d'un modèle de données conceptuel amélioré à un modèle de données logique (figure 5.4).

Example : Soit C^{PwR} une collection (telle que définie dans la définition 3

$$C^{PwR} = \{C^{Patient}\{C^{Patient\_records}\}\}$$

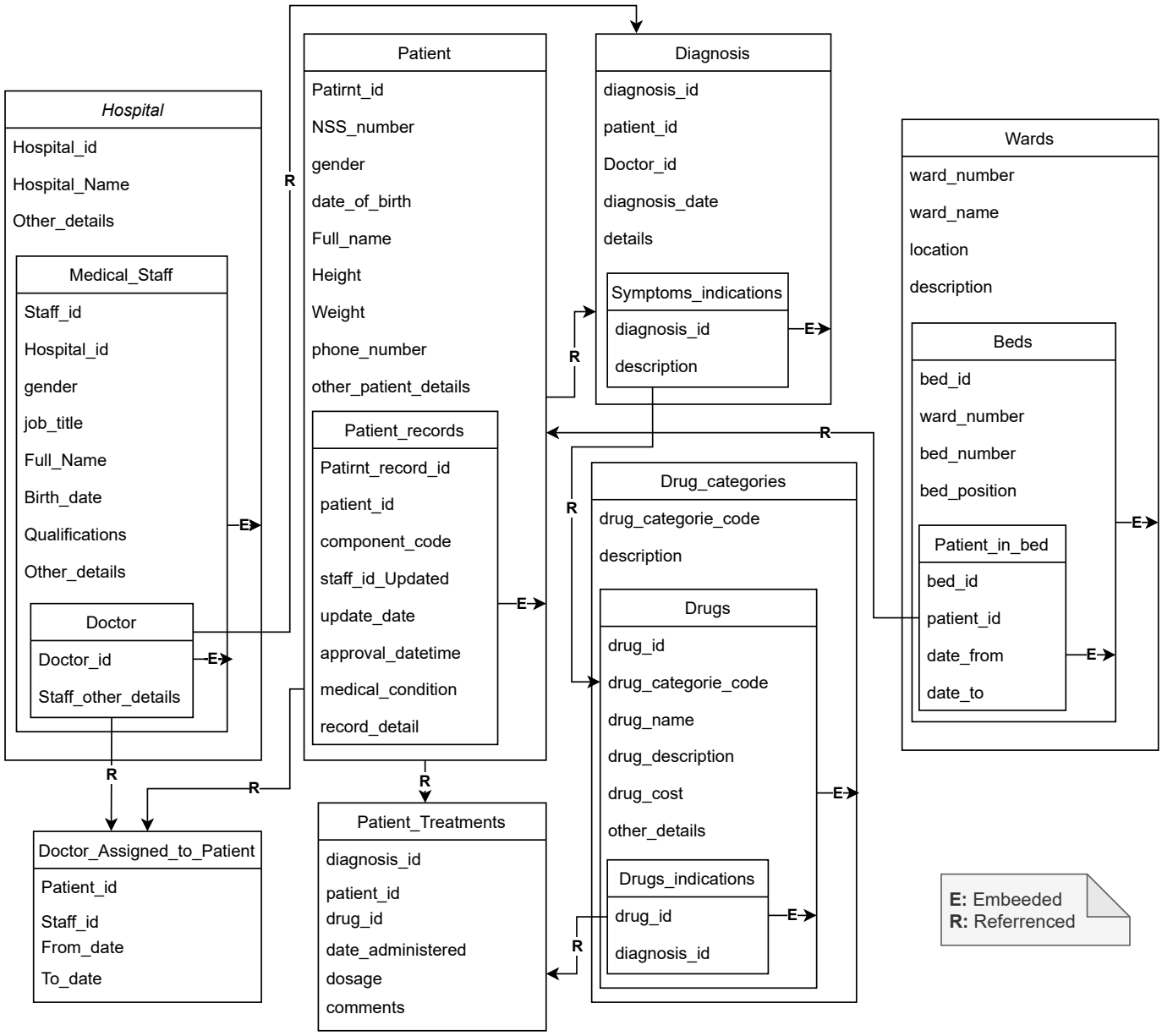


FIGURE 5.4 – Modèle logique orienté-document optimisé

Où

$$C^{Patient} = \{(Att, Val)^{Patient}_{Patient\_Id}, (Att, Val)^{Patient}_{NSS\_number}, (Att, Val)^{Patient}_{gender}, ...\}$$

$$C^{Patient\_records} = \{(Att, Val)^{Patient\_records}_{Patient\_Id}, (Att, Val)^{Patient\_records}_{Patient\_record\_Id}, ...\}$$

Le détail du document $D_{Patient\_record}$ est illustré par la figure 5.5. Par exemple $D_{Patient\_record}$ fournit le numéro d'un dossier de patient, du patient nommé "Younes GUELL" pour la date du 15/01/2021, ayant une condition médicale égale à *COVID-19*, mis à jour par un membre du personnel identifié par son id, .

La collection C^{PwR}, sera définie comme suit :

$$C^{PwR} = \left\{ \begin{array}{c} (Att, Val)^{Patient}_{Patient\_Id}, (Att, Val)^{Patient}_{NSS\_number}, (Att, Val)^{Patient}_{gender}, \\ \\ (Att, Val)^{Patient}_{Full\_name}, ..., (Att, Val)^{Patient\_records}_{Patient\_Id}, \\ \\ N^{Patient\_records} : \{(Att, Val)^{Patient\_records}_{Patient\_record\_Id}, (Att, Val)^{Patient\_records}_{record\_detail}, ...\} \end{array} \right\}$$

Où $Att^{Patient}_{Patient\_Id}$, $Att^{Patient}_{Full\_name}$, $Att^{Patient}_{gender}$ et $Att^{Patient}_{NSS\_number}$ sont des attributs simples, et les autres attributs sont des attributs composés. Ainsi, les valeurs dans les documents imbriqués pour cet exemple sont toutes des valeurs atomiques.

$$\begin{aligned} Val^{Patient}_{Patient\_Id} &= P012021L001 \\ Val^{Patient\_records}_{Patient\_record\_Id} &= 2021103 \\ Val^{Patient\_records}_{staff\_id\_Updated} &= DH001K02 \\ Val^{Patient\_records}_{approval\_datetime} &= 15/01/2021 \\ Val^{Patient\_records}_{medical\_condition} &= COVID-19 \\ Val^{Patient}_{Full\_name} &= Younes\ Guel \end{aligned}$$

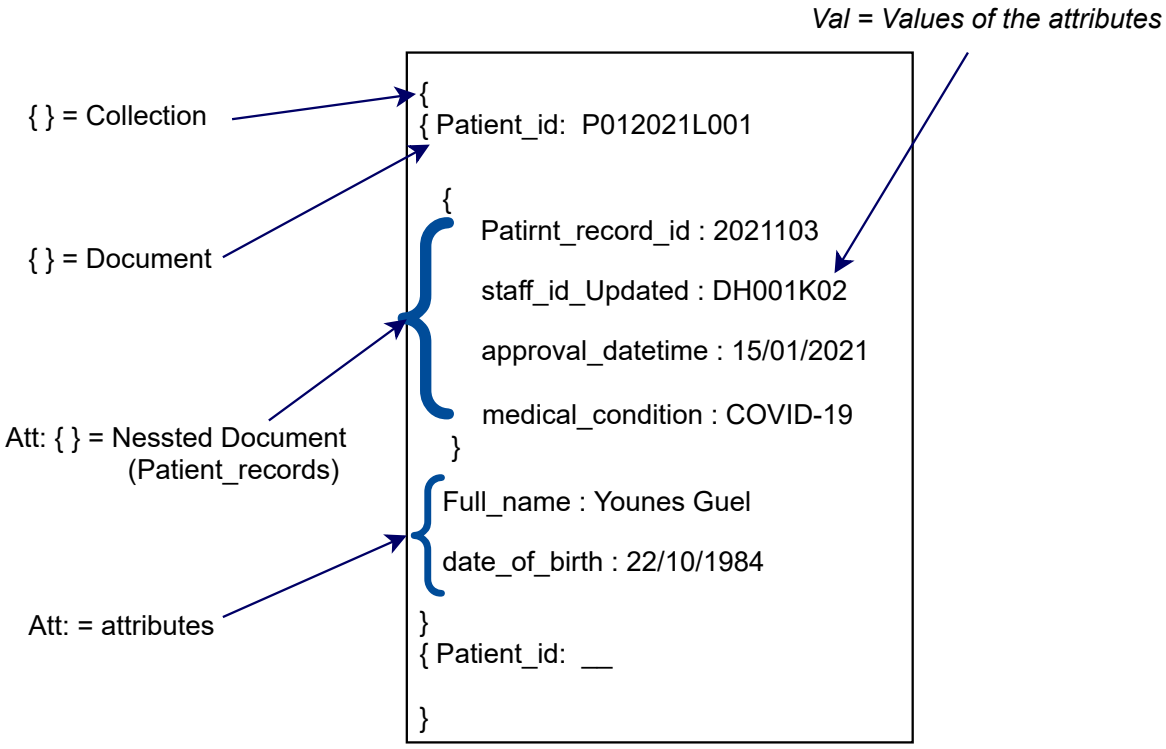


FIGURE 5.5 – Représentation graphique de la collection C^{PwR}

5.3 Conclusion :

Dans ce chapitre, nous avons évalué la méthodologie proposée à l'aide d'une étude de cas réelle dans le domaine de la santé. Nous avons généré des schémas de big data qui prennent en charge les exigences de l'utilisateur (dérivées des 4V's du big data).

À l'avenir, des efforts de recherche devraient être faits pour trouver des réponses à la question de l'évaluation. Et ce, pour que les performances des conceptions de bases de données NoSQL deviennent mesurables et comparables.

Chapitre 6

Conclusion perspectives

"With too little data, you won't be able to make any conclusions that you trust. With loads of data you will find relationships that aren't real... Big data isn't about bits, it's about talent" (Douglas Merrill)

6.1 Conclusion

Ces dernières années, le volume de données à analyser et à interpréter a augmenté de façon exponentielle, et le traitement et l'analyse des big data est l'un des principaux sujets d'intérêt pour les chercheurs et les développeurs dans les environnements industriels. Au sein de ce vaste sujet de recherche, la conception des systèmes big data a suscité un intérêt croissant, car la modèlisation avec des schémas appropriées est cruciale pour extraire des informations voir des connaissances précises des données et ainsi guider les décideurs pour qu'ils prennent des décisions mieux informées. Il est crucial de pouvoir fournir automatiquement la modèlisation la plus appropriée en fonction des besoins particuliers des utilisateurs finaux et de la nature et des caractéristiques des sources de big data.

Cependant, l'élaboration de schémas qui s'adaptent le mieux à un contexte donné est une tâche difficile, surtout lorsqu'on travaille avec de grandes quantités de données adaptées aux fonctions d'analyse en temps réel, dont l'objectif est d'améliorer la scalabilité et la flexibilité.
Le problème est aggravé lorsque les utilisateurs ne sont pas des experts, nous les appelons "Citizen developer", et n'ont pas une idée claire des objectifs pour lesquels ils construisent le projet.

Pour cette raison, dans cette thèse de doctorat, nous avons analysé les besoins actuels en matière de manipulation des données dans les bases de données NoSQL de type document. Par conséquent, nous avons présenté une approche de modélisation basée sur le diagramme des cas d'utilisation et le diagramme de classe de l'UML, dans le but d'améliorer le processus de conception des bases de données NoSQL de type document, qui prennent en charge les quatre Vs du big data.

Nous détaillons ci-dessous les objectifs fixés dans cette thèse et les résultats obtenus pour atteindre chacun d'entre eux.

Objectif1 - Conception d'une architecture de données big data. En systématisant et en formalisant le processus de conception, guidé par l'analyse des besoins des applications cibles en considérant les 4V du big data.

- Pour visualiser le comportement du système, nous utilisons une technique de data-up qui aboutit à un diagramme de cas d'utilisation.
- Nous fournissons également un modèle conceptuel sous la forme d'un diagramme de classe étendu, ainsi que des directives pour réussir la transition d'un diagramme de classe vers un modèle NoSQL orienté document.
- Nous définissons quelques termes utilisés dans la construction du modèle logique orienté document.
- Application de l'approche sur un exemple illustratif pour démontrer l'utilité de notre méthodologie.

Objectif2 - Une étude systématique de la litérature sur les approches consacrées au développement de méthodologies et d'outils destinés à la conception de bases de données NoSQL.

— Élaborer un protocole pour la revue de littérature.

— Définir les stratégie et les questions de recherche.

— Définir les données à extraire de chaque étude primaire.

— Utiliser les directives pour l'établissement des rapports de synthèse.

6.2 perspective

Contrairement aux travaux précédents, les modèles proposés par notre approche peuvent couvrir presque tous les détails conceptuels des bases de données NoSQL orientées documents. Néanmoins, comme notre approche n'en est qu'à ses débuts, elle présente certains limites qui seront traités à l'avenir.

- Nous avons concentré notre attention dans cette thèse uniquement sur les phases de compréhension des données à disposition et des besoins de l'utilisateur.
- Des travaux complémentaires sur les autres diagrammes UML sont en cours et seront présentés dans de futurs travaux, en fonction des résultats prometteurs.
- Nous avons l'intention de développer un benchmark standard pour évaluer notre proposition, ainsi que toute proposition future, afin que les performances de la conception des bases de données NoSQL puissent être mesurées et comparées.

Nous prévoyons également de mettre en œuvre la méthode que nous proposons, c'est-à-dire de créer un outil indépendant du domaine qui sera utilisé comme module d'extension pour les systèmes de bases de données orienté documents. Cet outil, permettra alos de définir un schéma (initial) de l'application.
Une telle mise en œuvre pourrait être adoptée par les fournisseurs de systèmes de bases de données orienté document. Enfin, nous avons l'intention d'étendre la méthode de modélisation des schémas proposée pour modéliser d'autres types de bases de données NoSQL.

References

[1] Michael Stonebraker et Uğur Çetintemel. “" One size fits all" an idea whose time has come and gone”. In : *Making Databases Work : the Pragmatic Wisdom of Michael Stonebraker*. 2018, p. 441-462.

[2] Richard De Courcy. “Les systèmes d’information en réadaptation”. In : *Québec, Réseau international CIDIH et facteurs environnementaux* 5.1-2 (1992), p. 7-10.

[3] Barbara Kitchenham. “Procedures for performing systematic reviews”. In : *Keele, UK, Keele University* 33.2004 (2004), p. 1-26.

[4] Michael Cox et David Ellsworth. “Application-controlled demand paging for out-of-core visualization”. In : *Proceedings. Visualization’97 (Cat. No. 97CB36155)*. IEEE. 1997, p. 235-244.

[5] Doug Laney et al. “3D data management : Controlling data volume, velocity and variety”. In : *META group research note* 6.70 (2001), p. 1.

[6] Ramesh Sharda et al. “Business intelligence and analytics”. In : *System for Decesion Support* (2014).

[7] Efraim Turban, Linda Volonino et Gregory R Wood. *Information technology for management : Digital strategies for insight, action, and sustainable performance*. Wiley Publishing, 2015.

[8] Helena Kościelniak et Agnieszka Puto. “BIG DATA in decision making processes of enterprises”. In : *Procedia Computer Science* 65 (2015), p. 1052-1058.

[9] *How much data is created every day ? - the bigger picture*. Fév. 2022. url : `https://earthweb.com/how-much-data-is-created-every-day/`.

[10] Amir Gandomi et Murtaza Haider. "Beyond the hype : Big data concepts, methods, and analytics". In : *International journal of information management* 35.2 (2015), p. 137-144.

[11] Arne von See. *Total Data Volume Worldwide 2010-2025*. Juin 2021. url : https://www.statista.com/statistics/871513/worldwide-data-created/.

[12] L Lorenzetti. "World Cup scores big on Twitter and Facebook". In : *online] http ://fortune. com/2014/07/14/world-cupscores-big-on-twitter-and-facebook/(accessed July 2014)* (2014).

[13] Chaffey Dave. *What happens online in 60 seconds in 2021 ?* Juin 2021. url : https://www.smartinsights.com/internet-marketing-statistics/happens-online-60-seconds/.

[14] Jinson Zhang. "Big data behaviour modelling and visual analytics". Thèse de doct. 2017.

[15] Carlo Strozzi. "NoSQL : A relational database management system". In : *Lainattu* 5 (1998), p. 2014.

[16] Eric A Brewer. "Towards robust distributed systems". In : *PODC*. T. 7. 10.1145. Portland, OR. 2000, p. 343477-343502.

[17] EF Codd. "A relational model of data for large relational databases". In : *Communications of the ACM* 13.6 (1970), p. 77-87.

[18] Keith D. Foote. *A brief history of non-relational databases*. Juin 2018. url : https://www.dataversity.net/a-brief-history-of-non-relational-databases/.

[19] A Reeve. "Big Data Architectures-NoSQL Use Cases for Key Value Databases". In : *Retrieved from EMC2 InFocus : http ://www. webcitation. org/6hADVIDWr* (2013).

[20] R Ho. "BigTable Model with Cassandra and HBase". In : *Retrieved from Database Zone : http ://www. webcitation. org/6hACnZO92* (2010).

[21] Deepak GC. "A Critical Comparison of NOSQL Databases in the Context of Acid and Base". In : (2016).

[22] Olsen Kyle. *What are the main differences between the four types of ...* URL : https://www.quora.com/What-are-the-main-differences-between-the-four-types-of-NoSql-databases-KeyValue-Store-Column-Oriented-Store-Document-Oriented-Graph-Database.

[23] Pascal Roques et Franck Vallée. "UML 2 en action". In : *De l'analyse des besoins à la conception J2EE, 3ème édition Eyrolles* (2004).

[24] Grady Booch, James Rumbaugh, Ivar Jacobson et al. *Le guide de l'utilisateur UML.* T. 3. Eyrolles, 2000.

[25] Andrew Gemino et Drew Parker. "Use case diagrams in support of use case modeling : Deriving understanding from the picture". In : *Journal of Database Management (JDM)* 20.1 (2009), p. 1-24.

[26] *UML use case diagram tutorial.* URL : https://www.lucidchart.com/pages/uml-use-case-diagram.

[27] Paolo Atzeni et al. "Data modeling in the NoSQL world". In : *Computer Standards & Interfaces* 67 (2020), p. 103149.

[28] Christopher Clayton McConnell et al. *Efficient Denormalization of Data Instances.* US Patent App. 16/740,081. Mai 2020.

[29] Michael Joseph Mior et al. "NoSE : Schema design for NoSQL applications". In : *IEEE Transactions on Knowledge and Data Engineering* 29.10 (2017), p. 2275-2289.

[30] Robin Hecht et Stefan Jablonski. "NoSQL evaluation : A use case oriented survey". In : *2011 International Conference on Cloud and Service Computing.* IEEE. 2011, p. 336-341.

[31] David Gil et Il-Yeol Song. *Modeling and management of big data : challenges and opportunities.* 2016.

[32] Ali Davoudian, Liu Chen et Mengchi Liu. "A survey on NoSQL stores". In : *ACM Computing Surveys (CSUR)* 51.2 (2018), p. 1-43.

[33] Chaimae Asaad et Karim Baina. "NoSQL databases–seek for a design methodology". In : *International Conference on Model and Data Engineering.* Springer. 2018, p. 25-40.

[34] Noa Roy-Hubara et Arnon Sturm. “Design methods for the new database era : a systematic literature review”. In : *Software and Systems Modeling* 19.2 (2020), p. 297-312.

[35] Diana Martinez-Mosquera, Rosa Navarrete et Sergio Lujan-Mora. “Modeling and management big data in databases—A systematic literature review”. In : *Sustainability* 12.2 (2020), p. 634.

[36] Harley Vera-Olivera et al. “Data Modeling and NoSQL Databases-A Systematic Mapping Review”. In : *ACM Computing Surveys (CSUR)* 54.6 (2021), p. 1-26.

[37] Gwendal Daniel, Gerson Sunyé et Jordi Cabot. “UMLtoGraphDB : mapping conceptual schemas to graph databases”. In : *International Conference on Conceptual Modeling*. Springer. 2016, p. 430-444.

[38] Alfonso de la Vega et al. “Mortadelo : A model-driven framework for NoSQL database design”. In : *International Conference on Model and Data Engineering*. Springer. 2018, p. 41-57.

[39] Matthias Sedlmeier et Martin Gogolla. “Design and Prototypical Implementation of an Integrated Graph-Based Conceptual Data Model”. In : *Information Modelling and Knowledge Bases XXVI*. IOS Press, 2014, p. 376-395.

[40] Noa Roy-Hubara et al. “Evaluation of a design method for graph database”. In : *Enterprise, Business-Process and Information Systems Modeling*. Springer, 2018, p. 291-303.

[41] Kwangchul Shin, Chulhyun Hwang et Hoekyung Jung. “NoSQL database design using UML conceptual data model based on Peter Chen’s framework”. In : *International Journal of Applied Engineering Research* 12.5 (2017), p. 632-636.

[42] Karamjit Kaur et Rinkle Rani. “Modeling and querying data in NoSQL databases”. In : *2013 IEEE international conference on big data*. IEEE. 2013, p. 1-7.

[43] Fatma Abdelhédi, Amal Ait Brahim et Gilles Zurfluh. “OCL Constraints Checking on NoSQL Systems Through an MDA-Based Approach”. In : *Int. J. Data Warehous. Min.* 17 (2021), p. 1-14.

[44] Cláudio Lima et Ronaldo Santos Mello. "On proposing and evaluating a NoSQL document database logical approach". In : *International Journal of Web Information Systems* (2016).

[45] Abdullahi Abubakar Imam et al. "Data modeling guidelines for NoSQL document-store databases". In : *International Journal of Advanced Computer Science and Applications, 9* (2018).

[46] Paola Gómez, Rubby Casallas et Claudia Roncancio. "Automatic Schema Generation for Document-Oriented Systems". In : *International Conference on Database and Expert Systems Applications*. Springer. 2020, p. 152-163.

[47] Wenduo Feng et al. "Transforming UML Class Diagram into Cassandra Data Model with Annotations". In : *2015 IEEE International Conference on Smart City/SocialCom/SustainCom (SmartCity)*. IEEE. 2015, p. 798-805.

[48] Xiang Li, Zhiyi Ma et Hongjie Chen. "QODM : A query-oriented data modeling approach for NoSQL databases". In : *2014 IEEE Workshop on Advanced Research and Technology in Industry Applications (WARTIA)*. IEEE. 2014, p. 338-345.

[49] Claudio de Lima et Ronaldo dos Santos Mello. "A workload-driven logical design approach for NoSQL document databases". In : *Proceedings of the 17th International Conference on Information Integration and Web-based Applications & Services*. 2015, p. 1-10.

[50] Pablo Suárez-Otero González, Maria José Suárez Cabal, Pablo Javier Tuya González et al. "Leveraging conceptual data models for keeping cassandra database integrity". In : *WEBIST 2018-Proceedings of the 14th International Conference on Web Information Systems and Technologies, V. 1*. 2018.

[51] Shady Hamouda et Zurinahni Zainol. "Document-oriented data schema for relational database migration to NoSQL". In : *2017 International conference on big data innovations and applications (innovate-data)*. IEEE. 2017, p. 43-50.

[52] D.A.F. Florencio et al. "Which Fits Better ? A Comparative Analysis about NoSQL Key-Value Databases". In : *IEEE Latin America Transactions* 15 (2017), p. 2251-2256.

[53] Avi Silberschatz, Henry F Korth et S Sudarshan. "Data models". In : *ACM Computing Surveys (CSUR)* 28.1 (1996), p. 105-108.

[54] Eric Evans. *Domain-Driven Design : Tackling Complexity in the Heart of Software*. Addison-Wesley, 2004.

[55] Veda C Storey et Il-Yeol Song. "Big data technologies and management : What conceptual modeling can do". In : *Data & Knowledge Engineering* 108 (2017), p. 50-67.

[56] Maria K Krommyda et Verena Kantere. "The Big Data Era : Data Management Novelties for Visualizing, Exploring, and Processing Big Data". In : *Analyzing Future Applications of AI, Sensors, and Robotics in Society*. IGI Global, 2020, p. 87-103.

[57] Anil Sharma, Gurwinder Singh et Shabnum Rehman. "A review of big data challenges and preserving privacy in big data". In : *Advances in Data and Information Sciences*. Springer, 2020, p. 57-65.

[58] Cristian Vidal Silva, Rodrigo Saens, Rodolfo Villarroel et al. "Aspect-oriented modeling : Applying aspect-oriented UML use cases and extending aspect-z". In : *Computing and Informatics* 32.3 (2013), p. 573-593.

[59] JC Stokes. "Managing the development of large software systems-Apollo real-time control center". In : (1970).

[60] JAEJUN YOO, KI-HOON LEE et YOUNG-HO JEON. "Migration from RDBMS to NoSQL Using Column-Level Denormalization and Atomic Aggregates". In : *Journal of Information Science and Engineering* 34.1 (2018), p. 243-259.

[61] Avinash Lakshman et Prashant Malik. "Cassandra : a decentralized structured storage system". In : *ACM SIGOPS Operating Systems Review* 44.2 (2010), p. 35-40.