MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE ABDELHAMID IBN BADIS - MOSTAGANEM

Faculté des Sciences Exactes et d'Informatique

Département de Mathématiques et informatique

Filière : Informatique

RAPPORT DE MEMOIRE

Option : **Réseaux et Systèmes**

Présenté par :

SAYAH Abderrahmane

THEME :

Mise en place d'un cluster pour le traitement des données sémantiques dans un environnement Big Data

Soutenu le : 24/05/2023

Devant le jury composé de :

ABID Meriem	président
ABDALLAH BENSALLOUA Charef	Examinateur
BENHAMED Siham	Encadrant

Année Universitaire 2022-2023

Résumé

L’adaptation du Framework RDF “Resource Description Framework” pour la représentation sémantique des données dans le web est en forte augmentation. Ceci a produit des quantités de données sémantiques à l’échelle du Big Data. Par conséquent, il est nécessaire d’intégrer de nouvelles technologies qui permettent de rendre le traitement des ensembles de données RDF possible étant donné que les solutions classiques des systèmes centralisés rendent la gestion des données en particulier de l’accès à la donnée et sa récupération difficile, voire impossible. Pour ce faire, nous proposons dans ce travail un état de l’art des systèmes distribués de traitement de données RDF à grand échelle qui utilisent des outils de calculs parallèles tout en exposant les notions appropriées dans chaque système.

Mots-clés :

Web sémantique, RDF, SPARQL, calculs parallèles, Big Data, système distribué, Apache Spark.

Abstract

The RDF "Resource Description Framework" adaptation for the data semantic representation on the web is increasing rapidly. This has resulted in Big Data-scale amounts of semantic data. Therefore, there is a need to integrate new technologies that make the processing of RDF datasets possible as classical solutions of centralised systems make data management in particular data access and retrieval difficult or impossible. To this end, we propose in this work a state of the art of large-scale distributed RDF data processing systems that use parallel computing tools while exposing the appropriate concepts in each system.

Keywords:

Semantic web, RDF, SPARQL, Parallel computing, Bigdata, Distributed system, Apache Spark

Dédicaces

Je dédie cet ouvrage

A ma maman qui m'a soutenu et encouragé durant ces années d''études qu'elle trouve ici le témoignage de ma profonde reconnaissance. Ton affection me couvre ta bienveillance me guide et la présence à mes côtés a toujours été ma source de force pour affronter les différents obstacles.

A celui qui m'a fait de moi un homme, mon père.

A mes très chers frères, pour leur appui et leur soutien moral.

A tous mes amis qui m'ont toujours encouragé, et à qui je souhaite plus de succès.

A tous ceux qui j'aime.

Abderrahmane Sayah

Remerciements

On remercie dieu le tout puissant de nous avoir donné la santé et la volonté d'entamer et de terminer ce mémoire.

Tout d'abord ce travail ne serait pas aussi riche et n'aurait pas pu avoir le jour sans l'aide et l'encadrement de ''Mme. Siham benhamed'' On le remercie pour la qualité de son encadrement exceptionnel pour sa patience sa rigueur et sa disponibilité durant notre préparation de ce mémoire.

Notre remerciement s'adresse à nos amis pour leur aide pratique et soutien moral et leurs encouragements.

Notre remerciement s'adresse également à tous nos professeurs pour leurs générosités et la grande patience dont ils ont su faire preuve malgré leurs charges académiques et professionnelles.

Liste des figures

Liste des tableaux

Liste des abréviations

Abréviation	Expression Complète	Page
RDF	Resource Description Framework	1
W3C	World Wide Web Consortium	5
HTML	Hyper Text Markup Language	6
XML	eXtensible Markup Language	6
URI	Uniform Resource Identifier	7
RDFS	Resource Description Framework Schéma	8
OWL	Ontology Web Language	13
SPARQL	Simple Protocol And RDF Query Language	14
SQL	Structured Query Language	14
BGP	Basic Graph Patterns	14
NoSQL	Not only SQL	18
HDFS	Hadoop Distributed File System	24
PRoST	Partitioned RDF on Spark Tables	24
IP	Internet Protocol	48
SSH	Secure Shell	56

Table des matières

Introduction Générale

De nos jours, la représentation des données par RDF (Resource Description Framework) permet de structurer les données de manière sémantique, de les relier entre elles et de les rendre plus facilement échangeables et réutilisables. Cela ouvre la voie à de nombreuses possibilités d'analyse et d'interprétation des données, ainsi qu'à de nouvelles applications et services web.

Le web sémantique [1] permet de structurer les données de manière sémantique, de les lier entre elles, de faciliter leur interopérabilité et leur recherche, et d'automatiser le traitement de données. Cela ouvre la voie à de nombreuses possibilités d'analyse et d'interprétation des données, ainsi qu'à de nouvelles applications et services web.

La taille des données RDF sur le web peut être considérée comme une forme de "big data" en raison de la quantité importante de données générées et stockées. Pour gérer ces volumes de données, des technologies de stockage de données tels que "NoSQL" (Not only SQL) sont souvent utilisées.

Le NoSQL (Not Only SQL) est un domaine de recherche et de développement de bases de données non relationnelles qui a gagné en popularité ces dernières années.

En général, le traitement des données RDF dans le big data peut-être complexe, mais il existe de nombreux outils et techniques pour aider les entreprises à gérer efficacement ces défis.

L’objectif de notre modèle ce que nous avons proposé une solution réseau à l’aide de la création un cluster pour répartition de l’exécution des requêtes sur les nœuds de calcul de notre cluster. Notre fonctionnement de notre modèle et de faire le partitionnement de graphe et l’envoie au cluster de calcul pour faire une exécution distribuée des requêtes pour réduire le temps d’exécution et donner un résultat plus vite.

Ce mémoire est organisé en quatre chapitres. En commençant par une introduction générale, ensuite le premier chapitre nommé état de l'art, le second chapitre parler sur les systèmes de traitement des données RDF, le troisième chapitre discuter sur la conception de notre modèle pour le traitement distribué des données RDF, le dernier chapitre nous avons faire l'implémentation de notre modèle que nous la proposé, Enfin nous terminons notre mémoire avec une conclusion générale pour le travail que nous avons fait.

On parle brièvement du contenu de deux chapitres comme suit :

Chapitre 01 : Etat de l'art

Ce chapitre contient deux partie, la première partie définit le web sémantique et leurs technologies. Et la deuxième partie présentée Le Big data avec la définition. Nous présentons base de données NoSQL avec leurs définition, utilisation, les différents types de bases de données. Ensuite nous avons parlé sur le calcul parallèle, Nous avons touché le modèle MapReduce avec leur outil.

Chapitre 02 : Les Systèmes de traitement de données RDF

Ce chapitre est consacré à la présentation les systèmes de traitement de données sémantiques qui utilisent les outils (Spark, HDFS). Pour chaque système nous avons parlé sur objectif, principe du système Structure du système et leur discussion

Chapitre 03 : Conception d'un modèle pour le traitement distribué des données RDF

Ce chapitre décrit le modèle que nous avons conçu et leur composant principale du notre modèle que la base de données est le cluster pour faire le traitement distribué des données RDF.

Chapitre 04 : Implémentation du modèle proposé

C'est le dernier chapitre, présente les résultats obtenus au cours des expérimentations et l'implémentation effectués pour évaluer notre modèle que nous avons proposée

Chapitre 1 : Etat de l’art

1.1 Introduction

Actuellement, le web sémantique peut être considéré comme une extension du web qui vise à donner un sens aux données, tandis que le Big Data est une capacité à collecter et à traiter de grandes quantités de données. Les deux concepts sont liés dans la mesure où le web sémantique peut aider à structurer et à donner un sens aux données collectées par le Big Data, ce qui permet d'obtenir des insights plus pertinents et plus utiles.

Ce chapitre est constitué en deux parties, on commencer par le concept du web sémantique, et seconde partie présente le concept de big data et le concept de bases de données NoSQL.

L'objectif principal du web sémantique est de faciliter la compréhension et l'exploitation des données par les machines et les humains. Contrairement au web traditionnel, où les informations sont essentiellement des documents écrits en langage naturel, le Web sémantique permet de décrire les connaissances de manière formelle et structurée, en utilisant des langages de représentation de données tels que RDF (Resource Description Framework), OWL (Web Ontology Language), SPARQL (SPARQL Protocol and RDF Query Language) et d'autres.

L'objectif de l'utilisation de technologies NoSQL dans le contexte du Big Data est de permettre l'analyse et la compréhension de ces volumes massifs de données, afin d'en extraire des informations et des insights utiles pour les entreprises et les organisations. En utilisant des technologies NoSQL, il est possible de stocker et de traiter des données de manière rentable, de manière à ce qu'elles puissent être exploitées pour des analyses en temps réel, des prévisions, des prises de décisions et des recommandations.

Partie 01 : web sémantique

1.2 Le web sémantique

Le terme "web sémantique" est un nom composé qui combine les mots "web" et "sémantique" Nous discuterons de leur définition mot pour mot

Web

Il est généralement admis que le web est né en 1989. C'est en fait la date à laquelle le premier document décrivant ce qui est devenu connu sous le nom de World Wide Web a été écrit. Son inventeur était Tim Berners-Lee, alors physicien au CERN, la Commission européenne pour la recherche nucléaire. Le web permet de consulter les pages du site grâce à un navigateur adapté et via des machines telles que des ordinateurs ou des smartphones. C'est un peu comme une immense bibliothèque de documents qui s'appuie sur les fondations complexes d'Internet. Le Web utilise de nombreuses technologies et protocoles pour en faciliter l'utilisation. Citons par exemple l'hypertexte, qui permet de passer d'un site à un autre en cliquant sur un lien ou une URL, qui identifie simplement l'adresse d'un site ou d'un contenu. [2]

La Sémantique

La sémantique c’est une branche de la linguistique, qui étudie le sens dans une langue.[3] C’est une discipline scientifique dont l’objet d’étude le sens des mots, des phrases, des énoncés

1.2.1 Définition du web sémantique

Le web sémantique n'est pas un web à part ou séparé, mais une extension du web actuel où les informations fournies ont un sens bien défini, permettant aux ordinateurs et aux humains de mieux travailler ensemble [1].

Le terme "web sémantique" vient de Tim Berners-Lee, qui a utilisé le concept pour désigner l'évolution du web comme un vaste espace d'échange de données entre les humains et les machines. L'un des objectifs fondamentaux du web sémantique est l'échange de ressources entre les machines pour permettre l'utilisation de vastes quantités d'informations et de services. Les ontologies jouent ici un rôle important car elles supportent la mise en œuvre du Web sémantique. En effet, ils permettent de fournir des vues structurées et partageables des ressources et de définir l'information par une sémantique formelle. [3]

L'ensemble des technologies du Web sémantique est organisé dans une architecture en couches appelée « Semantic Web stack » (Pile du web sémantique). Cette architecture forme la vision du W3C [4] du web sémantique. Elle est présentée dans la figure ci-dessous.

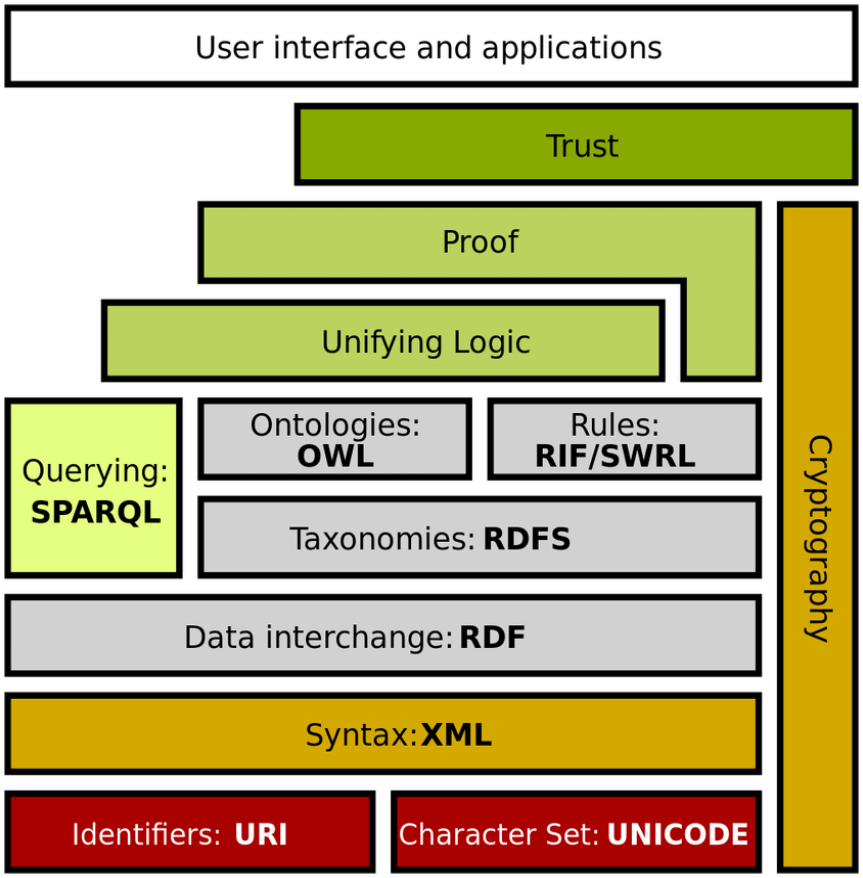


Figure 1 – Pile du web sémantique [5]

Les technologies qu'englobe la pile du web sémantique, y compris les langages et les protocoles, sont standardisées et ont toujours fait l'objet d'efforts internes de recherche et d'amélioration et de standardisation du W3C.

Certaines couches sont déjà implémentées, tandis que des couches supérieures sont en cours de développement. Chaque couche fait référence à la couche inférieure et a une fonction spécifique dans l'architecture [6]. Par conséquent, des langages et des protocoles pour remplir leurs fonctions existent déjà ou sont conçus et créés pour répondre aux spécifications de chaque couche. Dans cette section, nous présentons brièvement les principaux composants du Web sémantique, qui visent à modéliser sémantiquement les connaissances ainsi qu'à interroger et raisonner sur ces connaissances.

Dans le tableau ci-dessous, nous présentons la comparaison entre le web sémantique et le web classique qui est proposé par [7]

Tableau 1 – Comparaison entre web classique et sémantique [7]

Le Web classique	Le Web sémantique
C'est un ensemble de documents	C'est un Ensemble de connaissances
Basé sur HTML	Basé sur HTML & RDF & RDFS & XML & OWL
Recherché par les mots clé	Recherché par les concepts
Utilisable par l'humain	Utilisable par les machines et les humains

La principale différence entre une donnée et une connaissance est que la donnée est une information brute, tandis que la connaissance est une information interprétée et appliquée à un contexte particulier pour générer une signification ou une utilité.

1.2.2 Les technologies du web sémantique

1.2.2.1 RDF

Resource Description Framework (RDF) est un modèle de description de ressource Web [8]. Il fournit l'interopérabilité entre les applications échangeant des informations lisibles par machine sur le web à travers des métadonnées (Les métadonnées sont des données qui décrivent ou définissent d'autres données) [9].

RDF met l'accent sur le traitement automatique des ressources Web. RDF peut être utilisé dans une variété de domaines d'application.

Par exemple : dans la découverte de ressources pour fournir une meilleure fonctionnalité de moteur de recherche, dans les catalogues pour décrire le contenu et les relations de contenu disponibles sur un site Web, une page ou une bibliothèque numérique particulier, pour faciliter le partage des connaissances et les communications, en classifiant le contenu, en décrivant des ensembles de pages représentant un seul "document" logique, en décrivant la propriété intellectuelle des pages Web, en exprimant les préférences de confidentialité de l'utilisateur et la politique de confidentialité du site Web.

RDF avec des signatures numériques sera la clé de la création d'un "web de confiance" pour le commerce électronique, la collaboration et d'autres applications.

Ce modèle est associé à une syntaxe écrite en XML basée sur des triplets :

- Ressource (Sujet) : une entité d'informations pouvant être référencée par un identificateur. Cet identificateur doit être une URI.
- Propriété (prédicat) : l'attribut ou la relation utilisé(e) pour décrire une ressource.
- Valeur (objet) : la valeur d'une propriété associée à une ressource spécifique.

Nous proposons un petit exemple si dessous :

Lionel Messi est un joueur international d'Argentine né le 24 juin 1987.

Tableau 2 – triplet de RDF

Sujet	Prédicat	Objet
Lionel Messi	Nationalité	Argentine
Lionel Messi	Naissance	Id=54
Id=54	Date	24 juin 1987
Id=54	Lieu	Rosario

Et la représentation graphique de triplet de mon exemple que me proposé ce sont

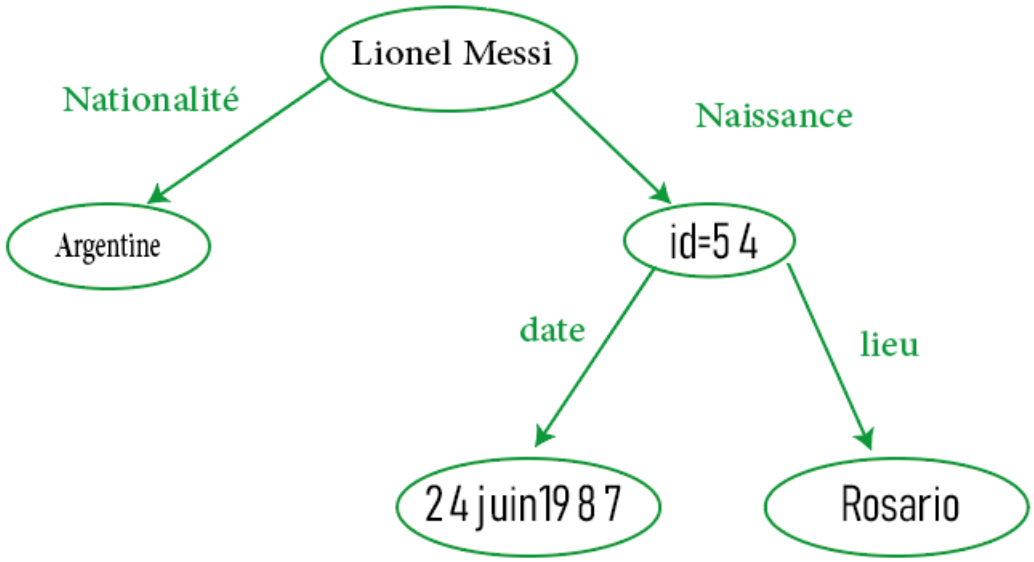


Figure 2 – graphe de triplet RDF

1.2.2.2 RDF Schéma

RDF Schéma ou RDFS est un langage qui permet de décrire un vocabulaire simple dans les modèles RDF [10]. Par conséquent, ce n'est qu'une extension sémantique de RDF. En

utilisant ce vocabulaire, la sémantique des prédicats et les classes d'entités peuvent être définies. Ceci donne la possibilité de déterminer des règles d'inférence afin de générer un grand nombre de triplets à partir de triplets existants, augmentant ainsi les capacités de raisonnement des agents logiciels et des programmes traitant des données du Web sémantique. RDFS permet spécifiquement la définition de liens "Subsomption" entre les classes et les relations à l'aide des primitives rdfs:sub ClassOf et rdfs:subPropertyOf. Ces primitives sont des "spécialisations" ou des liens "is-a" qui permettent aux classes et aux relations d'hériter des propriétés de leurs classes mères (ou de leurs relations) mères.

En autre façon, le RDFS a une architecture de méta modélisation non standard, ce qui fait que certains éléments du modèle ont un double rôle dans la spécification RDFS. En conséquence, cela peut être déroutant et difficile à comprendre et, plus important encore, la spécification de sa sémantique nécessite une théorie des modèles non standard [10]

Et la figure suivante représente un exemple de RDFS

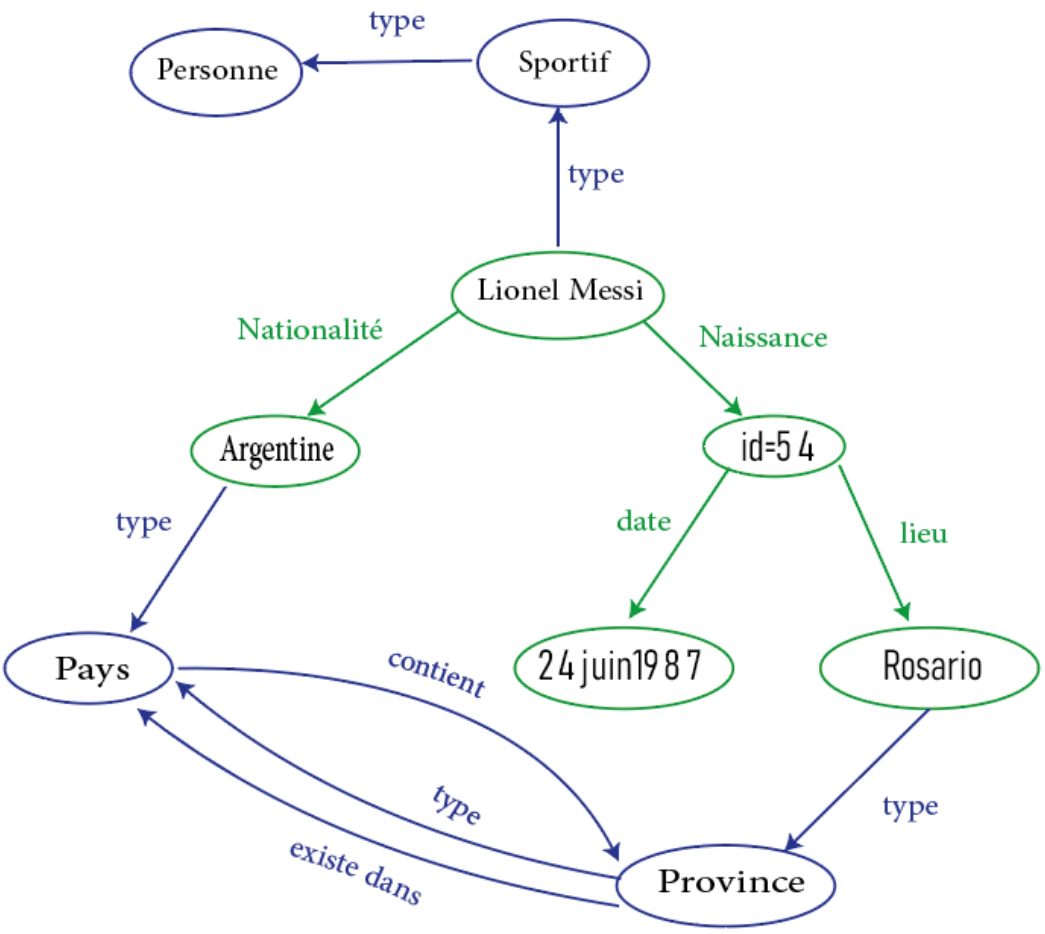


Figure 3 exemple de RDFS

1.2.2.3 Ontologie

Le mot « ontologie » est un terme de philosophie au départ. une ontologie est « une spécification explicite d'une conceptualisation ». Le terme « ontologie » signifie un compte rendu systématique de l'existence en philosophie. [11]

Et dans notre domaine de l'informatique, Une ontologie définit un ensemble de primitives de représentation qui modélisent un domaine de connaissance ou de discours. Les

primitives de représentation sont généralement des ensembles (ou des classes), des propriétés (ou des attributs) et des relations (ou des relations entre les membres de la classe).Les définitions des primitives de représentation incluent des informations sur leur signification et des contraintes sur leur application logiquement cohérente. Dans le contexte des systèmes de bases de données, une ontologie peut être considérée comme un niveau d'abstraction pour les modèles de données, similaire aux modèles hiérarchiques et relationnels, mais visant à modéliser les connaissances sur les individus, leurs attributs et leurs relations avec d'autres individus. Les ontologies sont généralement spécifiées dans des langages qui permettent l'abstraction des structures de données et des stratégies de mise en œuvre ; en pratique, les langages d'ontologie sont plus proches en puissance expressive de la logique du premier ordre que les langages utilisés pour modéliser les bases de données. Pour cette raison, les ontologies sont dites au niveau 'sémantique', tandis que les schémas de base de données sont des modèles de données au niveau 'logique'; ou 'physique' En raison de leur indépendance par rapport aux modèles de données de niveau inférieur. [12]

La figure suivante représente l’exemple de l’ontologie

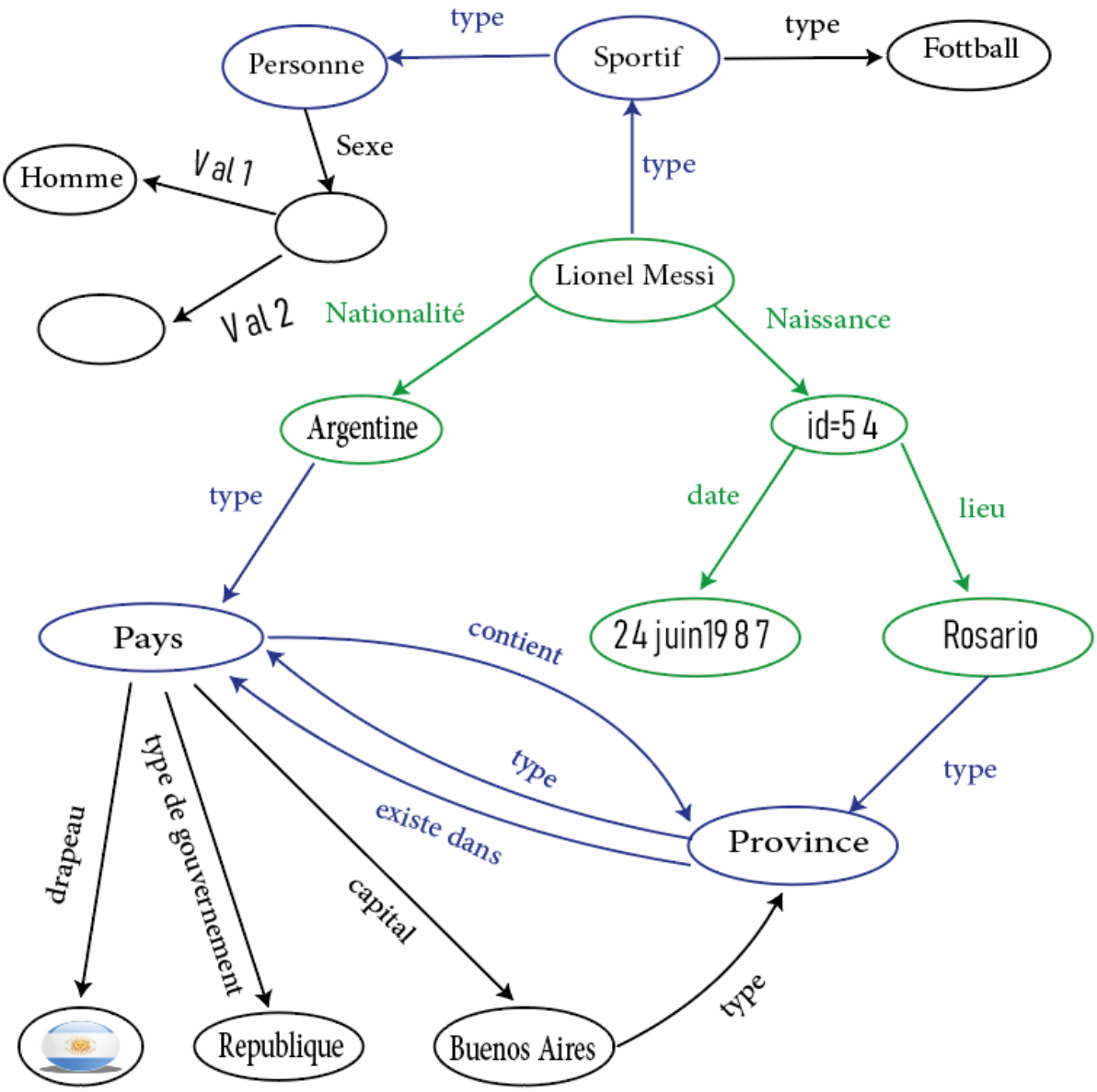


Figure 4 exemple de l'ontologie

1.2.2.3.1 Utilisation de l'ontologie

Les ontologies sont utilisées pour :

- ✓ Intégrer des ressources (base de données) hétérogènes pour faciliter l'accès,
- ✓ Permettre l'interopérabilité entre des systèmes disparates et spécifier des interfaces vers des services indépendants basés sur la connaissance.
- ✓ Partage d'informations et de connaissances, facilitant une compréhension commune de la structure de l'information entre les systèmes ainsi qu'entre les humains et les machines

Dans la pile technologique standard du web sémantique, l'ontologie est appelée couche explicite. [13] Le plus et le mieux langage utilisé de l'ontologie c'est le OWL « Ontology Web Language ».

1.2.2.4 OWL

Ontology Web Language (OWL) est un standard défini par le W3C pour la création d'ontologies depuis 2004. Basé sur le langage DAML+OIL, OWL est basé sur les primitives de base définies par RDF Schémas. Cependant, loin d'être une simple extension de RDF, il fournit toute la sémantique nécessaire à la description des connaissances comme les mécanismes de comparaison de classes (concepts équivalents, symétriques, etc.). En tant que tel, il est plus expressif que RDF et permet ainsi une modélisation plus fine. À sa place de définir un langage de modélisation complexe et difficile à utiliser, le W3C a volontairement choisi de fournir trois sous-langages de OWL de plus en plus expressifs : OWL-Lite, OWL

DL, et OWL Full. Chacun de ces sous-langages est lui-même une extension de son prédécesseur.[14]

Les trois sous langage sont :

- ✓ OWL Lite: Le plus simple sous langage de OWL ,ajoute la possibilité d'exprimer des définitions et des axiomes, ainsi qu'une utilisation limitée des propriétés pour définir des classes .

- ✓ OWL DL: prend en charge les utilisateurs qui souhaitent une expressivité maximale tout en conservant de bonnes propriétés de calcul par apport OWL-lite.
- ✓ OWL Full: est destiné aux utilisateurs qui souhaitent une expressivité maximale sans aucune garantie de calcul. ce sous langage permet l'ontologie d'améliorer la signification vocabulaire prédéfini (RDF ou OWL). [15]

1.2.2.5 SPARQL

SPARQL est un langage de requête standardisé pour interroger les bases de données RDF (Resource Description Framework), qui sont utilisées pour représenter des données structurées sous forme de graphes. SPARQL permet d'interroger des données RDF pour récupérer des informations pertinentes à partir de motifs de graphes, c'est-à-dire des schémas de relations entre les ressources RDF. [16]

SPARQL utilise une syntaxe similaire à SQL (Structured Query Language), le langage standard pour les bases de données relationnelles. Les requêtes SPARQL commencent généralement par des préfixes qui permettent de définir des noms abrégés pour les URI (Uniform Resource Identifier) utilisées dans la requête. Ensuite, la requête peut contenir une ou plusieurs BGP (Basic Graph Patterns), qui définissent les motifs de graphe à rechercher dans la base de données [17]. La requête peut également contenir des filtres pour limiter les résultats de la requête et des clauses pour ordonner les résultats.

SPARQL permet de récupérer des données à partir de sources multiples, telles que des bases de données RDF, des fichiers RDF ou des API de données RDF. Il est également possible de combiner les données RDF avec d'autres formats de données tels que JSON ou XML. SPARQL est largement utilisé dans des domaines tels que la recherche en sciences de la vie, la gestion des connaissances, la sémantique web, l'analyse de données et la fouille de textes.

Nous proposons un exemple simple ci-dessous :

Prefix : Etudiant M2 Resys <URI de la base de données des étudiants de faculté de FSEI.>

SELECT : ?nom ?date de naissance

WHERE {

?Etudiant étudiant M2 Resys : nom ?nom

?Etudiant étudiant M2 Resys : date de naissance ?date de naissance

FILTER(?Etudiant M2 Resys = **<URI de la base de données des étudiants de faculté de FSEI.>**

}

Cette requête récupère les informations (nom et prénom , date de naissance) des étudiants Master2 Resys de la faculté des science exacte et informatique de Mostaganem. Ensuite la requête utilise une BGP (basic graph pattern) pour décrire un motif de graphe minimal que les données doivent respecter pour être retournées.

Le résultat de l'exécution de cette requête est une liste de noms et prénoms des étudiants et de leurs dates de naissance. Par exemple, voici les cinq premiers résultats :

Tableau 3 résultat de requête SPARQL

Nom et prénom	Date de naissance
SAYAH Abderrahmane	18/04/1998
OULD BADJA Yacine	16/05/1997
MCHENGUEL Mohamed	24/07/1999
ZOUDJI Younes	19/08/1999
BELKACMI Rachid	05/09/1997

La BGP est utilisée pour décrire les relations entre les ressources dans la base de données et pour filtrer les résultats en fonction de ces relations. La BGP est l'un des principaux outils permettant d'interroger les bases de données RDF avec SPARQL.

Partie 02 : Big data

1.3 Big data

1.3.1 Définition de Big data

En sens littéraire Les Big Data sont des données trop grandes pour être manipulées et analysé par des protocoles de base de données traditionnels tels que SQL.

Le big data se définit comme suit : des données plus variées ou diversifiées, dans des volumes croissants et à une vitesse plus élevée. C’est ce que l’on appelle les trois « V ».

En d'autres termes, les mégadonnées (big data) consistent en des ensembles de données complexes, dont la plupart proviennent de nouvelles sources. Ces ensembles de données sont si volumineux que les logiciels de traitement de données traditionnels ne peuvent tout simplement pas les gérer. Mais ce déluge de données peut être utilisé pour résoudre des problèmes que vous n'auriez jamais pu résoudre auparavant. [18]

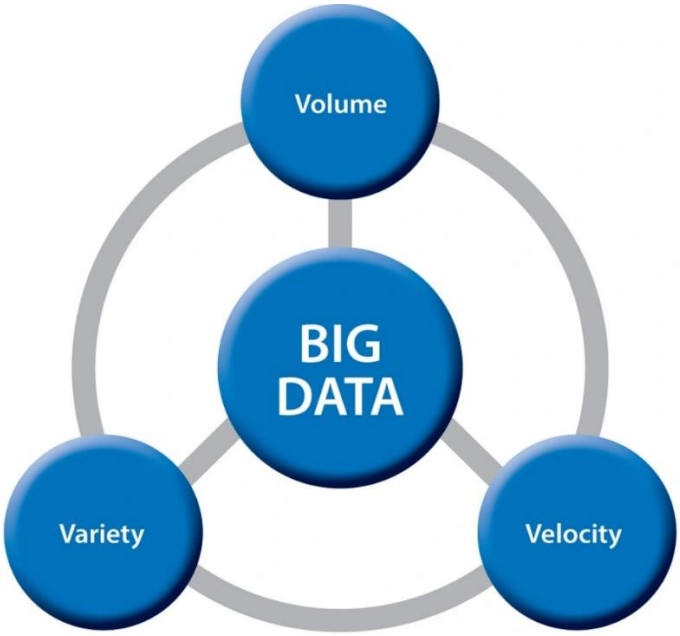


Figure 5 – Les 3V de big data [19]

Ensuite, nous voulons expliquer les Vs important ce sont les trois :

Volume (Données dans le support de stockage) : traiter de grandes quantités d'informations est le principal attrait de l'analyse de données volumineuses. La conséquence est que de nombreuses entreprises ont tendance à stocker une grande quantité de données.

Variété (données sous plusieurs formes). Ces données n'ont pas une structure fixe et se présentent rarement sous une forme parfaitement ordonnée et prête à être traitée.

Vitesse (données en mouvement). La vitesse implique des flux de données, la création d'enregistrements structurés et la disponibilité pour l'accès et la livraison.En effet, ce n'est pas seulement la vitesse des données entrantes qui est le problème : il est possible de diffuser des données à déplacement rapide dans un stockage en masse pour un traitement par lots ultérieur.[18]

Remarque :

Parmi les recherches que j'ai le fait j'ai trouvé que certains chercheurs utilisents 5Vs pour caractériser les Big Data En plus des trois dont j'ai parlé, ils ajoutent Valeur et la Véracité.

1.3.2 NoSQL

1.3.2.1 Définition de NoSQL

NoSQL est un type de base de données dont la particularité est non relationnelle. Ces systèmes prennent en charge le stockage et l'analyse des mégadonnées.

A l'heure du big data, les bases de données relationnelles ne sont plus d'actualité. Pour prendre en charge, stocker et analyser de grandes quantités de données, il faut s'appuyer sur de nouvelles solutions.

Une base de données NoSQL est une base de données «non relationnelle». Les données peuvent être stockées sous une forme non structurée sans suivre un modèle fixe. Les connexions ne sont plus nécessaires et la mise à l'échelle est facilitée.

Le terme « NoSQL » signifie en fait «Not only SQL» (pas seulement SQL). En effet, les bases de données relationnelles utilisent la syntaxe SQL pour stocker et analyser les données.

Ce n'est pas le cas avec les bases de données non relationnelles. Les systèmes NoSQL sont compatibles avec une grande variété de technologies qui permettent le stockage de données structurées, non structurées, semi-structurées ou polymorphes. [20]

La figure ci-dessous résumé le sens de la base de donnée NoSQL

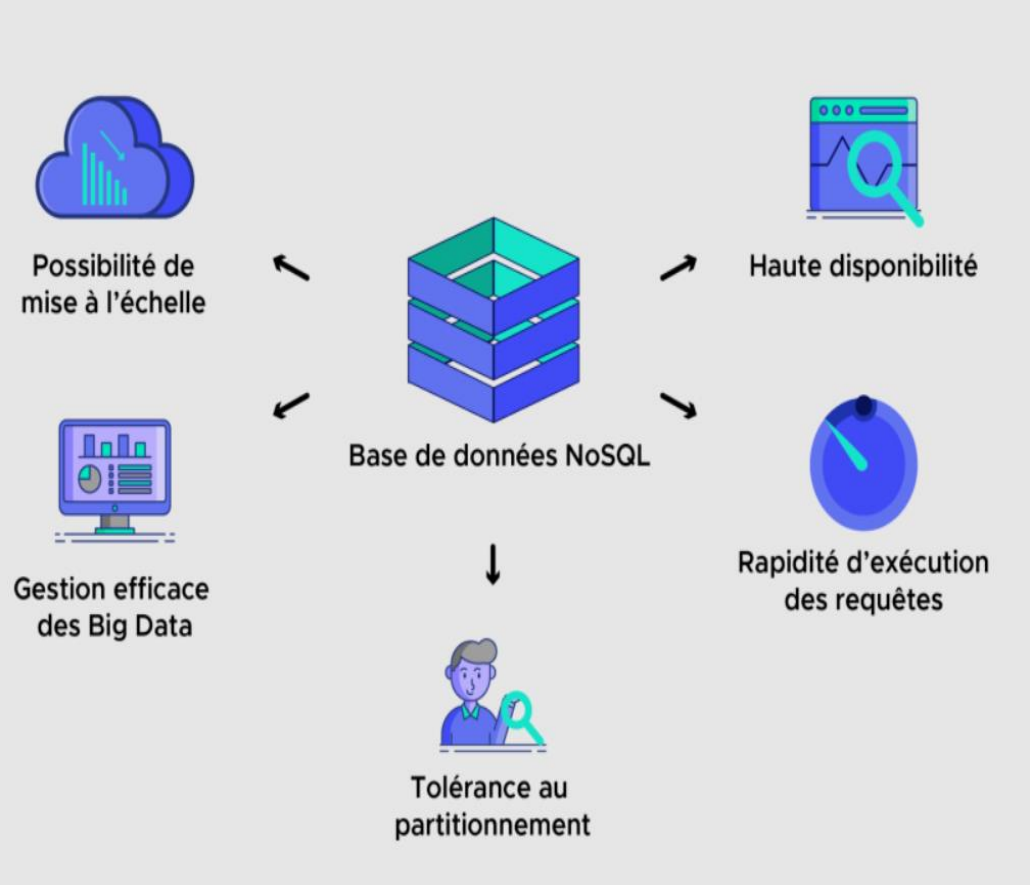


Figure 6 – NoSQL database [20]

La figure représente ou résumé le concept de la base de données NoSQL ça veut dire efficacité de la gestion des Big Data, les requêtes sont exécutées rapidement, la tolérance au partitionnement, la disponibilité en haute. Enfin la mise à l'échelle sont possible.

1.3.2.2 Utilisation de NoSQL

En particulier, les bases de données NoSQL sont utilisées pour le stockage de données distribué qui nécessite une capacité de stockage élevée. Par conséquent, NoSQL est utilisé pour le Big Data et les applications web en temps réel. Les géants de la technologie comme Twitter, Facebook ou Google collectent chaque jour des téraoctets de données de leurs utilisateurs. [20]

1.3.2.3 Les différents types de bases de données NoSQL

Il existe quatre types de bases de données NoSQL

1) Bases de données orientées Clé-valeur.
2) Bases de données orientées Colonnes.
3) Bases de données Graph-Based.
4) Bases de données orientées Document.

- **La première (Clé-valeur) :** Ces bases de données sont basées sur une architecture très basique basée sur HashMap, c'est-à-dire en utilisant une clé pour stocker une valeur, un nombre ou un texte, et ce sera le seul moyen d'y accéder.
- **La deuxième (Colonnes) :** Dans les bases de données relationnelles, les colonnes sont statiques et existent dans chaque ligne, par contre, dans les bases de données orientées colonnes, les colonnes sont dynamiques car elles n'existent que lorsque cela est nécessaire. Dans ce cas, les colonnes peuvent être ajoutées dynamiquement.
- **La troisième (Graph-Based) :** Utilisez une structure graphique pour stocker, cartographier et rechercher des relations. Ils résolvent des problèmes très complexes que les bases de données relationnelles ne peuvent pas résoudre. Par Example (social media: meta, twitter).
- **Le dernier (Document) :** Ce type de base de données contient des documents qui sont considérés comme des lignes de la table. Chaque document contient généralement une clé unique pour l'identifier. [21]

1.3.3 Calcul parallèle

1.3.3.1 Modèle MapReduce

MapReduce est un modèle de programmation et une implémentation associée pour le traitement et la génération de grands ensembles de données. Les utilisateurs spécifient une fonction de mappage qui traite une paire clé/valeur pour générer un ensemble de paires clé/valeur intermédiaires, et une fonction de réduction qui fusionne toutes les valeurs intermédiaires associées à la même clé intermédiaire. De nombreuses tâches du monde réel sont exprimables dans ce modèle. [22]

MapReduce est un moteur de traitement de données qui représente le deuxième composant principale d'Hadoop. Le concept de MapReduce sera d'abord présenté au public par les ingénieurs de Google en 2004 (pour la premiere fois) pour répondre à un problème de création d'index de recherches web. Il s'agit du module de traitement des données (structuré ou non structuré) qui réside dans le système de fichiers distribué Hadoop. L'efficacité et la rapidité de l'approche MapReduce se manifestent dans la capacité à traiter de gros volumes de données en parallèle, dépassant généralement 1 To (téraoctets) sur les nœuds de cluster. [23]

1.3.3.2 Outil de calcul parallèle

Hadoop

Apache Hadoop est un Framework libre (open source) développé en Java, principalement inspiré du déploiement de MapReduce, développé par Yahoo!, en 2009 Mr. Doug Cutting créé Hadoop qui a fait partie de leur projet de la fondation logicielle Apache. L'objectif principal du Hadoop est de remédier le déficit enregistré par les outils traditionnels de traitement et d'analyse de données massives La solution Hadoop se caractérise par l'idée de parallélisation distribué de traitement des données entre les nœuds de calcul qui accélère l'exécution des tâches et diminue la latence à travers des serveurs beaucoup moins couteux, il peut également offrir la possibilité de traitement de tous les types de données (structuré, semi structuré, non structuré).plusieurs compagnie comme Yahoo choisissez Hadoop pour qu'il soit la base de architectures informatiques qui géré activités vitales. [23]

Relation entre Hadoop et MapReduce :

Hadoop garantie l'exécution des applications qui sont traitées par MapReduce sur grande quantité de clusters qui atteindra et des pétaoctets (un pétaoctets égale 10^3 téraoctets) de données en très peu de temps.

Spark

Spark est un moteur de traitement en mémoire pour le Big Data sur un cluster de machines. Il utilise la mise en cache en mémoire et un moteur d'exécution avancé de graphes

acycliques dirigés (DAG) pour créer des plans de requête efficaces pour les transformations de données. Spark exécute des programmes jusqu'à 100 fois plus rapidement en mode de traitement en mémoire et 10 fois plus rapidement en mode de traitement sur disque que Hadoop MapReduce.[24]

1.4 Conclusion

Le web sémantique peut aider à ajouter de la signification aux données Big Data, tandis que les technologies NoSQL peuvent fournir des solutions de stockage et de traitement de données flexibles et évolutives pour les applications Big Data. Les deux concepts sont donc complémentaires et peuvent être utilisés ensemble pour tirer le meilleur parti des données.

MapReduce offre une solution efficace pour le traitement de données distribuées en permettant le traitement parallèle de données volumineuses, la tolérance aux pannes et la flexibilité et à faible cout pour traiter différents types de données. Ces avantages en font un outil de choix pour les entreprises cherchant à traiter de grandes quantités de données dans des environnements de traitement distribué.

Hadoop et Spark sont des outils de traitement de données distribuées offrent une solution évolutive, tolérante aux pannes, flexible et à faible coût pour le traitement de grandes quantités de données distribuées. Leur capacité à traiter différents types de données et à fournir des résultats rapides en fait un outil de choix pour les entreprises cherchant à traiter de grandes quantités de données dans un environnement distribué

Chapitre 2 : Les Systèmes de traitement de données RDF

2.1 Introduction

Les systèmes de traitement de données RDF sont conçus pour permettre la gestion, la manipulation et l'analyse de données structurées sur le web, en utilisant des formats et des protocoles standardisés pour faciliter l'interopérabilité et l'échange de données entre différents systèmes.

Dans ce chapitre nous avons présenté une «étude pour les systèmes de traitement de données RDF et ils sont les suivants Prost, SparkRDF, Approche de Mahmudul et Ecole informatique et enfin l'Approche de Mahmudul et Srividya et pour chaque système ou approche nous avons parlé de Objectif, Principe du système Structure du système et on termines par la Discussion point faible et points fort .

2.2 PRoST

Prost ou (Partitioned RDF on Spark Tables) et proposé par les chercheurs Matteo Cossu et Michael Färber et Georg Lausen.

L'idée principale derrière PRoST est de stocker les données deux fois, partitionnées de deux manières différentes. Chacune des deux structures de données peut être plus bénéfique (en termes de performances) que l'autre sur certains types de requêtes, principalement en raison de la localisation différente des données dans le cluster [25]

2.2.1 Objectif :

Décrire l'approche de Partitioned RDF on Spark Tables (PRoST) qui vise à améliorer l'efficacité des requêtes sur les données RDF, couvrant ainsi un large éventail de requêtes.

Proposer un nouveau modèle de stockage pour charger les graphes RDF dans Hadoop, y compris une stratégie appropriée pour traduire les requêtes SPARQL en plans d'exécution Spark.

2.2.2 Principe du système

Démontrent que leur proposition surpasse les systèmes de pointe (S2RDF et SPARQLGX , Rya) par rapport au temps d'exécution pour un large éventail de types de requêtes et sans aucune phase de précalcule étendue.

Méthode utilisée :

- Partitionnement vertical
- Tables de propriétés

2.2.3. Structure du système

Type (architecture système proposé)

Un cluster contenant un système centralisé au niveau de la base de données et distribuer au niveau de calcul (exécution des requêtés)

Outils utilisés :

- Apache SPARK
- HDFS

HDFS (Hadoop Distributed File System) est un système de fichiers distribué open source conçu pour stocker et gérer de très gros volumes de données sur des clusters de serveurs. Il est principalement utilisé avec Apache Hadoop, une plateforme de traitement et d'analyse de données distribuée.

Architecture de système

Après nous avons lu et compris l'article, nous proposons ci-dessous une architecture de système PRoST

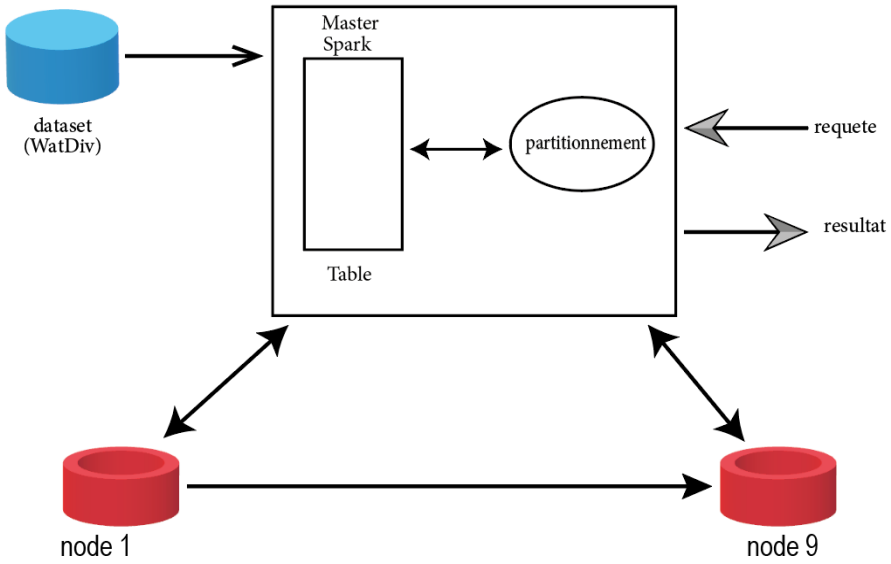


Figure 7 Architecture de Approche PRoST [25]

2.2.4. Discussion

Cette approche est particulièrement appropriée pour les applications du monde réel, pour lesquelles le type de requête et le jeu de données sont inconnus a priori.

Point faible :

Le système n'a pas été comparé par les systèmes et son architecture n'a pas été adapté par ces systèmes

2.3 SparkRDF

SparkRDF se base sur Spark et s'appuie sur le partitionnement du graphe RDF en sous-graphe selon les relations et les classes.[26]

SparkRDF est un moteur RDF basé sur Spark qui partitionne le graphe RDF en MESG (Multi-layer Elastic SubGraphs) selon les prédicats (P) et les classes (C) en construisant cinq types d'indices (C, P, CP, PC, CPC) avec différentes granularités pour supporter une évaluation efficace pour les différents motifs de triplets de requêtes.

SparkRDF crée un fichier d'index pour chaque structure d'index et stocke ces fichiers directement dans le HDFS, qui sont la seule représentation des triplets utilisés pour l'exécution de la requête. Ces indices sont modélisés sous forme de sous-graphes discrétisés résilients (RDSG), une collection de sous-graphes en mémoire partitionnés sur des nœuds.

2.2.1 Objectif

- Réduire l'espace de recherche et éviter la surcharge de mémoire avec présentent un schéma de fractionnement de graphe basé sur **MESG**.
- implémenter le processus de requête dans la mémoire distribuée avec un modèle de requête itératif basé sur **RDSG** est utilisé.
-introduisent également une méthode d'estimation des coûts et plusieurs stratégies d'optimisation sur la base du modèle de données et du modèle de requête.

2.2.2 Principe du système

Démontrer que SparkRDF peut implémenter efficacement des jointures non sélectives plus rapidement que les dépôts (stores) distribués et centralisés à la pointe de la technologie, tout en étant capable de traiter d'autres requêtes en temps réel, en s'adaptant linéairement à la quantité de données.

Outils utilisés :

- Apache Spark
- HDFS

Méthode utilisée :

- MESG

- RDSG

2.2.3. Structure du système

Type (architecture système proposé) :

Un cluster rapide au niveau de la mémoire (distribuée) contenant un système de calcul (nœud de calcul).

Types de l'ensemble de données :

L'ensemble de données largement utilisé c'est LUBM

Architecture système proposé

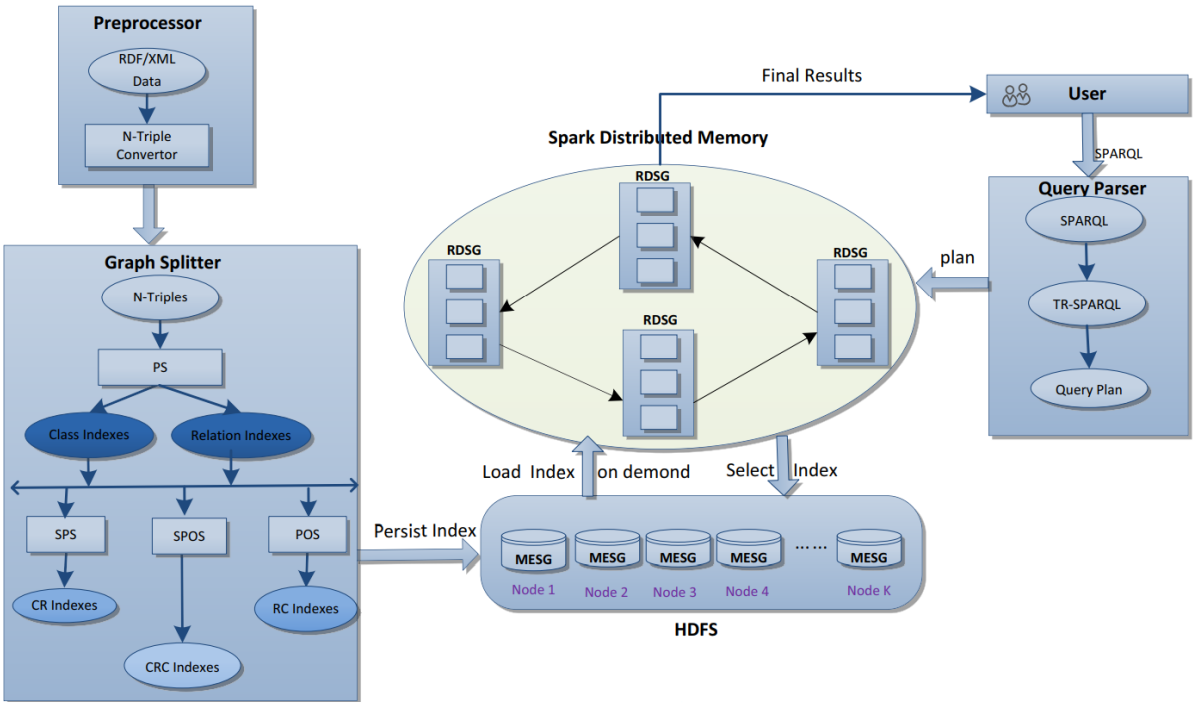


Figure 8 Architecture de système SparkRDF [26]

2.2.4. Discussion

Point fort :

Spark RDF est donné des bons résultats donc on peut dire que SparkRDF est un système de traitement RDF évolutif et en temps réel.

Nous trouvons que plusieurs chercheurs comparants leurs systèmes avec SparkRDF et cela indique que c'est un bon système

Point faible :

Le système s'il se développer présentera des très bons résultats qu'il a donné

Problème traité :

Preuve expérimentale que SparkRDF est un système de traitement RDF évolutif et en temps réel.

2.4 Approche de Mahmudul et Ecole informatique :

Ce système est proposé par le chercheur Mahmudul Hassan en collaboration avec école d'informatique de l'université d'état de Arizona, C'est une Interrogation de données sémantiques sur des bases de données NoSQL avec Apache Spark [27]

2.2.1 Objectif

Hébergement d'ensembles de données RDF massifs dans l'environnement distribué ainsi que pour une livraison plus rapide des réponses aux requêtes, et cela par proposent des systèmes de gestion de données RDF basés sur des systèmes NoSQL avec traitement en mémoire (Spark SQL) qui semblent prometteurs.

2.2.2 Principe du système

Présentent une méthodologie pour le traitement distribué des requêtes RDF à l'aide de bases de données NoSQL et le traitement en mémoire des données à l'aide d'Apache Spark comme moteur de traitement des requêtes.

Ils décrivent le modèle de données et la traduction des requêtes SPARQL vers SPARK SQL pour les schémas de stockage HBase et Cassandra. Afin de réécrire la requête SPARQL en SPARK SQL, ils ont développé un compilateur de requêtes.

Problème traité

Ils ont créé un compilateur personnalisé basé sur le leurs dispositions de données HBase et Cassandra spécifiques parce que les compilateurs existants ne peuvent pas être utilisés avec leur schéma de stockage de données proposé.

Outils utilisés :

- Apache Spark
- Base de données NoSQL
- Systèmes HBase et Cassandra (systèmes de stockage)

Méthode utilisée :

- Schéma de table de propriétés (pour HBase)
- Schéma de partitionnement vertical (pour Cassandra)

2.2.3. Structure du système

Type (architecture système proposé)

Un cluster

Types de l'ensemble de données

- BSBM (Berlin SPARQL Benchmark)
- SP2 Bench (SPARQL Performance Benchmark)

Architecture système proposé

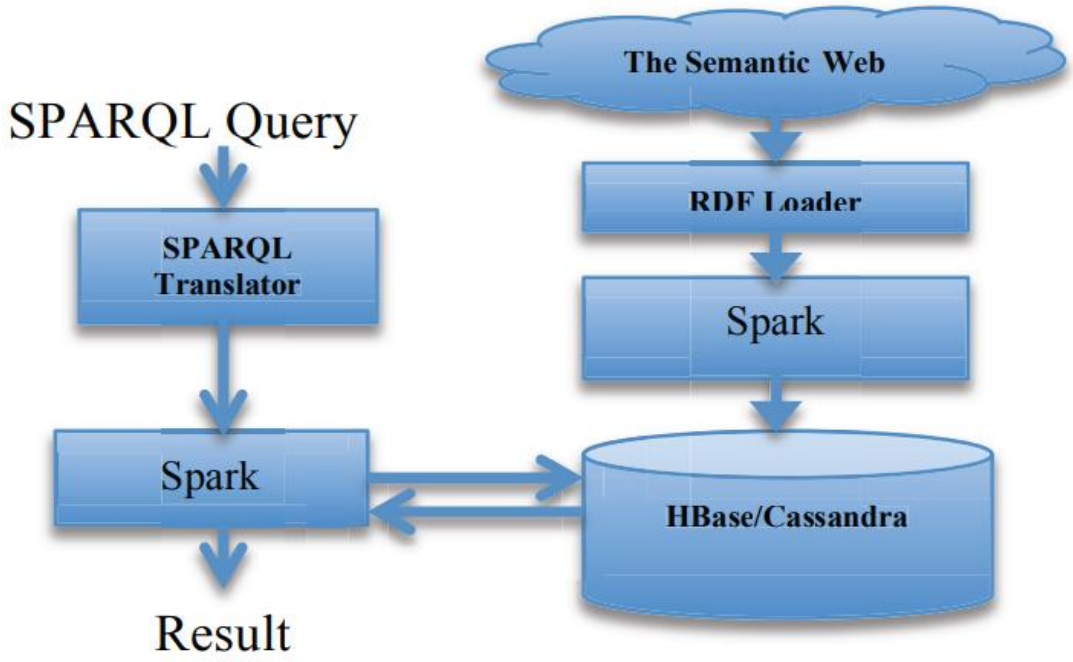


Figure 9 Architecture de l'approche de Mahmudul et Ecole informatique [27]

2.2.4. Discussion

Point fort :

Le traitement de données en mémoire s'est avéré plus rapide pour l'analyse de données à grande échelle par rapport les autres systèmes.

Point faible :

Le point faible de l'Approche stockage des données est centralisé

2.5 Approche de Mahmudul et Srividya :

Ce système est proposé par les chercheurs Mahmudul Hassan et Srividya K. Bansal, C'est un Schéma de partitionnement des données pour une Interrogation RDF distribution efficace

2.2.1 Objectif

Proposer un schéma de partitionnement relationnel qui s'appelle Subset Property Table (SPT) et (en français Tableau des propriétés du sous-ensemble) pour les données Ressourse descreption framework (RDF) qui partitionne davantage l'approche existante de la table de

propriétés en sous-ensembles de tables afin de minimiser l'entrée de requête et l'opération de jointure. [28]

Construit une approche par combinée Subset Property Table plus partitionnement verticale (SPT + VP).

2.2.2 Principe de l'Approche

Démontrent que leur approche proposée et combinée par Subset Property Table & partitionnement verticale surpasse les systèmes de pointe (S2RDF, SPARQLGX, S2X) basés sur un moteur de traitement en mémoire dans un environnement distribué.

Problème traité

Comparent la performance de leur système au autre système de point (S2RDF , SPARQLGX , S2X) par rapport au temps de chargement des données RDF, temps d'exécution des requêtes pour (LUBM & WatDiv) , temps de requête moyen regroupé par type de requête en échelle logarithmique

Outils utilisés

- Apache Spark
- HDFS

Méthode utilisée

- Table de propriété PT.
- Partitionnement verticale VP.

2.2.3. Structure de l'approche

Type (architecture système proposé)

Un cluster de calcul

Types de l'ensemble de données

- WatDiv
- LUBM

Nous proposons ci-dessous l'architecture de système

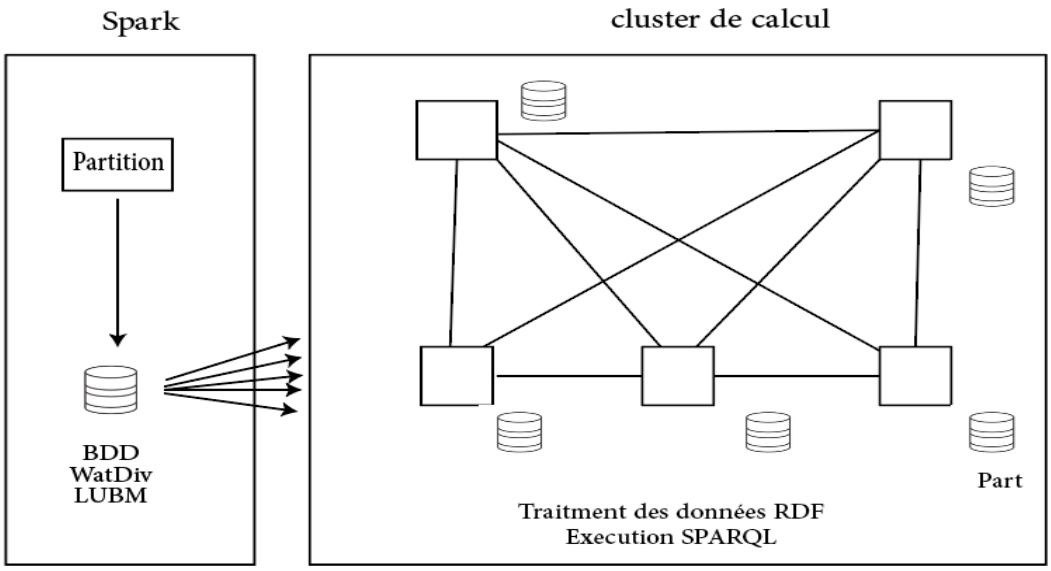


Figure 10 : Architecture de L'approche de Mahmudul et Srividya [28]

2.2.4. Discussion

Point fort :

La solution de gestion (SPT + VP) RDF proposée surpasse tous les types de modèles de requêtes, sauf de quelques requêtes complexes où S2RDF surpasse en raison de son étape de prétraitement étendue.

Point faible :

Le système a encore besoin des améliorations en termes de performances de stockage et de requête en particulier pour les requêtes complexes.

2.6 Conclusion

Le traitement de données RDF (Resource Description Framework) est devenu de plus en plus important dans le domaine de l'informatique et des données en raison de l'explosion des données structurées sur le web. En effet, le RDF fournit un format standardisé pour la description des ressources sur le Web, ce qui permet l'interopérabilité entre différents systèmes et la réutilisation des données. Cela facilite également l'intégration de données à partir de différentes sources et leur représentation de manière cohérente à l'aide d'ontologies partagées. Les systèmes de traitement de données RDF permette également des requêtes efficaces, telles que SPARQL, qui permettent d'extraire des informations précises à partir de grandes quantités de données RDF. Cela permet une recherche et une récupération efficaces de données sur le web

Chapitre 3 : Conception d'un modèle pour le traitement distribué des données RDF

3.1 Introduction

Pour gérer des données Big Data, il est généralement recommandé d'utiliser une solution réseau distribuée pour exploitant les ressources de plusieurs serveurs. Parmi les solutions les plus populaires, nous pouvons citer Hadoop, Apache Spark et Apache HBase.

En utilisant l'une de ces solutions de réseau pour gérer des données Big Data, nous pouvons diviser les tâches de traitement en morceaux plus petits et les exécuter simultanément sur plusieurs serveurs et machines, ce qui permet de réduire le temps de traitement et d'optimiser la gestion de vos données.

Le stockage des données RDF du Web sémantique dans des bases de données NoSQL, qui font partie du Big Data, permet de répondre aux défis spécifiques liés au volume, à la flexibilité et à la performance des données RDF. Ces technologies offrent des solutions évolutives, flexibles et performantes pour gérer et exploiter efficacement les données RDF à grande échelle, tout en s'intégrant à l'écosystème plus large du Big Data

La solution en général de notre travail on a utilisé la base de données NoSQL (not only SQL) pour le stockage de données et on a utilisé un outil de calcul distribuer pour exécution des requêtes

3.2 Modèle

Le modèle proposé permet d'effectuer le traitement d'un ensemble des données sémantique (ensembles des données RDF), Ce modèle contient deux conditions essentielles un serveur de stockage et un cluster de calcul.

Premièrement on a produit cluster contient le Master (serveur de calcul) et trois esclaves/workers lorsque administrateur lancer la requête ou exécuter la requête donc le master partager au les esclaves (nœuds workers) pour récupérer le résultat auquel l'administrateur veut accéder

Deuxièmement on utilise un serveur externe pour le stockage des données (les données sera importer par l'administrateur) et pour partitionner les données et exporter au cluster (workers) et ici le travail du serveur se terminer. Le dessin ci-dessous représente l'architecture du modèle sur lequel nous avons travaillé et qui clarifie les paroles que nous avons déjà prononcées.

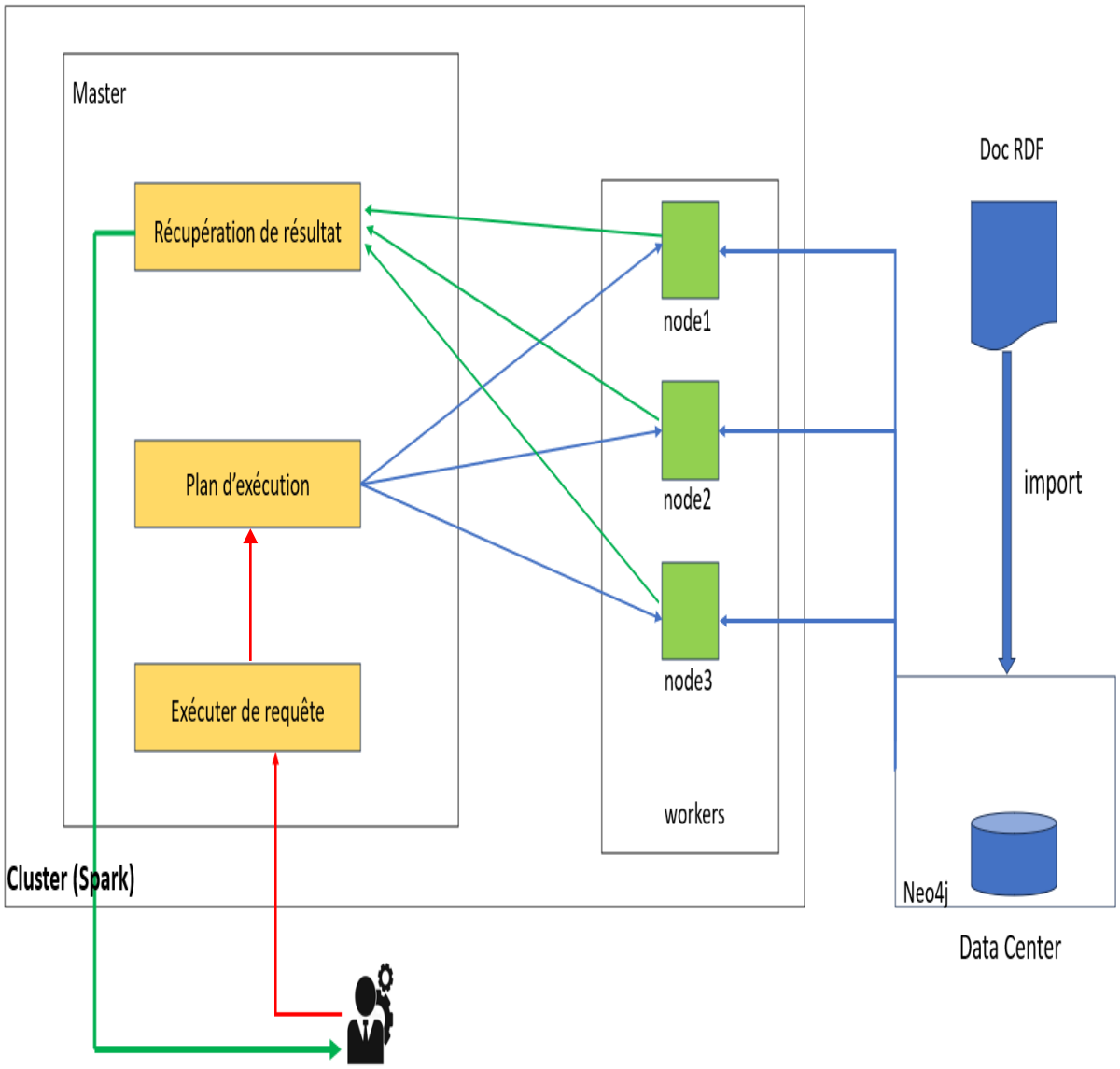


Figure 11 : Architecture du modèle

3.2.1 Principe du modèle :

Ce modèle est présenté par un diagramme de séquence pour nous expliquer une description générale du processus de notre système

Un diagramme de séquence est un type de diagramme de modélisation des interactions entre les objets dans un système, dans le cadre de l'analyse et de la conception orientées objet. Il montre comment les différents objets dans un système interagissent entre eux et dans quel ordre les messages sont échangés entre eux pour accomplir une fonctionnalité particulière. Les diagrammes de séquence sont principalement utilisés pour visualiser et comprendre le comportement dynamique et temporel d'un système.

Le diagramme que nous avons proposé faire les processus que nous mentionnerons. L'administrateur (Admin) commencera par importer un fichier RDF dans le serveur de stockage. Après le serveur de stockage reçoit le fichier RDF et le convertit en graphe. Ensuite, le serveur de stockage partitionne le graphe en plusieurs parties et envoie chaque partition à un worker différent. Puis chaque nœud worker reçoit sa partition respective et la sauvegarde sur un nœud différent. en ce moments l'administrateur lance ou exécute une requête dans le maître (master).Alors le maître reçoit la requête de l'administrateur et la partage avec les workers. Chaque worker traite la requête qu'il a reçue et envoie le résultat correspondant au maître, le maître collecte les résultats de chaque worker et les regroupe. Enfin, le maître affiche le résultat final au niveau de l'administrateur.

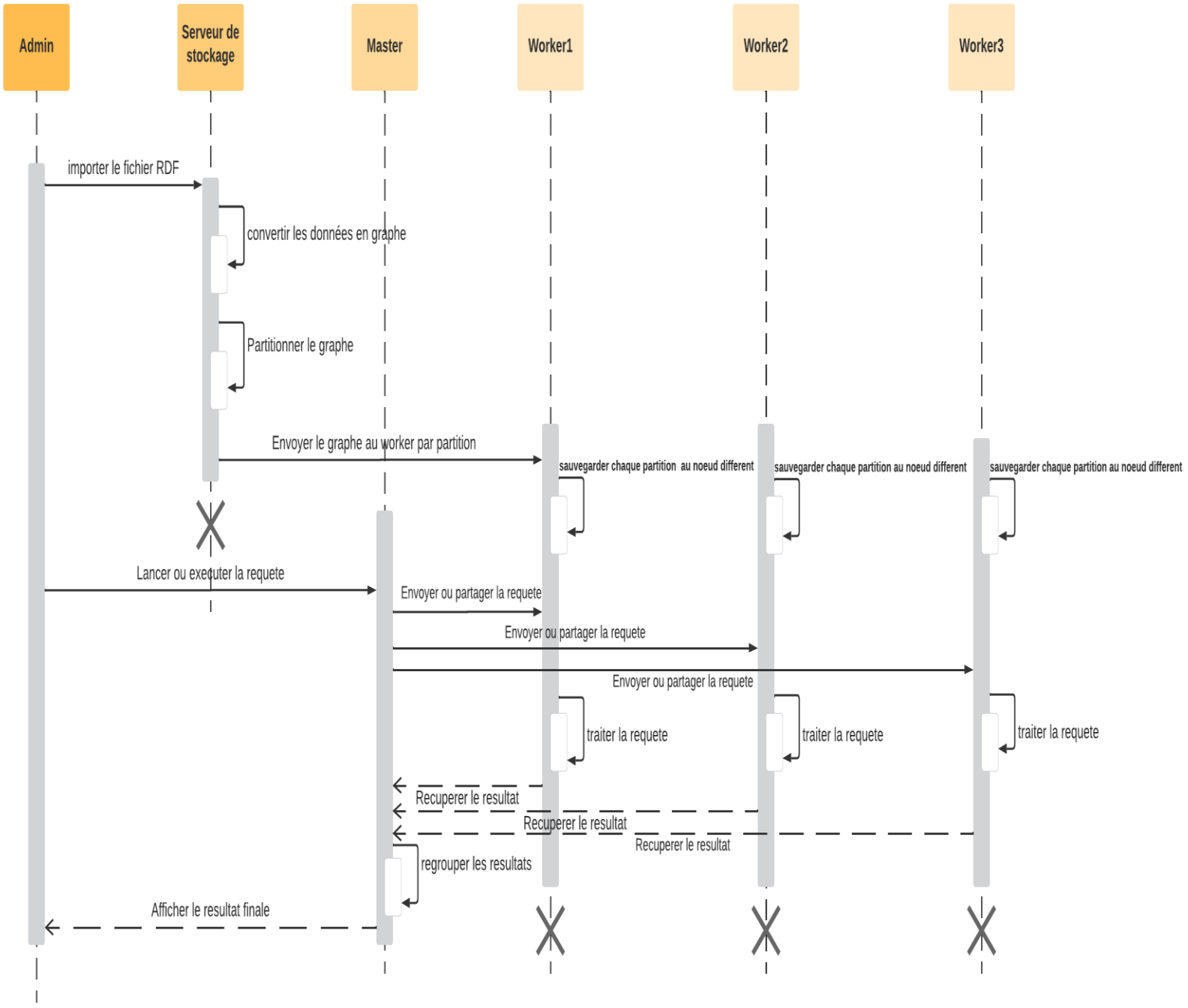


Figure 12 le diagramme de processus de notre modèle

3.2.2 La base de données

Le document qui contient les données e-commerces issus du web sémantique d'un environnement qui produit une grande masse de données et que nous devons le stocker dans un SGBD NoSQL pour qu'on puisse accéder aux données afin de lancer une série de requêtes.

La base de données BSBM (Berlin SPARQL Benchmark) [29] est un ensemble de données de test couramment utilisé pour évaluer les performances des systèmes de gestion de base de données RDF (Resource Description Framework) et SPARQL (SPARQL Protocol and RDF Query Language). Cette base de données contient des informations provenant de magasins en ligne et est composée de 10 millions de triplets RDF.
La structure de la base de données BSBM est décrite en détail dans l'article de recherche intitulé "The Berlin SPARQL Benchmark" publié en 2008 par Christian Bizer, Andreas Schultz et Maximilian Düsterhöft. Cet article décrit la motivation derrière le développement du benchmark, la conception de la base de données, les requêtes de test et les mesures de performance utilisées.
La base de données BSBM [29] simule un scénario d'application de vente en ligne et comprend plusieurs entités principales, telles que les produits, les critiques, les fournisseurs et les offres. Chaque entité est représentée par une table et est liée à d'autres entités par des relations.
Les triplets RDF dans BSBM sont organisés en termes de produits, de clients et d'achats. Les produits sont décrits par des informations telles que le nom, la description, la catégorie, le prix et la disponibilité. Les clients sont décrits par des informations telles que le nom, l'adresse, l'âge et le sexe. Les achats sont décrits par des informations telles que l'ID de commande, le produit acheté, le client, la date de la commande et le prix payé. la figure ci-dessous expliquer une Présentation du modèle de données abstrait

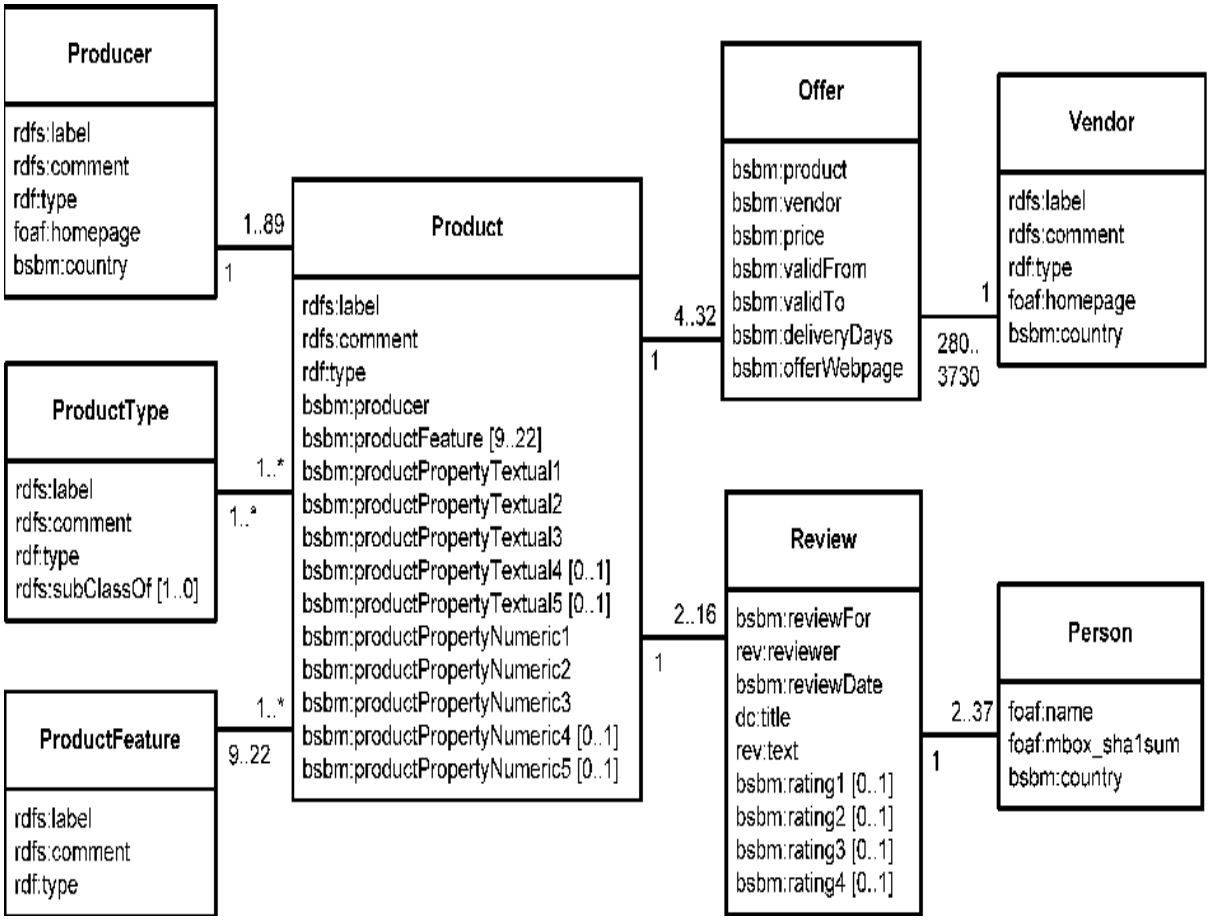


Figure 13 Présentation du modèle de données abstrait [30]

3.2.3 Transformation des données RDF en graphe

3.2.3.1 Fichier NT

Un fichier Nt (Notation3) est un format de fichier RDF (Resource Description Framework) qui est utilisé pour représenter des données liées sur le web sémantique. Le format NT utilise une syntaxe de type texte brut pour écrire des triplets RDF sous forme de sujets, prédicats et objets. Chaque triplet est séparé des autres par des sauts de ligne et des espaces. Les fichiers NT sont

souvent utilisés pour stocker des informations structurées telles que des ontologies, des métadonnées et des bases de connaissances.

3.2.3.2 Neo4j

Neo4j est une base de données orientée graphe open source développée par Neo4j [31], Inc. Elle est conçue pour stocker, gérer et interroger des données sous forme de graphes. Les graphes sont des structures de données constituées de nœuds reliés entre eux par des relations. En autre façon, Neo4j est un système de gestion de base de données orienté graphique qui offre une façon efficace et évolutive de stocker, gérer et interroger des données de graphique à grande échelle. Il est conçu pour représenter et traiter des données fortement interconnectées, ce qui le rend adapté à des applications telles que les réseaux sociaux, les moteurs de recommandation, la détection de fraudes, la gestion de réseaux, et bien d'autres.

Les avantages de Neo4j en tant que système de gestion de base de données orientée graphe incluent la capacité à effectuer des requêtes simples et puissantes sur des données hautement connectées, ainsi que la possibilité de gérer des données non structurées et semi-structurées.

Neo4j est largement utilisé dans divers secteurs, notamment les médias sociaux, la finance, le commerce électronique (E-commerce), la santé, les sciences de la vie, la sécurité, ainsi que dans les domaines de la logistique et de la gestion de l'information.

3.2.3.3 Processus de transformation

Lorsque l’administrateur importer les données d’un document RDF dans le serveur de stockage (les données de Berlin Sparql Benchmark), la première du chose il faut analyser le document si le document est déjà traiter alors il afficher le graphe directement et terminer les taches sinon le système sera extraction les données si le triples n’existe pas alors le graphe sera créer et afficher au administrateur sinon le système créer le nœud S (sujet) après créer le nœud o (objet) pour la création d’une arrêt en sujet et objet, le système faire une boucle pour terminer tous les triples . La figures ci-dessous représente l’organigramme de processus de convertir ou transformer le document RDF sous forme de graphe

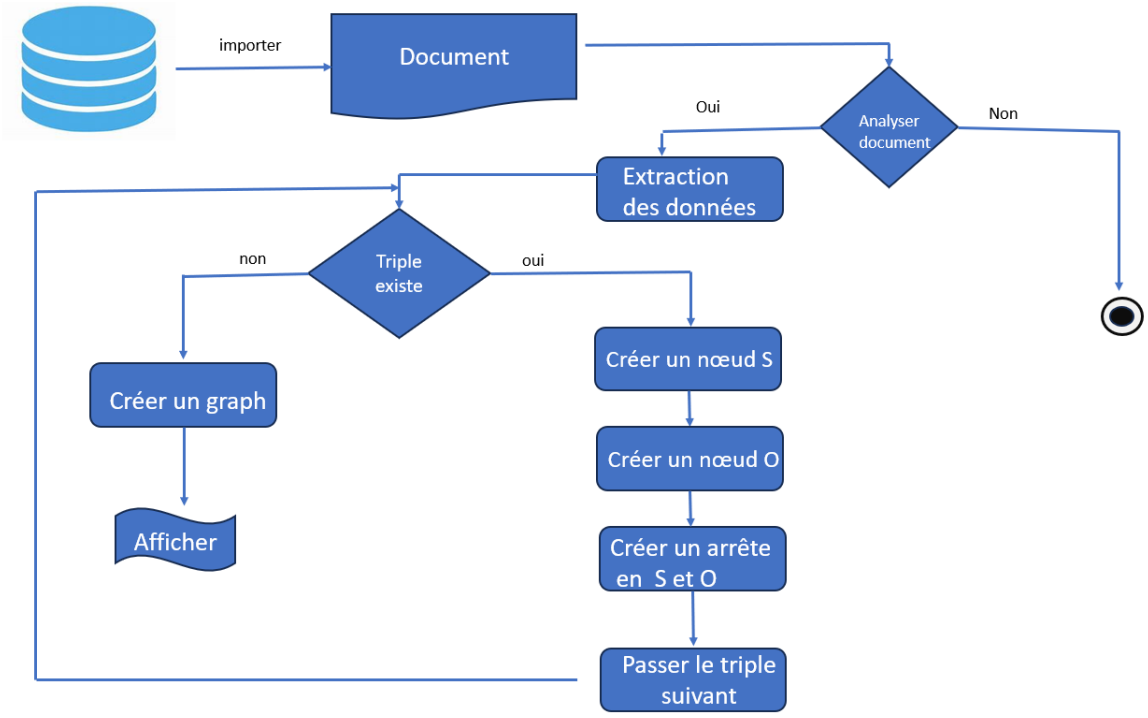


Figure 14 Organigramme de transformation RDF en graphe

3.2.4 Cluster de calcul

Notre cluster que nous avons créé est composé de deux composants principaux qui sont comme suit des nœuds de type workers (esclaves) et un nœud de type Master, et notre cluster sera gérer par Apache Spark.

3.2.4.1 Workers

Ce composant contient trois nœuds esclaves nommées (workernode1, workernode2, workernode3) et chaque esclave contient un sous graphe enregistrer dans un document (le graphe qui représente les données RDF sera partitionner en trois et distribuer au workers), Ce sont eux qui font les calculs.

3.2.4.2 Master

Le Master ou un serveur de calcul c'est le principal élément de notre cluster nommé (masternode) celui qui le gère a travers l'outil (Spark). C'est la machine qui fait la majorité des tâches dans notre cluster nous les mentionnons comme suit :

- L'administrateur de notre modèle lancer ou bien exécuter la requête (SPARQL).
- Le master envoyer la requête au nœud workers pour faire le calcul cette tache appelé le plan d'exécution.
- Lorsque les workers termine les calculs le master récupérer les sous résultats et produit le résultat final à travers MapReduce.
- Le master affiche le résultat qu'il cherché par administrateur.

3.2.4.3 Le plan d'exécution

Lorsque le system reçoit un graphe RDF alors l'administrateur sélectionner une requête dans le master nœud et on a vérifié si le graph n'a pas été partitionner en passer a la deuxième condition de parcour de partition et vérifier si le parcoure terminé, alors la requête sera exécuter au workers qui contient les sous graph pour obtient des sous résultats, avant le dernier le master sera regrouper les sous résultats pour produit le résultat final et afficher au administrateur, la figure ci-dessous représente un organigramme de notre plan de travail de notre cluster

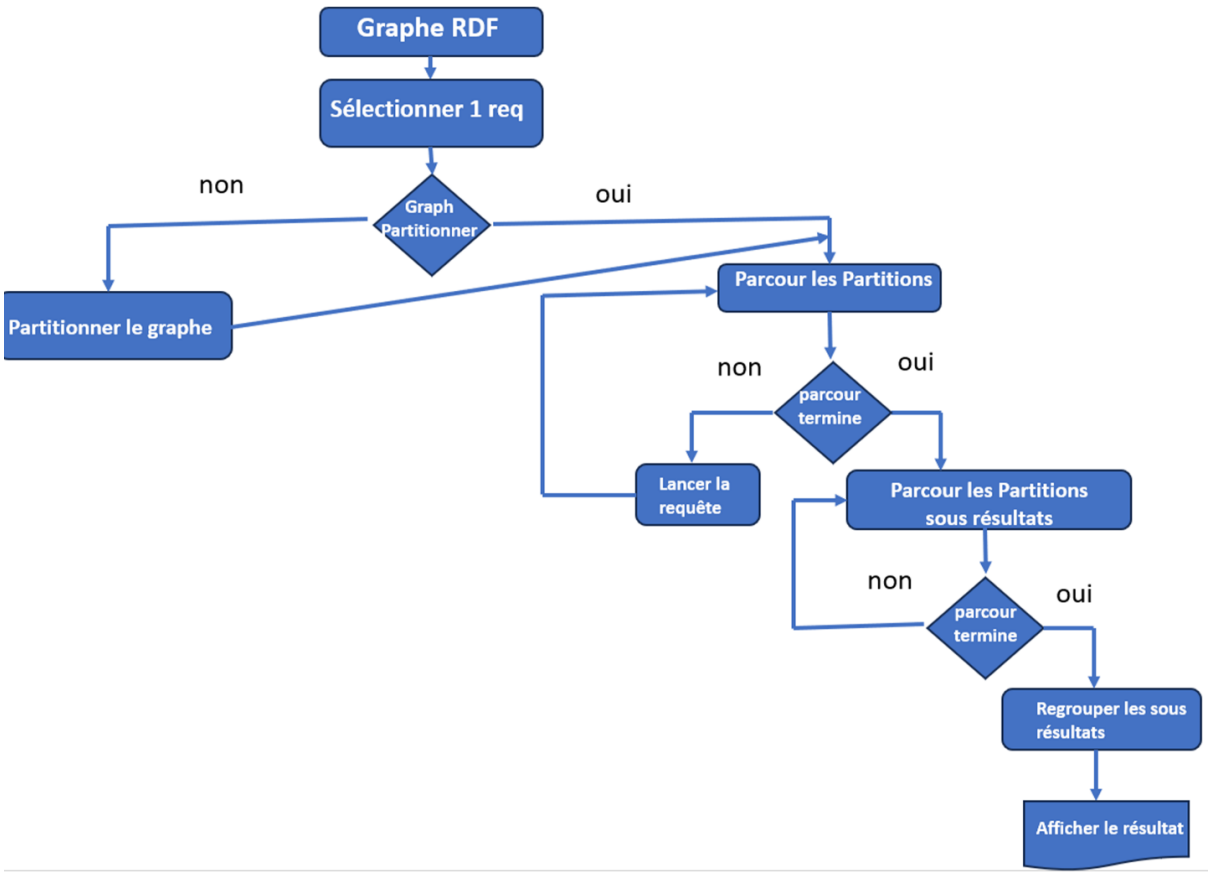


Figure 15 Organigramme de plan d'exécution de cluster

3.2.5 Requête SPARQL

Les requêtes du BSBM sont conçues pour simuler des scénarios d'application réels et représentent différentes opérations de recherche et d'analyse de données. Voici un exemple de requête SPARQL pour le BSBM :

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX                bsbm:                <http://www4.wiwiss.fu-
berlin.de/bizer/bsbm/v01/vocabulary/>

SELECT ?product ?label ?price
WHERE {
          ?product rdf:type bsbm:Product ;
                    rdfs:label ?label ;
                    bsbm:productFeature <http://www4.wiwiss.fu-
                   berlin.de/bizer/bsbm/v01/instances/dataFromRati
                   ngSite1/ProductFeature115> ;
                    bsbm:price ?price .
}
ORDER BY DESC(?price)
LIMIT 10
```

Figure 16 requête SPARQL de BSBM

Cette requête récupère les dix produits les plus chers de la base de données BSBM. Elle sélectionne les variables "?product", "?label" et "?price" pour chaque produit qui a le type "bsbm:Product", une étiquette définie par "rdfs:label", une caractéristique spécifique représentée par "bsbm:productFeature", et un prix défini par "bsbm:price". Les résultats sont triés par ordre décroissant de prix (DESC) et se limitent à dix éléments (LIMIT 10).

La représentation graphique de cette requête dans SPARQL est comme suit :

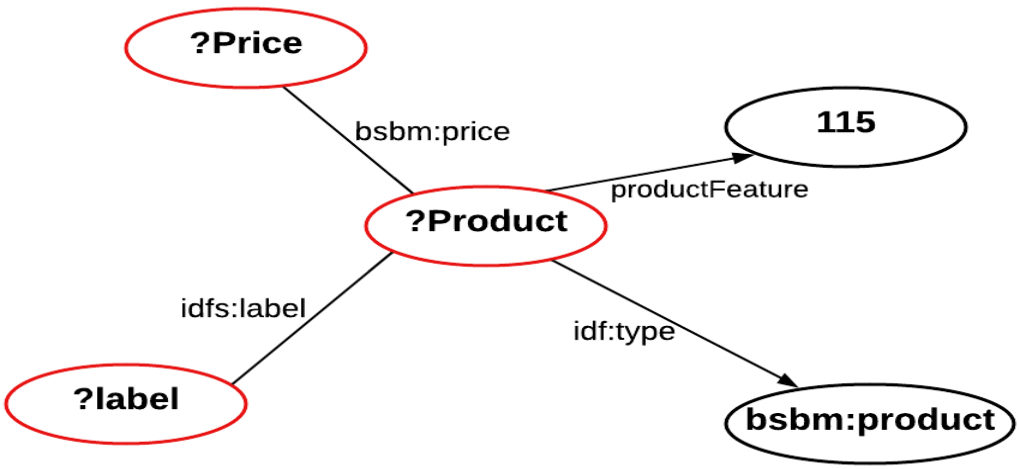


Figure 17 représentation graphique de requête

3.3 Conclusion

Au terme de ce chapitre, nous dériverons les outils du modèle et les principales tâches auxquelles nous nous référons, telles que la conversion des données RDF en graphes et le traitement des données dans les clusters que nous créerons.

Ce travail est fait par la répartition de l'exécution des requêtes sur les nœuds de calcul de notre cluster dont nous avons parlé plus tôt pour obtenir notre objectif qui est trouver une solution réseau pour exécuter une requête dans un environnement big data pour accélérer la mise œuvre et pour la rapidité de trouve les résultats

Dans le chapitre suivant nous allons présenter la mise en place de notre modèle nous utilisant les outils et les logiciels nécessaire pour la réussite de nos travaux.

Chapitre 4 : Implémentation du modèle proposé

4.1 Introduction

Pour valider notre contribution dans le chapitre président, ce chapitre est consacré pour l'implémentation de notre modèle pour obtenir les résultats que nous cherchons à atteindre et pour la mise en œuvre nous avons besoin des hôtes (machines) très puissante pour le faire mais nous essaierons dans notre machine d'une capacité moyenne

La première chose que nous avons faite est d'installer neo4j dans un serveur de stockage pour obtenir les données en graphe , ensuite nous avons créé un cluster de calcul nous avons fait la configuration réseau et la sécurité de notre cluster et nous avons installer apache spark pour la supervision de notre cluster enfin nous avons essayé d'exécuter notre requête SPARQL pour obtenir une résultat .

4.2 Outils de l'implémentation et la configuration machine

4.2.1 VMware Workstation

VMware Workstation 16 Pro (version 16.2.4) est un logiciel puissant de virtualisation qui permet aux utilisateurs de créer, gérer et exécuter des machines virtuelles sur leur ordinateur. Il est largement utilisé dans le domaine professionnel pour le développement de logiciels, les tests de compatibilité, la démonstration de produits et d'autres tâches liées à la virtualisation.

4.2.2 Système d'exploitation (Ubuntu)

Ubuntu (version 22.04.1) est un système d'exploitation basé sur Linux, largement utilisé et apprécié pour sa convivialité et sa stabilité. Il est connu pour sa simplicité d'utilisation et sa compatibilité avec une large gamme de matériels. Ubuntu est développé par une communauté et bénéficie d'un soutien professionnel par Canonical Ltd.

4.2.3 Caractéristique des machines

La machine physique

•Processeur Intel® Core™ i5-6300HQ @ 2.30GHz ,4 cœurs, 4 processeur logique.

• RAM 16.00 Go.

• Disque 1000 Go HDD & 240 Go SSD

• Système exploitation : Windows 10

• Type de système :64 bits.

Les machines virtuelles :

Serveur de stockage : sous le nom DATA

•nombre de Processeurs :2

• RAM 4.00 Go.

• Disque 20 Go SSD

Caractéristique du cluster

Master (serveur de calcul)

• Nombre de Processeurs :2

• RAM 4.00 Go.

•Disque 20 SSD

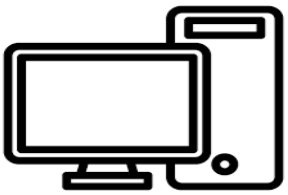
IP:192.168.48.129

workers (trois workers)

• Nombre de Processeurs :1

• RAM 4.00 Go.

• Disque 20 SSD



IP :192.168.48.x

Noeud1 x=131 Noeud2 x=132 Noeud3 x=130

4.3 Etape de création de serveur de stockage

Cette section comprend les étapes obligatoires pour installer Neo4j et importer les données pour la représenter en graphe. Après la création de la machine virtuel qui s'appelle « DATA » la tout d'abord, Nous mettons à jour la liste des packages en exécutant la commande suivante dans le terminal **sudo apt update**

```
data@data-virtual-machine:~/Bureau$ cd /
data@data-virtual-machine:/$ sudo apt update
[sudo] Mot de passe de data :
Réception de :1 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Atteint :2 http://dz.archive.ubuntu.com/ubuntu jammy InRelease
Réception de :3 http://dz.archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Réception de :4 http://dz.archive.ubuntu.com/ubuntu jammy-backports InRelease [108 kB]
Réception de :5 http://dz.archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [672 kB]
Réception de :6 http://dz.archive.ubuntu.com/ubuntu jammy-updates/main i386 Packages [419 kB]
Réception de :7 http://dz.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 Packages [920 kB]
2,348 ko réceptionnés en 3s (705 ko/s)
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
28 paquets peuvent être mis à jour. Exécutez « apt list --upgradable » pour les voir.
data@data-virtual-machine:/$
```

Figure 18 faire mise a jour du système de serveur de stockage

- Étape de installer java

Neo4j nécessite Java pour s'exécuter. Nous pouvons installer OpenJDK en utilisant la commande suivante **sudo apt install openjdk-8-jdk -y**
Ensuite, Accédons au site Web de Neo4j (https://neo4j.com/download-center/) et téléchargeons la version Community Edition de Neo4j pour Linux.

- Étape de installation Neo4j

Nous avons téléchargé le système neo4j, la version 1.5.7, Accédons au site Web de Neo4j (https://neo4j.com/download-center/) et téléchargeons la version Community Edition de Neo4j pour Linux. Ensuite dans le terminal en exécutant la commande **cd bureau/** (pour changer le répertoire de travail dans lequel on retrouve neo4j),ensuite utilisons la commande suivante pour extraire l'archive téléchargée :

tar xvf artifact.php?name=neo4j-community-5.7.0-unix.tar.gz

Ensuite en exécutant la commande **./neo*.App*** pour démarrer neo4j

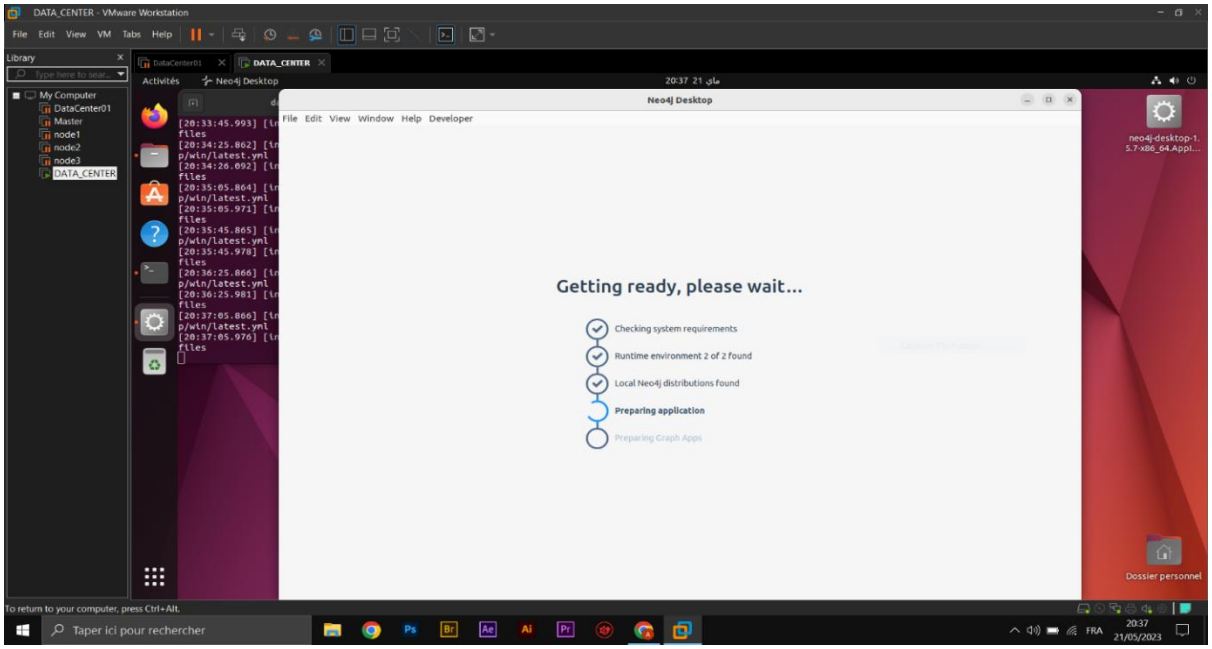


Figure 19 : installation de neo4j

➢ Étape de téléchargements les extensions

Après le démarrage de système en a téléchargé le plugin (extensions) neosemantics et APOC et démarrer la base de données (cliquerons sur start)

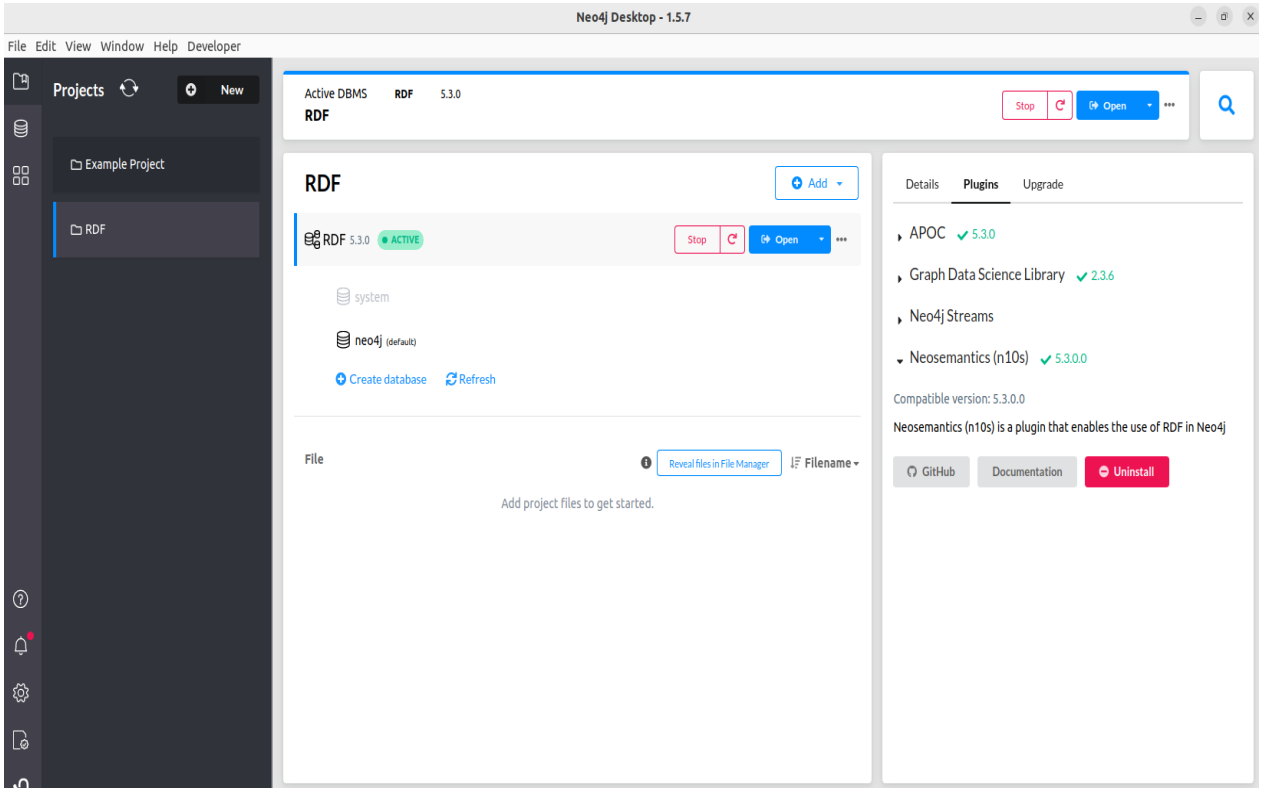


Figure 20 : télécharger les extensions

➢ Étape de Accédons à l'interface de gestion

Ouvrons un navigateur Web et accédons à l'URL suivante : http://localhost:7474

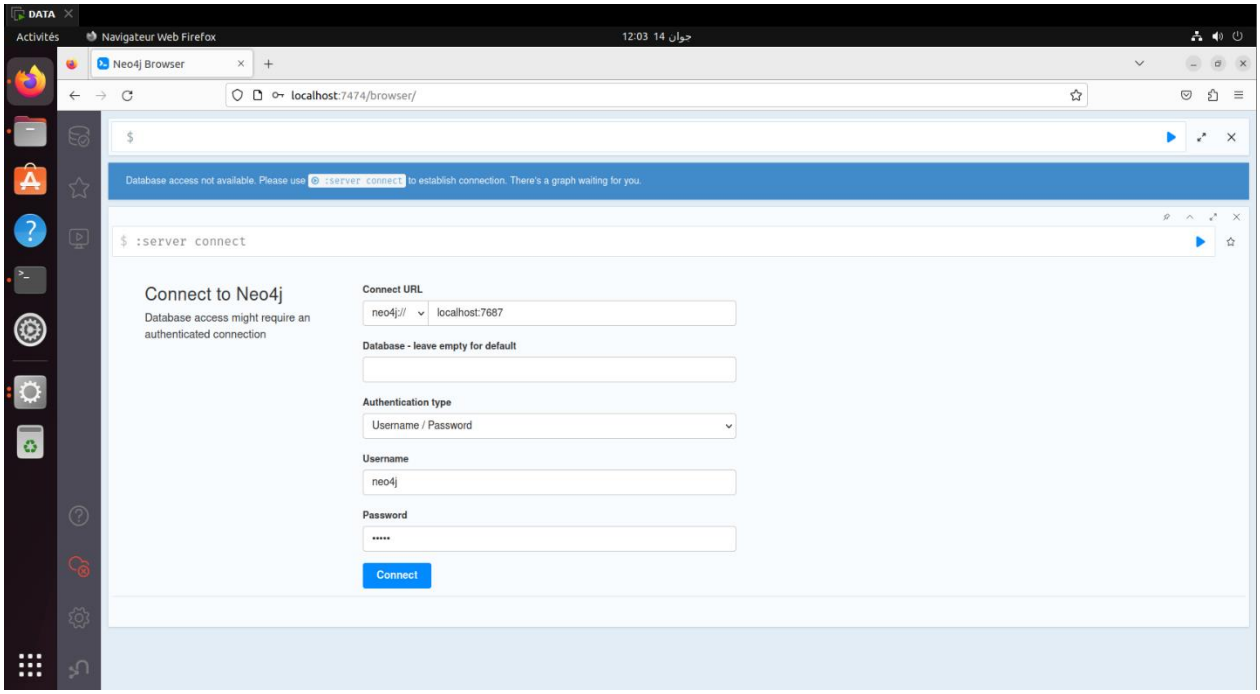


Figure 21 : neo4j Browser

Nous devrions voir l'interface de gestion de Neo4j. Nous pouvons nous connecter avec le nom d'utilisateur par défaut ' neo4j ' et le mot de passe 'neo4j'

Nous avons installé Neo4j avec succès sur Ubuntu 22.04. Nous pouvons maintenant commencer à utiliser Neo4j pour nos travails.

4.4 Etape de création de serveur de cluster de calcul

Cette section comprend les étapes obligatoires pour créer un cluster qui contient un serveur de calcul (Master) et trois nœud esclaves (workers). La première du chose qui avons fait, nous avons créé un server de calcul virtuel nommé (masternode) après nous arrête la machine pour créer les esclaves (workers avec la méthode clone répèterons la méthode trois fois pour créer 3 esclaves.

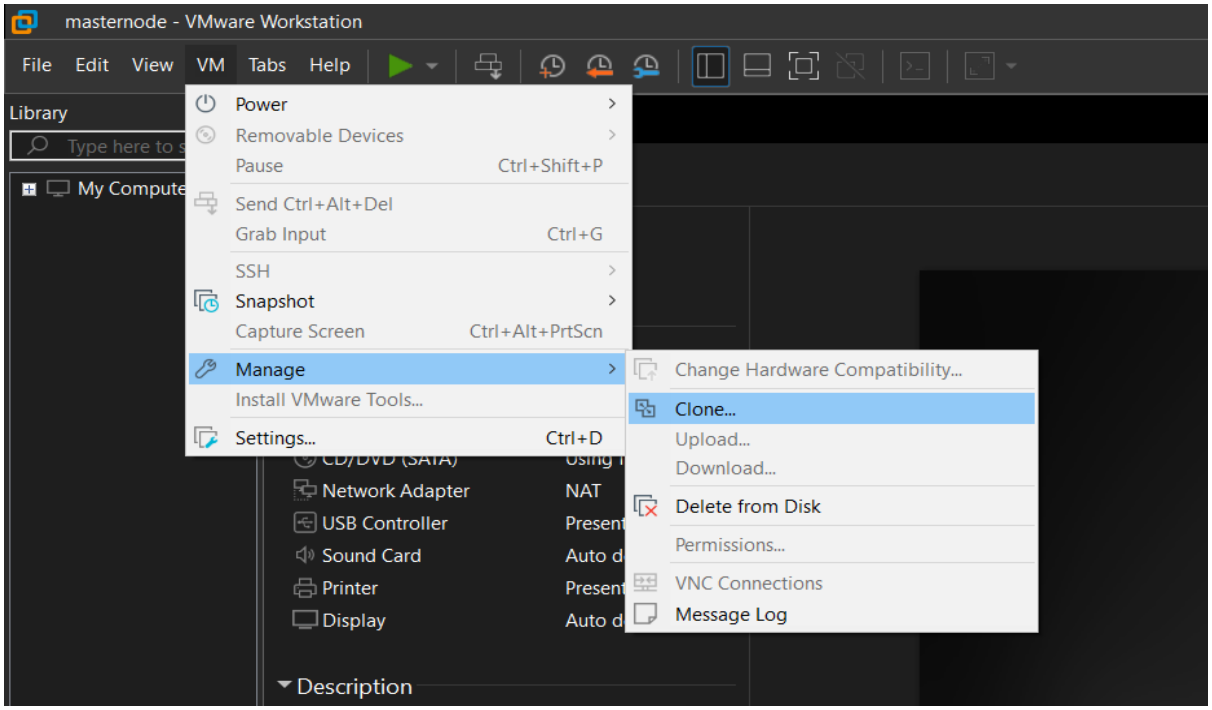


Figure 22 cloner master pour créer les workers

Après la création des esclaves nous lancerons les machines et exécutons la commande suivant **sudo nano /etc/hostname** sur chaque machine pour définir le nom de chaque hôte.

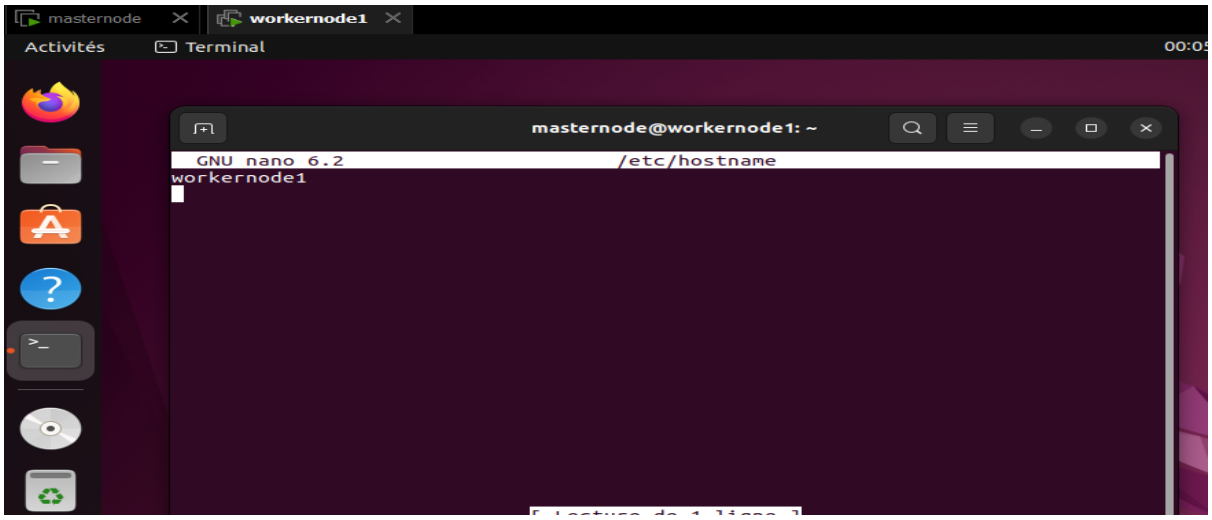


Figure 23 définir le nom de chaque machine

Ensuite, nous expliquerons comment configurer Apache Spark « 3.1.1 » sur notre cluster à plusieurs nœuds, ce qui inclut l'installation du maître Spark et des travailleurs. Nous fournirons des instructions étape par étape pour configurer Spark sur Ubuntu 22.04

- Étape de connaissance l'adresse IP des machines virtuelles

Exécuterons la commande ip addr pour connaitre l'adresse IP des machines virtuelles

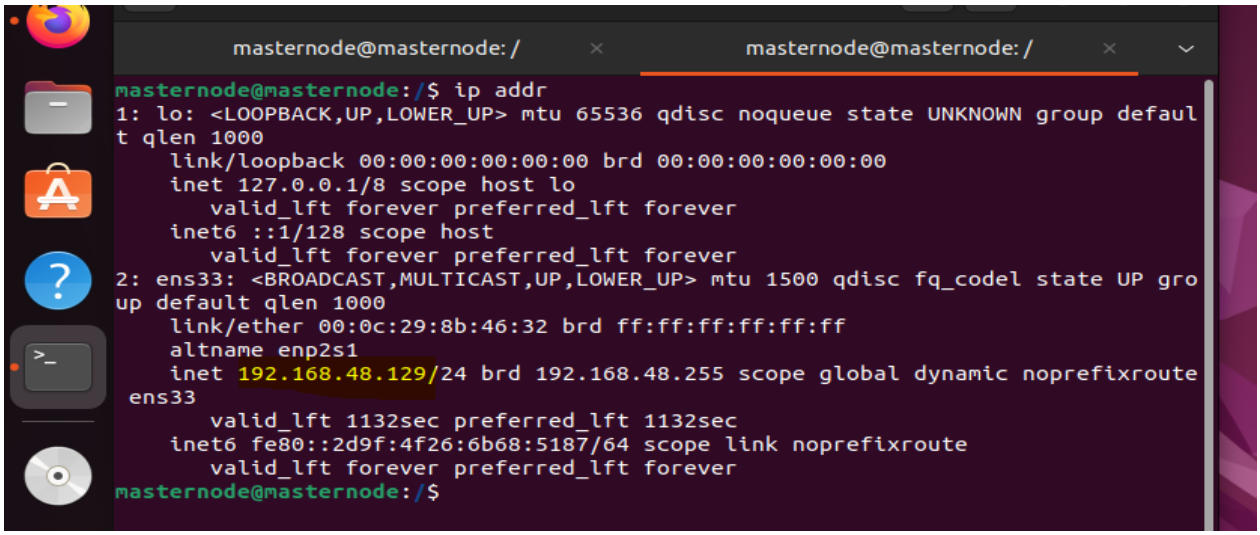


Figure 24 Affichage l'adresse IP des machines

Par exemple l'adresse ip de la machine master c'est 192.168.48.129 avec le masque 255.255.255.0

- Étape de modification le fichier hosts

Nous avons édité le fichier hosts sur tous les machines (master + esclaves) avec exécution de la commande suivants **sudo nano /etc/hosts** et ajouterons des informations sur l'adresse IP et le nom d'hôte, et l'enregistrerons

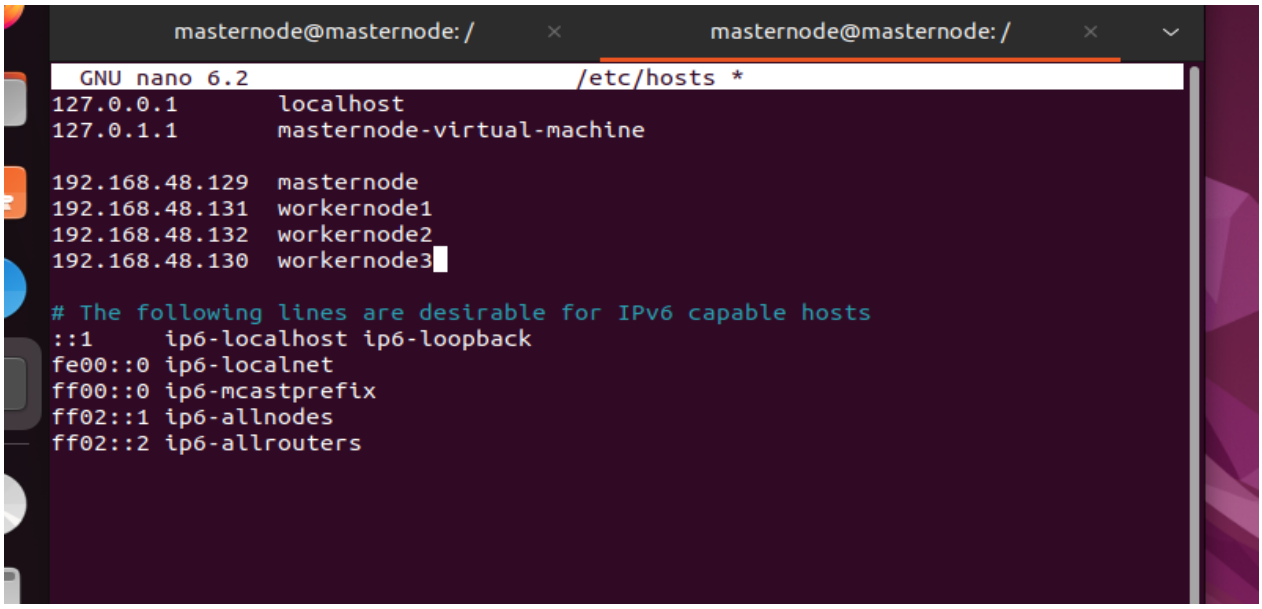


Figure 25 modifier le fichier host

Redémarrerons chaque machine avec la commande **sudo reboot**

- Étape de l'update et installation java

Sur toutes les machines (master & workers) nous ferons les mises à jour et installer java avec les commandes suivantes

- **sudo apt-get update**
- **sudo apt install openjdk-8-jdk -y**

Pour vérifier si java est installer et la version, exécuterons la commande suivante **java -version**

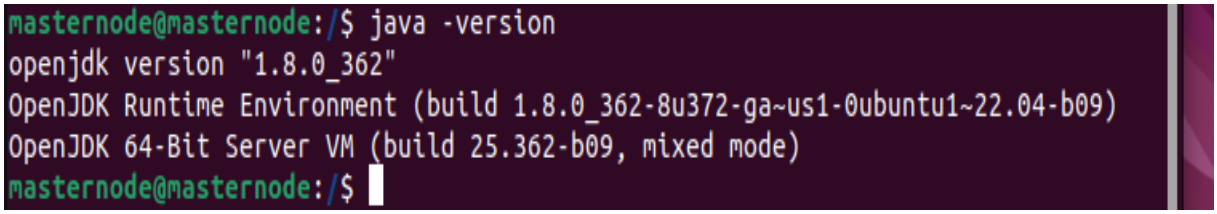


Figure 26 vérifier installation du java

- Étape de configuration SSH

Configurerons maintenant Open SSH server-client sur le maître. Pour configurer Open SSH server-client, exécuterons la commande suivante :

- **sudo apt-get install openssh-server openssh-client**

L'étape suivante consiste à générer des paires de clés. Pour cela, exécuterons la commande suivante :

- **ssh-keygen -t rsa -P ""**

Exécuterons la commande suivante pour autoriser la clé :

- **cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys**

  Nous avons copie maintenant le contenu du maître de formulaire .ssh/id_rsa.pub vers .ssh/authorized_keys (tous les travailleurs/esclaves ainsi que le maître). exécute les commandes suivantes :
  ssh-copy-id masternode@192.168.48.129

ssh-copy-id workernode1@192.168.48.131

ssh-copy-id workernode2@192.168.48.132

ssh-copy-id workernode2@192.168.48.130

Il est maintenant temps de vérifier si tout est correctement installé. Exécuterons la commande suivante sur le maître pour connecter aux esclaves/travailleurs : **ssh 192.168.48.131**
et pour quitter la machines esclaves exécuter la commande **exit**

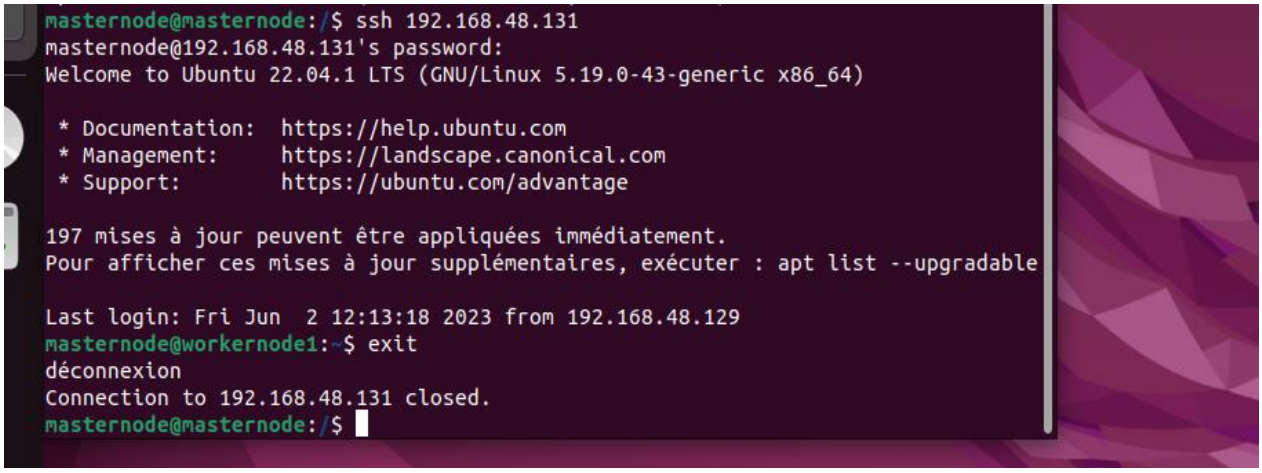


Figure 27 vérification de la configuration du SSH

- Étape de installation Spark

Téléchargeons la version stable d'Apache Spark. Nous allons installer spark-3.1.1 avec Hadoop-3.2. Pour télécharger spark-3.1.1 avec Hadoop-3.2, exécuterons la commande suivante :

- **wget https://archive.apache.org/dist/spark/spark-3.1.1/spark-3.1.1-bin-hadoop3.2.tgz**

Pour décompresser le fichier Spark tar Exécuterons maintenant la commande suivante :

- **tar xvf spark-3.1.1-bin-hadoop3.2.tgz**

Exécuterons la commande suivante pour déplacer les fichiers Spark vers le répertoire Spark (/usr/local/bin) :

- **sudo mv spark-3.1.1-bin-hadoop3.2 /usr/local/spark**

Pour configurer l'environnement pour Apache Spark, nous devons modifier le fichier .fichier bashrc . Exécutez la commande suivante pour modifier le fichier .bashrc :

- **sudo nano ~/.bashrc**

ajoute la ligne suivante au fichier et enregistrez.

- **export PATH=$PATH:/usr/local/spark/bin**

La ligne ci-dessus définit l'emplacement (Path) où se trouve le fichier du logiciel Spark dans la variable PATH.

Exécute la commande suivante pour apporter des modifications effectives au fichier .bashrc :

- **source ~/.bashrc**

➢ étape de configuration Spark Master

Pour configurer la configuration Apache Spark Master, Nous avons modifie le fichier spark-env.sh. Accéder au dossier spark/ conf et faites une copie du fichier spark-env.sh. fichier de modèle en tant que spark-env.sh

- **cd /usr/local/spark/conf**
- **cp spark-env.sh.template spark-env.sh**

Maintenant, pour modifier le fichier de configuration spark-env.sh , exécutez la commande suivante :

- **sudo nano spark-env.sh**

Ajouter les paramètres suivants (ligne de code) à la fin du fichier, enregistrez et quittez.

- **export SPARK_MASTER_HOST=192.168.48.129**
- **export JAVA_HOME=</usr/lib/jvm/java-8-openjdk-amd64>**

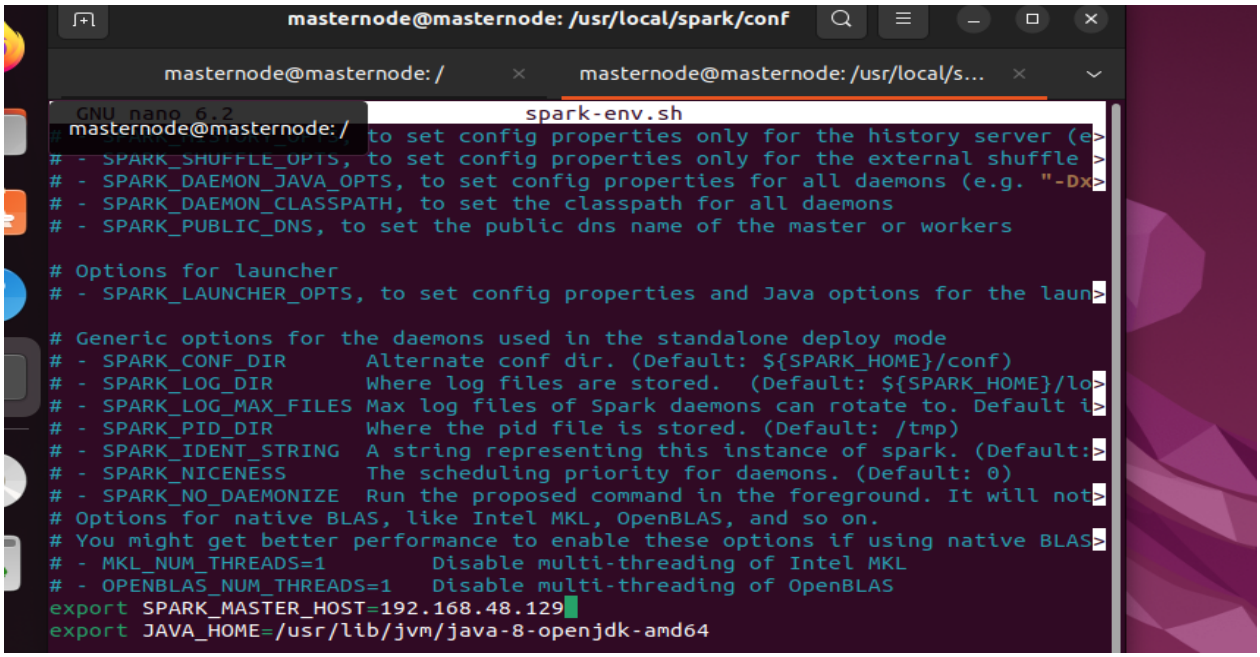


Figure 28 configuration de spark

Ajouter des travailleurs ou des esclaves

Maintenant, pour éditer le fichier de configuration usr/local/spark/con f /slaves, exécutez la commande suivante sur master :

- **sudo nano /usr/local/spark/conf/slaves**

Et ajouter le nom du maître et des travailleurs/esclaves (indiqué ci-dessous) dans le fichier mentionné ci-dessus, enregistrer et quitter.

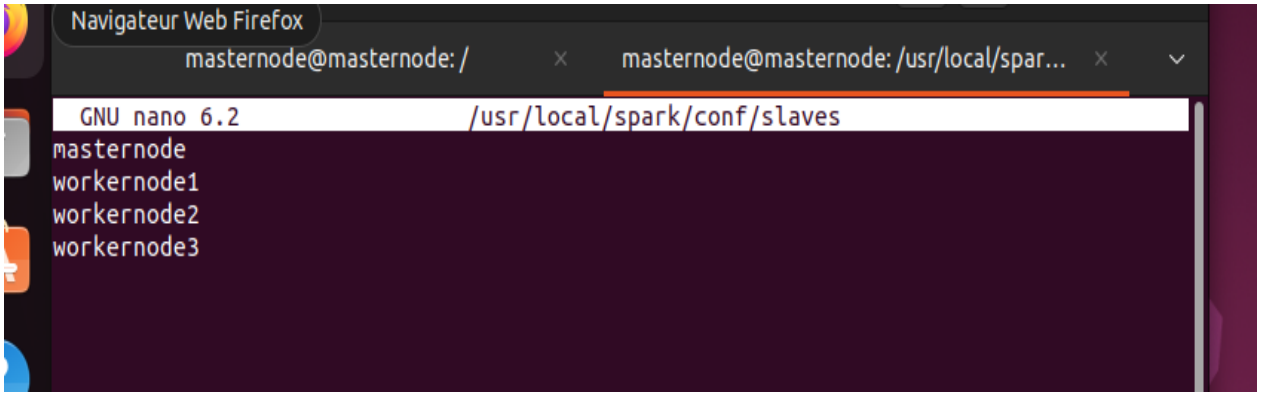


Figure 29 éditer le fichier de configuration spark

- étape de démarrer le cluster

Maintenant, notre cluster Apache Spark est prêt. Pour démarrer le cluster Apache Spark, exécuterons la commande suivante sur le maître :

- **cd /usr/local/spark**
- **./sbin/start-all.sh**

Maintenant, pour vérifier les services démarrés par spark, exécutez la commande suivante : **jps**

```
masternode@masternode:/usr/local/spark/conf$ jps
488083 Jps
318290 Worker
317932 Master
```

- étape de voir état de notre cluster

Pour voir l'état du cluster Apache Spark dans le navigateur (UI), accédons à notre navigateur et tapons : https://192.168.48.129:8080

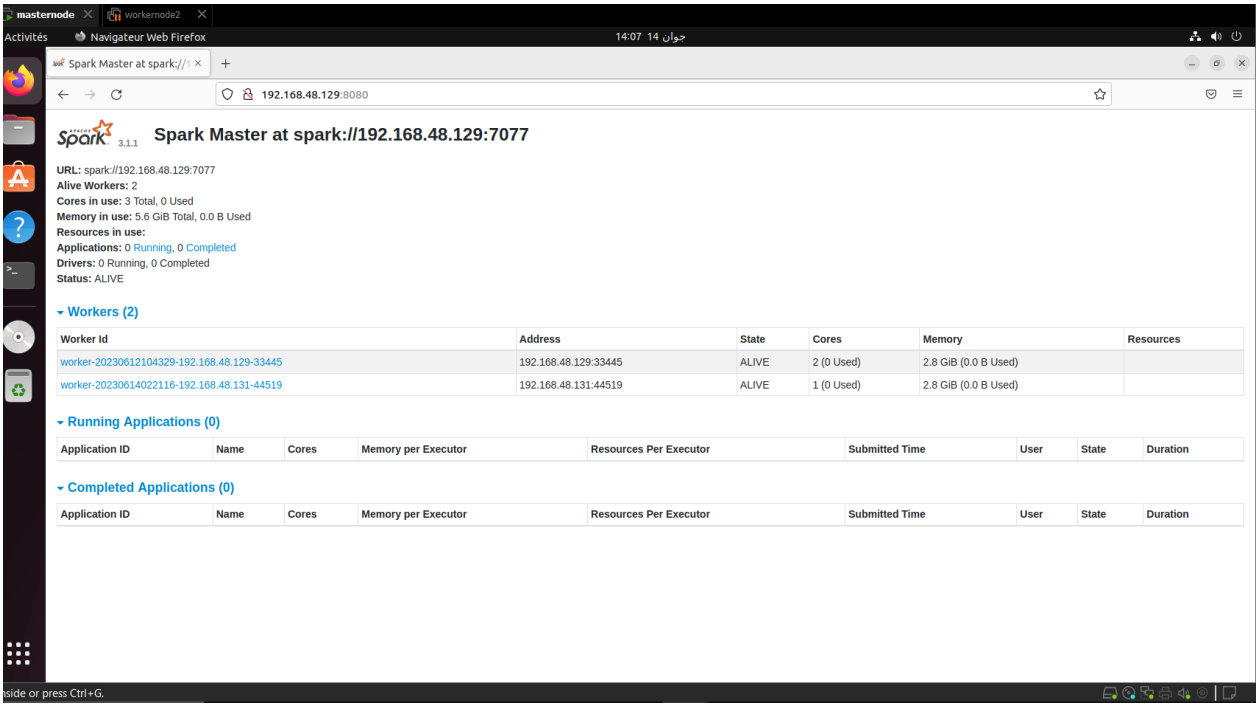


Figure 30 UI de Spark

Nous avons installé et configurer notre cluster avec succès sur Ubuntu 22.04. Nous pouvons maintenant commencer à utiliser cluster pour nos travails.

4.5 Importation du document RDF

Cette section comprend les étapes obligatoires pour la représentation de la document RDF en graphe alors en suivre les étapes suivantes

- Étape 1 : ouvrirons le browser neo4j (http://localhost:7687) et nous avons appellerons l'extension neosemantics a travers de l'exécution de la commande **call n10s.graphconfig.init()**

Figure 31 appeler extension neosemantics

- Étape 2 : importons nos document RDF qui s'appelle BSBM_250.nt avec leur emplacement a travers exécution de la commande suivants **call n10s.rdf.import.fetch("file:///home/data/Bureau/BSBM_250.nt,"Turtle")**
- Étape 3 : exécutons la commande suivante pour afficher le graphe de notre documents **MATCH (n:Resource) RETURN n LIMIT 300**

Le système représenter les données de fichier RDF en graphe et donner le résultat ci-dessous

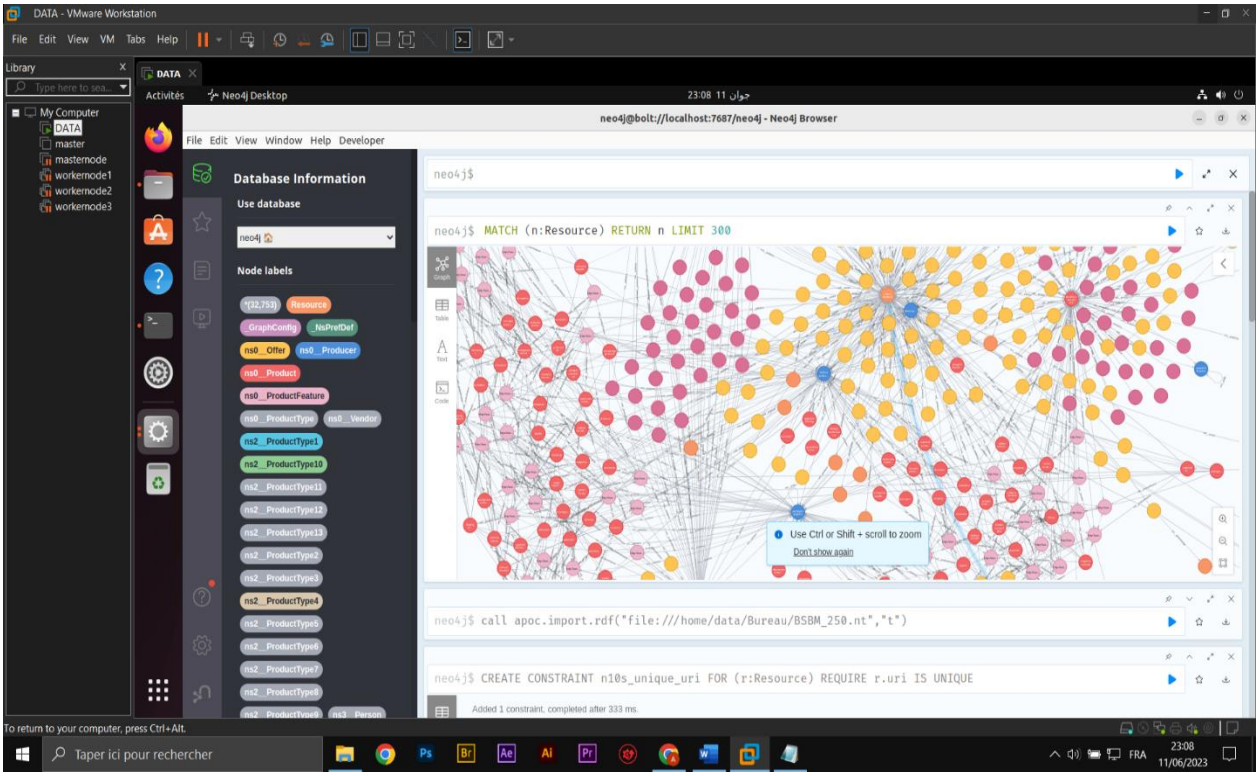


Figure 32 le graphe de fichier RDF de BSBM

Et puis nous partitionnerons le graphe et enverrons des partitions aux nœuds de calcul (workers/esclaves).

4.6 Création des requêtes SPARQL

La création des requêtes être dans le serveur de calcul (Master) de notre cluster) pour obtenir le résultat sur lequel nous travaillons

Nous avons essayé d'exécuter la requête ci-dessous pour obtenir des offres pour un produit donné qui répondent à des exigences spécifiques. Par exemple dans le cas de «le consommateur souhaite acheter auprès d'un fournisseur dans un pays spécifique qui est en mesure de livrer dans les 3 jours et recherche l'offre la moins chère qui répond à ces exigences »

```
 8
 9 PREFIX bsbm: <http://www4.wiwiss.fu-berlin.de/bizer/bsbm/v01/vocabulary/>
10 PREFIX dc: <http://purl.org/dc/elements/1.1/>
11
12 SELECT DISTINCT ?offer ?price
13 WHERE {
14         ?offer bsbm:product %ProductXYZ% .
15         ?offer bsbm:vendor ?vendor .
16     ?offer dc:publisher ?vendor .
17         ?vendor bsbm:country %CountryXYZ% .
18         ?offer bsbm:deliveryDays ?deliveryDays .
19         FILTER (?deliveryDays <= 3)
20         ?offer bsbm:price ?price .
21     ?offer bsbm:validTo ?date .
22     FILTER (?date > %currentDate% )
23 }
24 ORDER BY ?price
25 LIMIT 10
26
```

Figure 33 exécution de requête 1

Nous avons essayé d'exécuter une autre requête pour rechercher des produits dont l'étiquette contient des mots spécifiques. Par exemple le consommateur se souvient de certaines parties d'un nom de produit lors de recherches antérieures. Il veut retrouver le produit en recherchant les parties du nom dont il se souvient

```
11
12 PREFIX rdfs : <http://www.w3.org/2000/01/rdf-schema#>
13 PREFIX rdf : <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
14 PREFIX bsbm : <http://www4.wiwiss.fu-berlin.de/bizer/bsbm/v01/vocabulary/>
15
16 SELECT ?product ?label
17 WHERE {
18         ?product rdfs:label ?label .
19     ?produit rdf:type bsbm:Produit .
20         FILTER regex(?étiquette, "%mot1%|%mot2%|%mot3%")
21 }
22
```

Figure 34 exécution de requête 2

Nous avons essayé d'exécuter la troisième requête pour rechercher des produits pour un ensemble donné de fonctionnalités génériques. Par exemple « Un consommateur recherche un produit et a une idée générale de ce qu'il veut »

```
PREFIX bsbm-inst: <http://www4.wiwiss.fu-berlin.de/bizer/bsbm/v01/instances/>
PREFIX bsbm: <http://www4.wiwiss.fu-berlin.de/bizer/bsbm/v01/vocabulary/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

SELECT DISTINCT ?product ?label
WHERE {
    ?product rdfs:label ?label .
    ?product rdf:type %ProductType% .
    ?product bsbm:productFeature %ProductFeature1% .
    ?product bsbm:productFeature %ProductFeature2% .
    ?product bsbm:productPropertyNumeric1 ?value1 .
        FILTER (?value1 > %x%)
        }
ORDER BY ?label
LIMIT 10
```

Figure 35 exécution de requête 3

4.7 Type d'évaluation

Après le partitionnement de notre fichier RDF en trois, aussi notre graphe RDF en trois Alors nous allons obtenir quatre types de données différent (fichier RDF, fichier RDF partitionné,

graph RDF, graphe RDF partitionné) pour faire essai de notre travail, le tableau ci-dessous représente une description de type d'évaluation avec leur numéro

Tableau 4 type d'évaluation

Numéro de types dévaluation	Description
1	Lancer les requêtes sur un fichier RDF
2	Lancer les requêtes sur un fichier RDF partitionné
3	Lancer les requêtes sur un graphe RDF
4	Lancer les requêtes sur un graphe RDF partitionné

4.8 Résultat

Le tableau ci-dessous représente le temps d'exécution de la requête par rapport à le types d'évaluation nous avons vu que le temps d'exécution des données (fichier RDF et le graphe RDF) partitionnées beaucoup moins que les données n'est pas partitionnées, et le résultat de graphe partitionné c'est mieux que le fichier partitionné

Tableau 5 Temps d'exécution de la requête par rapport types d'évaluation

Numéro de la requête	Numéro de types dévaluation	Temps d'exécution de la requête total
1	1	34,28 s
	2	6,447 s
	3	26,991 s
	4	5,966 s
2	1	23,962 s
	2	6,047 s
	3	17,761 s
	4	5,582 s
3	1	30,575
	2	5,069
	3	25,987
	4	4,348

Après l'exécution des requêtes nous conclurons que la représentation graphique et le partitionnement des données ça réduire le temps d'exécution.

4.9 Conclusion

Dans ce chapitre nous avons présenté l'implémentation de notre modèle nous avons créé un server de stockage virtuel et convertis les données RDF en forme de graphe, ensuite nous avons créé un cluster de calcul virtuel qui contient un master et trois esclaves et nous avons appliqué la configuration réseau et la sécurité pour faire exécution de requête pour obtenir des résultats très rapides pour réduire le temp d'exécution.

Avantage de notre modèle c'est l'obtenir ou trouver une solution réseau pour faciliter l'opération de trouver des résultats des requêtes dans les environnements Big Data et pour distribuer les tâches de travail entre les hôtes (machines).

Le problème rencontré dans l'implémentation de notre travail ce que la machines sur lequel nous utilisé elle ne pouvait pas le supporter la charge il n'est pas tellement puissant qu'il install un serveur de stockage et un cluster de calcul et travailler aux mêmes temps. Ce travail nécessite des outils très puissante surtout le processeur doit contenir un grand nombre de cœurs.

Conclusion Générale

Le web sémantique, propose une nouvelle plateforme permettant une gestion plus intelligente du contenu, à travers sa capacité de manipuler les ressources sur la base de leurs sémantiques. Le web sémantique favorise ainsi les coopérations Homme/Machine et permet de s'ouvrir à de nouvelles possibilités d'automatisation sur le Web.

Le web sémantique offre des avantages majeurs en termes d'interopérabilité des données, de recherche d'informations améliorée, de représentation formelle des connaissances et de possibilités d'automatisation des tâches. Ces avantages favorisent une utilisation plus efficace et une exploitation plus riche des informations disponibles sur le web, ouvrant ainsi de nouvelles perspectives pour le développement de technologies et d'applications avancées.

Utilisation de systèmes de gestion de base de données NoSQL dans le contexte du web sémantique permet d'améliorer la gestion des données massives à l'échelle du Big Data. Ces systèmes offrent une extensibilité, une flexibilité de modélisation et des performances élevées, ce qui facilite le traitement et l'exploitation des informations sémantiques présentes sur le web.

Dans notre modèle nous avons proposé une solution réseau pour résoudre le problème de temps d'exécution élevé ,et nous avons utilisé le système de neo4j pour transformer les données RDF en forme de graphe nous avons aussi essayé de partitionner le graphe pour partager au plusieurs machines pour faciliter étape de exécution des requêtes. nous avons aussi utilisé Spark pour répartir les tâches de calcul sur plusieurs nœuds de traitement ça veut dire chaque nœud du cluster exécute des requêtes sur les graphe partitionner dans nous avons parlé plus tôt ce qui permet effectuer des calculs en parallèle et regrouper le résultat pour accélérer le traitement global

Apache Spark offre des avantages considérables en termes de vitesse de traitement élevée, de polyvalence, de facilité d'utilisation, d'évolutivité horizontale, de tolérance aux pannes et d'écosystème riche. Ces avantages en font un choix populaire pour le traitement des données à

grande échelle, offrant des performances optimales et une flexibilité pour une variété de scénarios de traitement des données

Dans les travaux futures nous pouvons faire d'appliquer notre modèle dans des machines réels a la place de l'utilisation des machines virtuel pour commencerons amélioration de notre modèle parmi eux de faire des partitionnement des requêtes afin d'être prêt à l'emploi et efficace.

Bibliographie

[1] T.Berners-Lee, The Semantic Web Roadmap, Septembre 1998

[2] La naissance du web https://home.cern/fr/science/computing/birth-web

[3] Mme O.BELLILET, cours de La sémantique, Université Constantine Mentouri

[4] WEB SÉMANTIQUE, https://www.w3.org/standards/semanticweb/

[5] T.Berners-Lee, J.Hendler, & O.Lassila, The Semantic Web. Scientific American,2001

[6] Ba-Huy Tran, Une approche sémantique pour l'exploitation de données environnementales,P33-34.

[7] Z.Allaoua , Développement De Services Web RESTFUL Et La Construction D'une Ontologie Dans Le Domaine De La Mécanique., mémoire fin d'étude

[8] Ora Lassila & R. Swick, W3C,Resource Description Framework(RDF) Model and Syntax Specification, le 22 février1999.

[9] A.Hassan A.Mrabet P.Darmon "Vers une modélisation orientée privacy des métadonnées d'un lac de données". Conférence : EDA 2022 : 18ème journées Business Intelligence & Big Data

[10] Z.Pan and I.Horrocks, RDFS(FA) and RDF MT: Two Semantics for RDFS: University of Manchester, p30-31.

[11] P.li,Semantic Reasoning on the Edge of Internet of Things,University of oulu,November 2016 .

[12] Tom Gruber, Ontologie:Encyclopedia of Database Systems,Springer- Verlag , 2008.

[13] Tom Gruber, Ontologie:Encyclopedia of Database Systems,Springer- Verlag , 2008.

[14] Arnaud.V, Modélisation ontologique des connaissances expertes pour l'analyse de comportements à risque: doctorat paristech, 2012

[15] Grigoris.A, Enrico.F, Frank.H, Introduction to Semantic Web Ontology Languages, University of Bozen–Bolzano, p7-8

[16] S.Harris, Garlik , A.Seaborne, W3C , Langage de requête SPARQL 1.1 , 21 mars 2013.

[17] M.Stocker, A.Seaborne, D.Reynolds and all "Optimisation de modèle de graphe de base SPARQL à l'aide d'une estimation de sélectivité". Conférence internationale sur le World Wide Web, WWW 2008, Pékin, Chine, 21-25 avril 2008

[18] Oracle, big data, https://www.oracle.com/fr/big-data/what-is-big-data/

[19] Dingeman knap, embedded, Making the most of growing streams of industrial data, https://www.embedded.com/making-the-most-of-growing-streams-of-industrial-data/ ,17 juillet 2020.

[20] Datascientest,NoSQL:Tout comprendre sur les bases de données non relationnelles, 02/juin/2021 https://datascientest.com/nosql

[21] A.sarah , Traitement Des Données Sémantique Basé Sur L'utilisation De L'outil Metis, mémoire fin d'étude

[22]J.Dean and S.Ghemawat, MapReduce: Simplified Data Processing on Large Clusters,google,Inc p 137

[23] Z.SAYAH,"Intégration de la Sémantique dans le Big Data",document de doctorat.

[24] M.hassan,"Interrogation de données sémantiques sur des bases de données NoSQL avec Apache Spark, Conférence IEEE 2018, P365

[25] M.Cossu, M.Farber, Georg.Lausen, article university ,"PRoST: Distributed Execution of SPARQL -eries Using Mixed Partitioning Strategies".

[26] X.Chen, H.Chen,N.Zhang, S.Zhang, 'SparkRDF : moteur de traitement de graphes RDF discret élastique avec distribution Mémoire', Département d'informatique Université du Zhejiang Hangzhou, Chine.

[27] M.Hassan, école d'informatique, Interrogation de données sémantiques sur des bases de données NoSQL avec Apache Spark, Conférence internationale IEEE 2018.

[28] M.Hassan, S.Bansal, Schéma de partitionnement des données pour une distribution efficace Interrogation RDF à l'aide d'Apache Spark, 2019 IEEE 13e ICSC.

[29] O.Erling & I.Mikhailov, Analyse comparative des systèmes de bases de données RDF : métriques et approches, Web sémantique, p 63-86, 2013

[30] La référence SPARQL de Berlin, https://www.semanticscholar.org .

[31] J.Webber & I.Robinson. Bases de données de graphes. O'Reilly Media, 2013.