Framework Apache Spark - MongoDB

Abdoulaye SERE [1] José Arthur OUEDRAOGO [2]

[1]Université Nazi BONI(UNB), UFR-ST

[2]Ecole Supérieure d'Informatique (ESI)

josearthur.oued@outlook.com

February 10, 2021

Overview

Présentation générale de Apache Spark

Naissance de Apache Spark

- Spark a vu le jour en 2009 en tant que projet au sein de l'AMPLab (Algorithms, Machines and People Laboratory) à l'Université de Californie à Berkeley. Et Spark a été écrit en scala.
- Apache Spark est un framework de traitements Big Data open source construit pour effectuer des analyses sophistiquées et conçu pour la rapidité et la facilité d'utilisation. C'est également Framework de conception et d'exécution Map/Reduce
- Logiciel libre de la fondation Apache(Licence Apache).

Présentation générale de Apache Spark

Langage et système de stockage

Les langages supportés actuellement pour le développement d'applications par Apache Spark sont :

- Java;
- Scala;
- Python;
- R.

Spark utilise le système de fichiers HDFS pour le stockage des données. Il peut fonctionner avec n'importe quelle source de données compatible avec Hadoop, dont HDFS, HBase, Cassandra, etc.

Ecosystème de Apache Spark (1/6)

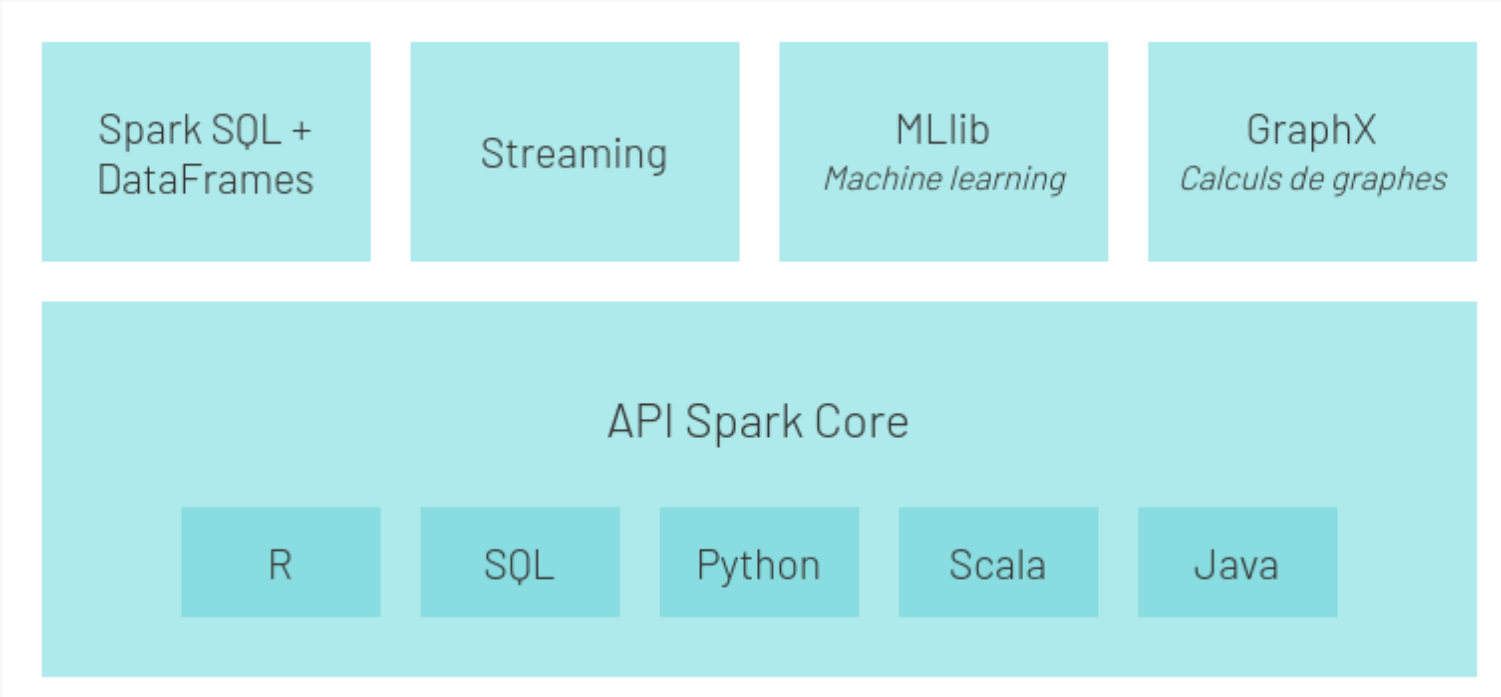

Figure: Ecosystème de Apache Spark

SparkSQL DataFrames (2/6)

Figure: SparkSQL DataFrames

Spark Streaming (3/6)

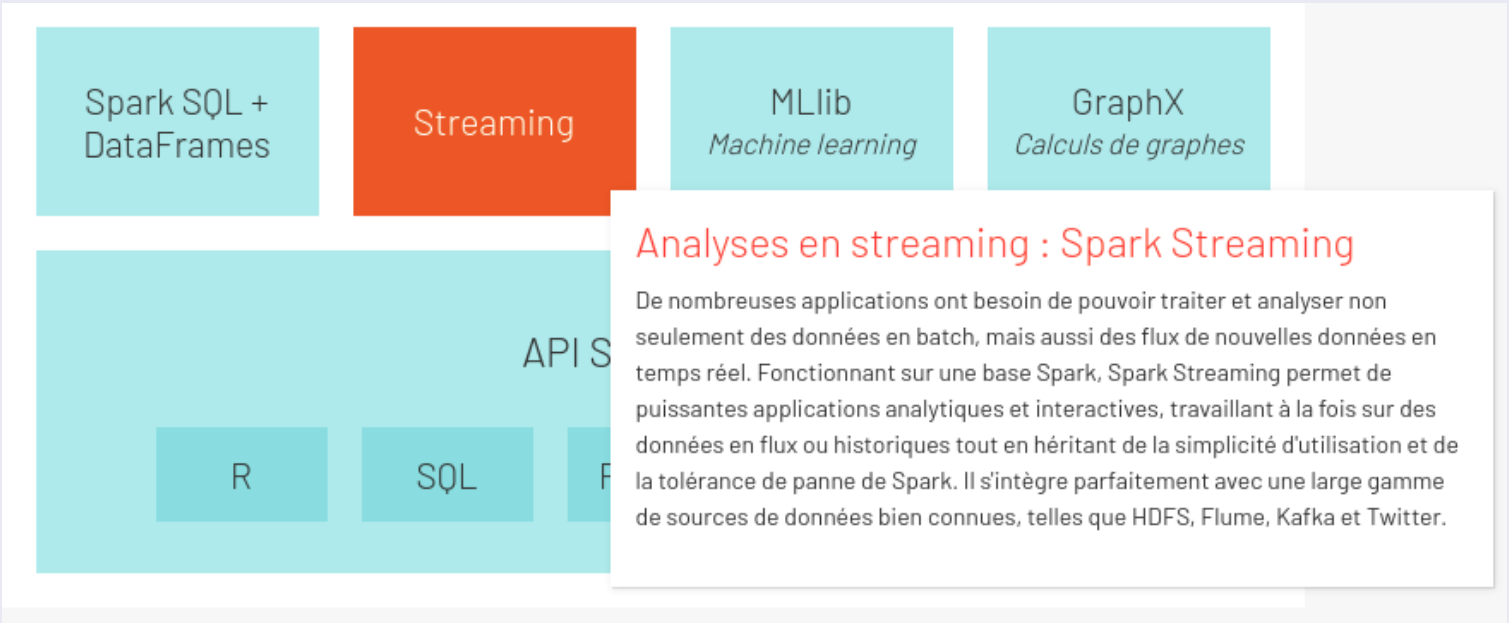

Figure: Spark Streaming

MLlib Machine Learning (4/6)

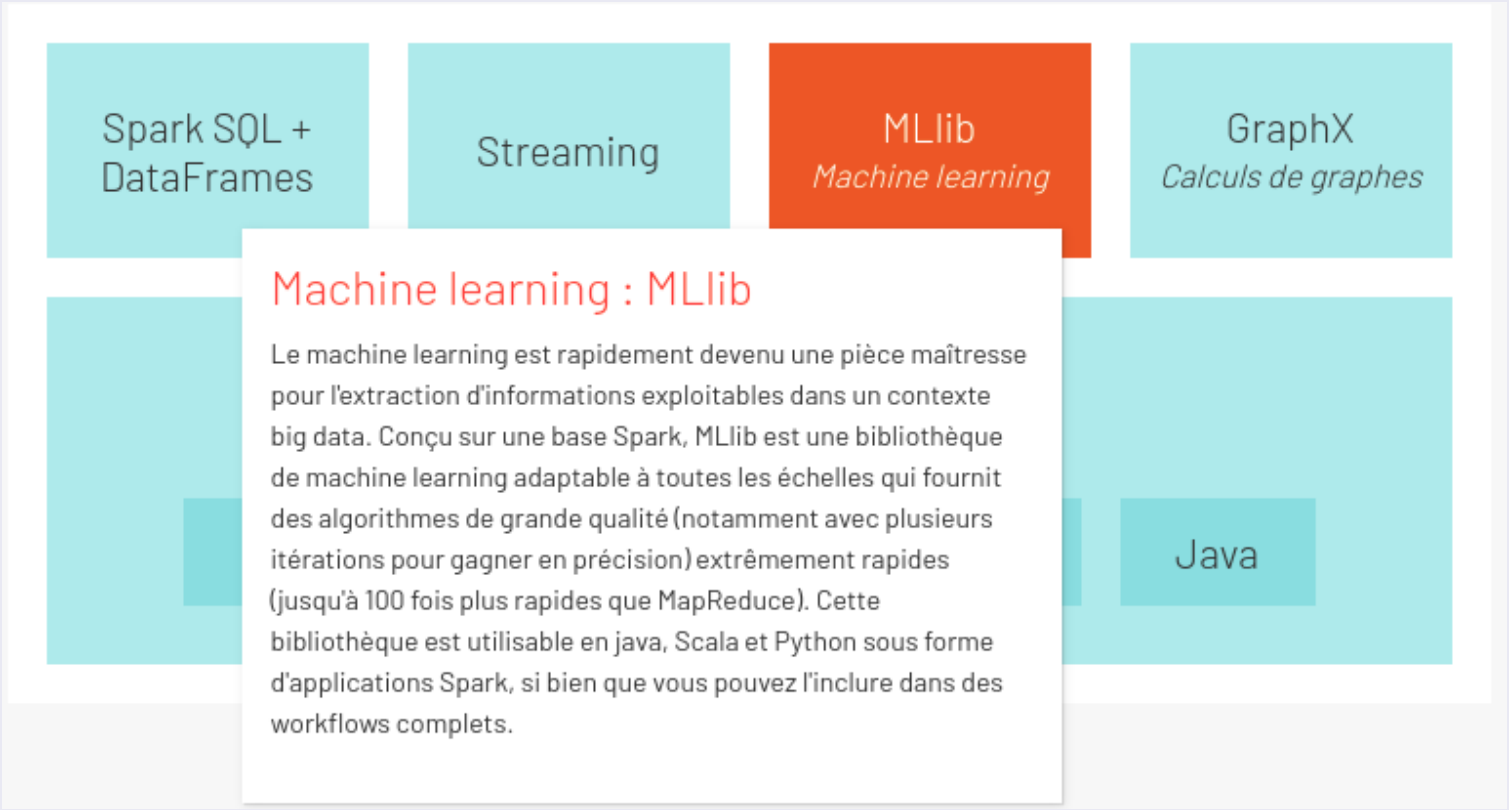

Figure: MLlib

GraphX (5/6)

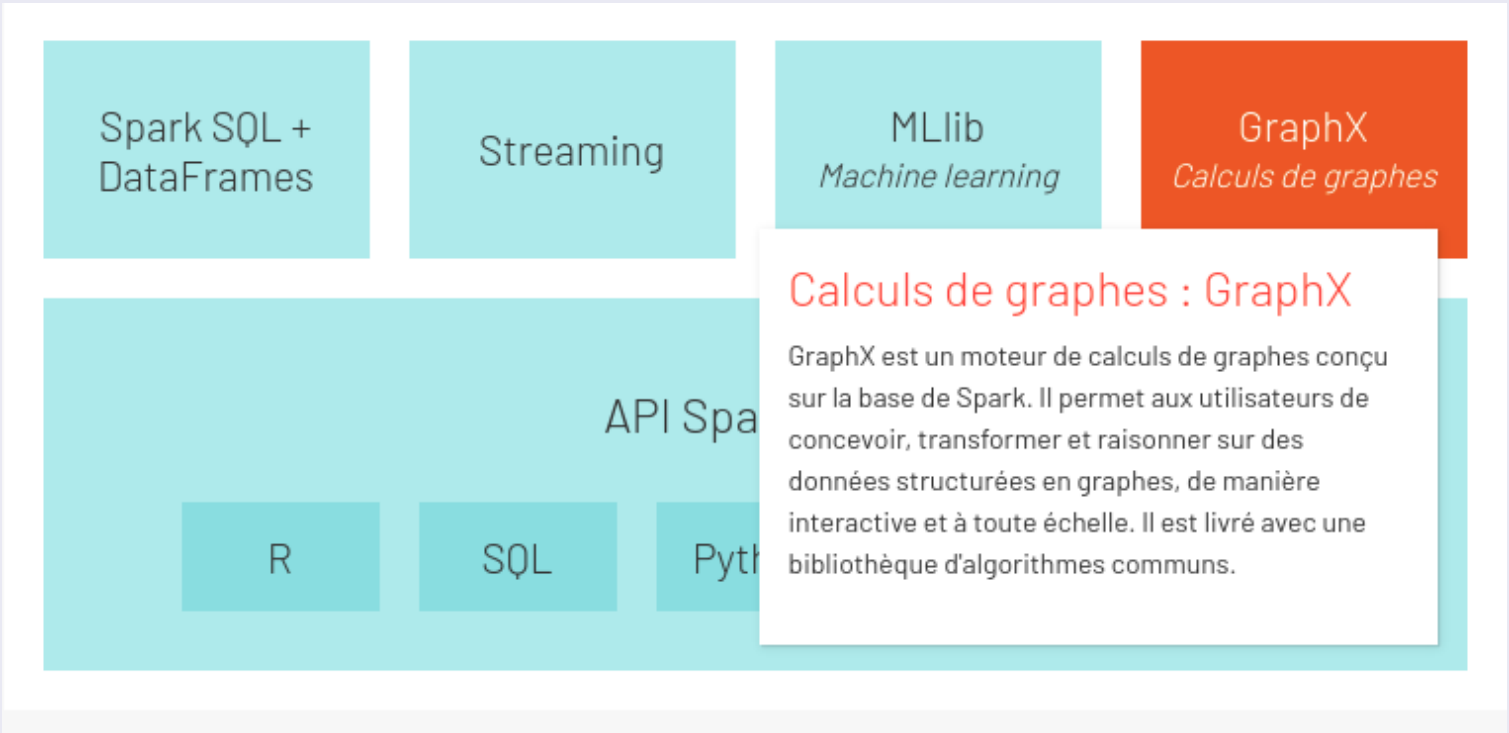

Figure: GraphX

API Spark Core (6/6)

Exécution générale : Spark Core

Spark Core est le moteur d'exécution sous-jacent de la plateforme Spark, sur lequel reposent toutes les autres fonctionnalités. Il apporte des capacités de calcul en mémoire pour plus de rapidité, un modèle d'exécution généralisé capable de prendre en charge une vaste gamme d'applications, et des API Python, Scala et java pour un développement facilité.

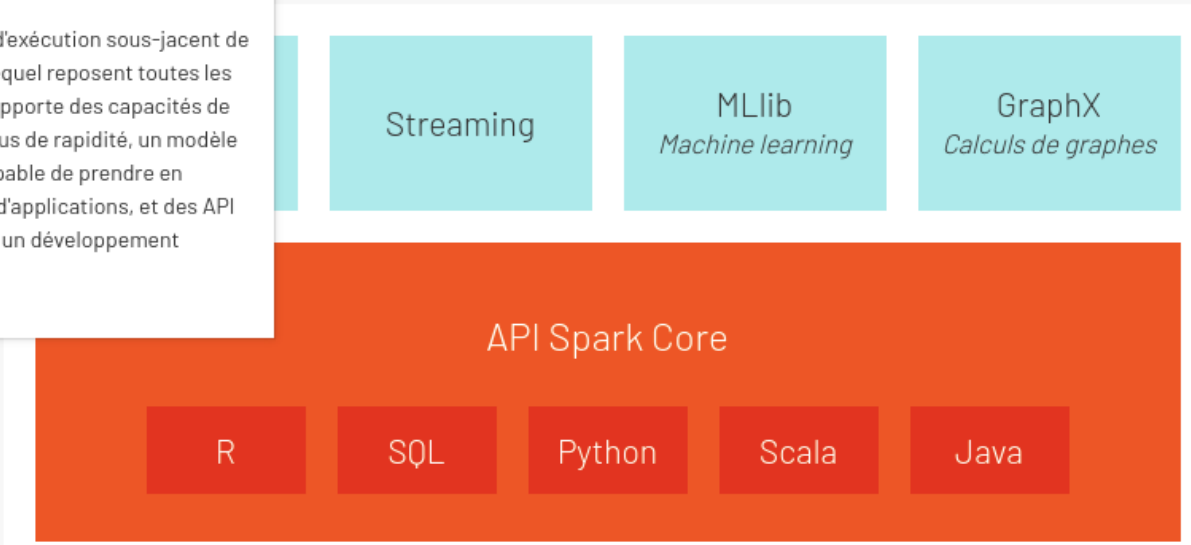

Figure: API Spark Core

Architecture - RDD (1/2)

Au centre du paradigme employé par Spark, on trouve la notion de **RDD**, pour Resilient Distributed Datasets.
Il s'agit de larges hashmaps stockées en mémoire et sur lesquelles on peut appliquer des traitements.
Ils sont:

- Distribués.
- Partitionnés (pour permettre à plusieurs noeuds de traiter les données).
- Redondés (limite le risque de perte de données).
- En lecture seule: un traitement appliqué à un RDD donne lieu à la création d'un nouveau RDD.

Architecture - RDD (2/2)

Deux types d'opérations possibles sur les RDDs:

- Une transformation: une opération qui modifie les données d'un RDD. Elle donne lieu à la création d'un nouveau RDD. Les transformations fonctionnent en mode d'évaluation lazy: elles ne sont exécutées que quand on a véritablement besoin d'accéder aux données. "map" est un exemple de transformation.
- Une action: elles accèdent aux données d'un RDD, et nécessitent donc son évaluation (toutes les transformations ayant donné lieu à la création de ce RDD sont exécutées l'une aprés l'autre). "saveAsTextFile" (qui permet de sauver le contenu d'un RDD) ou "count" (qui renvoie le nombre d'éléments dans un RDD) sont des exemples d'actions.

Avantages et inconvenients de Apache Spark

Avantages

- Performances supérieures à celles de Hadoop pour une large quantité de problèmes; et presque universellement au moins équivalentes pour le reste.
- API simple et bien documentée; très simple à utiliser. Paradigme plus souple qui permet un développement conceptuellement plus simple.
- Très intégrable avec d'autres solutions; peut très facilement lire des données depuis de nombreuses sources, et propose des couches d'interconnexion très faciles à utiliser pour le reste (API dédiée Spark Streaming, Spark SQL).
- APIs dédiées pour le traitement de problèmes en machine learning (Spark MLlib) et graphes (Spark GraphX).

Avantages et inconvenients de Apache Spark

Inconvénients

- Spark consomme beaucoup plus de mémoire vive que Hadoop, puisqu'il est susceptible de garder une multitude de RDDs en mémoire. Les serveurs nécessitent ainsi plus de RAM.
- Il est moins mature que Hadoop.
- Son cluster manager (Spark Master) est encore assez immature et laisse à désirer en terme de déploiement / haute disponibilité / fonctionnalités additionnelles du même type; dans les faits, il est souvent déployé via Yarn, et souvent sur un cluster Hadoop existant.

MapReduce

Definition

- MapReduce est un Framework de traitement de données en clusters. Composé des fonctions Map et Reduce, il permet de répartir les tâches de traitement de données entre différents ordinateurs, pour ensuite réduire les résultats en une seule synthèse.
- MapReduce est un modèle de programmation popularisé par Google. Il est principalement utilisé pour la manipulation et le traitement d'un nombre important de données au sein d'un cluster de nœuds. MapReduce consiste en deux fonctions map() et reduce().

MapReduce

Map

Dans l'étape Map le nœud analyse un problème, le découpe en sous-problèmes, et le délègue à d'autres nœuds (qui peuvent en faire de même récursivement). Les sous-problèmes sont ensuite traités par les différents nœuds à l'aide de la fonction Map qui à un couple (clé, valeur) associe un ensemble de nouveaux couples (clé, valeur) :
map(clé1,valeur1) → list(clé2,valeur2)

Reduce

Dans l'étape Reduce, où les nœuds les plus bas font remonter leurs résultats au nœud parent qui les avait sollicités. Celui-ci calcule un résultat partiel à l'aide de la fonction Reduce (réduction) qui associe toutes les valeurs correspondantes à la même clé à une unique paire (clé, valeur). Puis il remonte l'information à son tour. À la fin du processus, le nœud d'origine peut recomposer une réponse au problème qui lui avait été soumis : **reduce(key2,list(valeur2))→ valeur2**

Fonctionnement de MapReduce

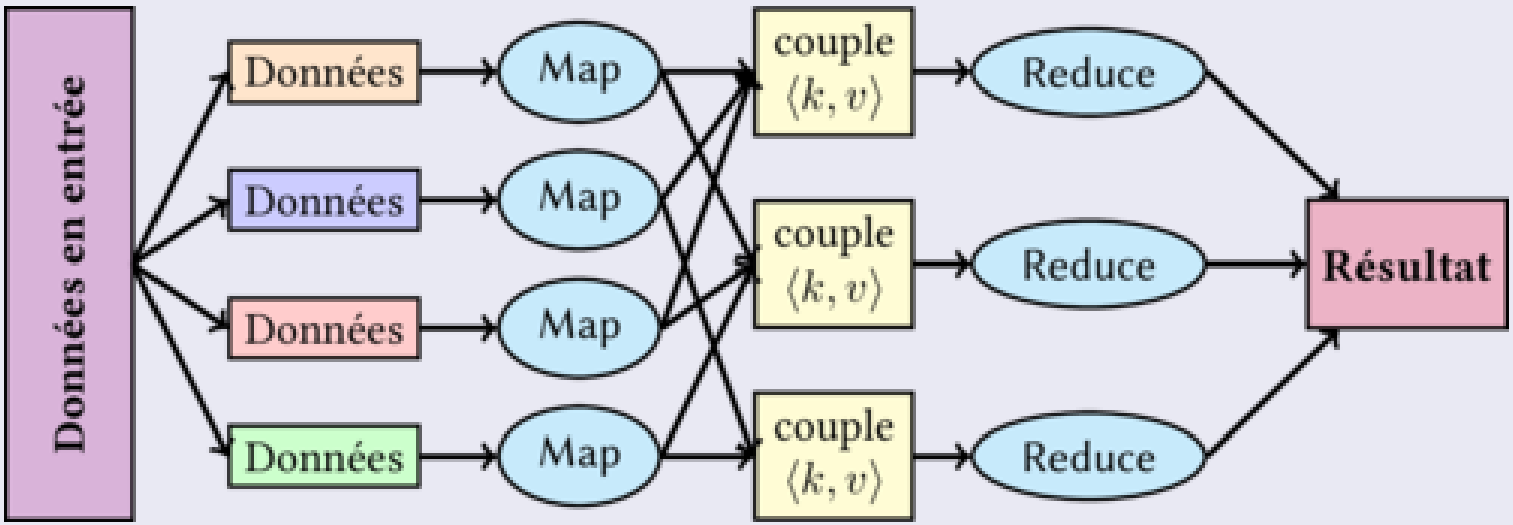

Figure: Schéma de fonctionnement du MapReduce

Apache Spark sous Windows ou sous Linux

Procédure d'installation sous Windows

- Consultez le site officiel de Apache Spark pour l'installer. *Apache Spark*
- *Installation de Apache Spark sous Windows*

Procédure d'installation sous Linux

- Consultez le site officiel de Apache Spark pour telecharger la version de votre choix. *Apache Spark*
- Extraire les fichier téléchargé **tar xzf Spark-3.0.1-bin-hadoop3.2.tgz**
- Déplacer le dossier dans le dossier où vous souhaiter installer Spark **mv Spark-3.0.1-hadoop3.2 /opt/**

Installer java 8 et python 3 si ce n'est pas encore fait Configuration de Spark pour exécution en mode cluster sur la machine master et sur les machines slaves(les workers)

NoSQL

Objectifs du NoSQL

Le modèle NoSQL qui signifie Not Only SQL a vu le jour en 2009, source. Le NoSQL ne signifie pas No SQL, bien au contraire il utilise même parfois le langage SQL, il est alors un complément au SGBDR pour des besoins spéciques mais pas une solution de remplacement. Son objectif premier est le stockage des données volumineuses qui était une entrave pour des entreprises comme Google, Facebook et Amazon.

Avantages du NoSQL

La force du NoSQL repose sur les multiples avantages suivants :

- Il est évolutif car il utilise la scalabilité horizontale dont le principe est d'augmenter les serveurs en parallèles pour repartir les charges, cela augmente aussi la tolérance aux pannes ;
- Il est flexible car ces applications peuvent stocker les données de divers formats ou structures ;
- Il est très économique si bien que le coût des serveurs est moins élevé par rapport à ceux du relationnel et les solutions NoSQL sont majoritairement libres ;
- Il est facile à déployer et constitue un appui pour le cloud computing.

Differents modèles de Base de données NoSQL

Bases de données NoSQL (1/2)

- **Le Modèle clé-valeur(key-value) :** Dans ces types de bases de données, les données sont représentées par un couple clé/valeur. Comme exemples d'implémentation de ce modèle nous avons: Voldemort, Redis et Riak.
- **Le Modèle Colonne (Column) :** Pour ce modèle, le stockage des données se fait par colonne. Des exemples d'implémentations sont : HBase, Cassandra, Hypertable, Spark SQL, Elasticsearch.

Differents modèles de Base de données NoSQL

Bases de données NoSQL (2/2)

- **Le Modèle Document :** Ce modèle stocke une collection de "documents" dont un document est constitué de champs et des valeurs associées. Des exemples d'implémentations de ce modele sont : MongoDB, CouchDB, Couchbase.
- **Le Modèle Graphe :** Le stockage des données se fait en utilisant la théorie mathématique des graphes. Comme exemples d'implémentations : Neo4j, OrientDB, Oracle Graph.

MongoDB

Présentation de MongoDB

- Mongodb est un SGBD orienté documents, et non relationnel.
- Le document est l'unité de base : il peut être inséré, supprimé, recherché... On peut manipuler simultanément des types de documents différents sans mettre en place la moindre contrainte entre ceux-ci. Les documents sont contenus dans des collections réunies en bases de données.
- Grossièrement, on peut faire un parallèle entre un enregistrement et un document, une table et une collection.

Comparaison entre SQL et MongoDB

SQL	MongoDB
database	database
table	collection
enregistrement	document
colonne	champ
index	index
jointure	objet, dénormalisation
clef primaire	clef primaire (Dans MongoDB la clef primaire est automatiquement attribué au champ _id)
clef étrangère	reférence

Figure: SQL vs MongoDB

Installation de MongoDB

La mise en place de MongoDB est simple et rapide. Les packages disponibles au téléchargement contiennent à la fois un serveur et un client prêts à l'emploi.
Veuillez donc télécharger la dernière version de MongoDB adaptée à votre système d'exploitation.
Consultez le lien officiel de MongoDB : MongoDB

References

- Apache Spark, (consulté le feb. 08, 2021)
- Tout savoir du big data, (consulté le feb. 08, 2021)
- Avantages et inconvenient de Spark, (consulté le feb. 08, 2021)
- Comprendre les RDD pour mieux Développer en Spark, (consulté le feb. 08, 2021)
- Documentation de MongoDB, (consulté le feb. 09, 2021)
- MongoDB CRUD, (consulté le feb. 09, 2021)

The End : A nos machines !!!