BASES DE DONNÉES NOSQL

Préparé par :BEN ARBIA Mohamed Taher

2020-2021

Plan

- Introduction: l'évolution de NoSQL
- Définition de NoSQL
- Les Caractéristiques NoSQL
- Types de bases de données NoSQL
- Théorème CAP
- Théorèmes ACID et BASE
- Avantages et désavantages des bases de données NoSQL
- NoSQL vs. Les bases de données relationnelles
- Conclusion

Introduction

Le problème de l'extensibilité de SQL a été reconnu par les entreprises Web 2.0, qui ont des besoins importants en matière de données et d'infrastructure, tels que Google, Amazon et Facebook. Seuls, ils devaient trouver leurs propres solutions à ce problème, avec des technologies telles que BigTable, DynamoDB et Cassandra.

Cet intérêt croissant a donné lieu à une série de systèmes de gestion de base de données NoSQL (SGBD), axés sur la performance, la fiabilité et la cohérence. Plusieurs structures d'indexation existantes ont été utilisées et améliorées dans le but d'améliorer la recherche et la performance de lecture.

Premièrement, il existait des types de bases de données NoSQL (propriétaires), développées par des grandes entreprises pour répondre à leurs besoins spécifiques, telles que BigTable de Google, censé être le premier système NoSQL, et DynamoDB d'Amazon.

Le succès de ces systèmes brevetés a initié le développement de plusieurs systèmes de bases de données de source ouverte et appartenant à des propriétaires similaires étant les plus populaires Hypertable, Cassandra, MongoDB, DynamoDB, HBase et Redis.

Définition de NoSQL

Qu’est-ce que NoSQL?

NoSQL = Not Only SQL

«NoSQL» ;

Le terme NoSQL (de l'anglais Not only SQL) est apparu en 1989. C'est à cette année-là que Carlo Stozzi le prononça pour la première fois en public ; c'était lors de la présentation de son système de gestion de base des données relationnelles open source. Il l'a appelé ainsi à cause de l'absence de l'interface SQL pour communiquer.

Plus tard en 2009, le mot réapparait lorsqu'Eric Evans l'utilisa pour spécifier le nombre grandissant des bases de données distribuées open source.

«NoSQL» ;

NoSQL désigne une catégorie de systèmes de gestion de base de données (SGBD) qui n'est plus fondée sur l'architecture classique des bases relationnelles. L'unité logique n'y est plus la table, et les données ne sont en général pas manipulées avec le langage d'interrogation qu'est le SQL.

Selon Shashank Tiwari dans son livre Professional NoSQL, « les auteurs de ce néologisme ont probablement voulu signifier non-relationnel, mais ont préféré le mot NoSQL parce qu'il sonne mieux » que, par exemple, NoREL (pour Not only Relational). En bref, le mot est aujourd'hui utilisé pour exprimer tous les SGBD et les logiciels de stockage de données qui ne sont ni relationnels, ni objets, ni hiérarchiques et ni réseaux.

«NoSQL» ;

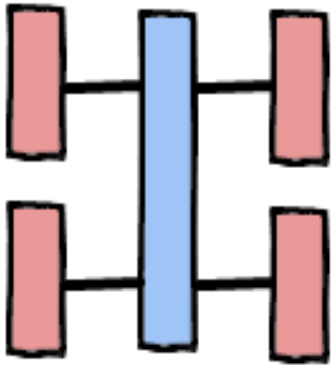
Constitue une alternative aux bases de données relationnelles dans lesquelles les données sont placées dans des tables et dont le schéma de données est soigneusement conçu avant la construction du base de données.

Les bases de données NoSQL sont particulièrement utiles pour travailler avec de grands ensembles de données distribuées..

Base de données SQL

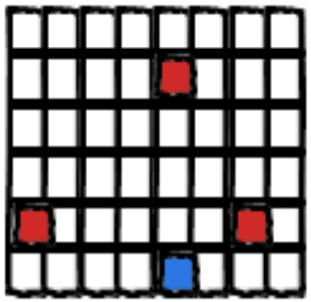
Relationnelle

OLAP

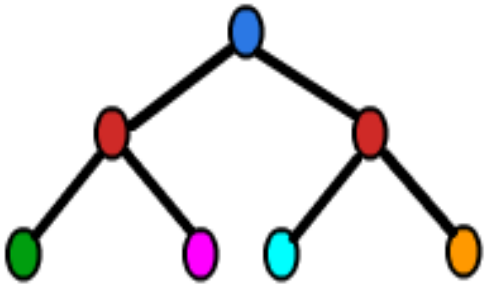


Base de données NoSQL

Orienté colonne

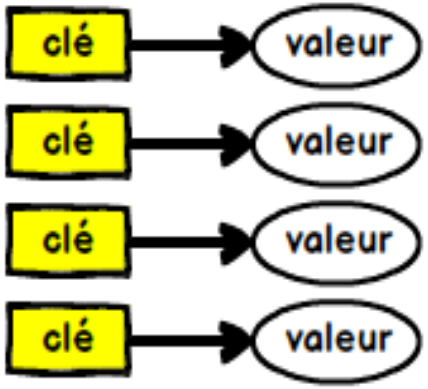


Document

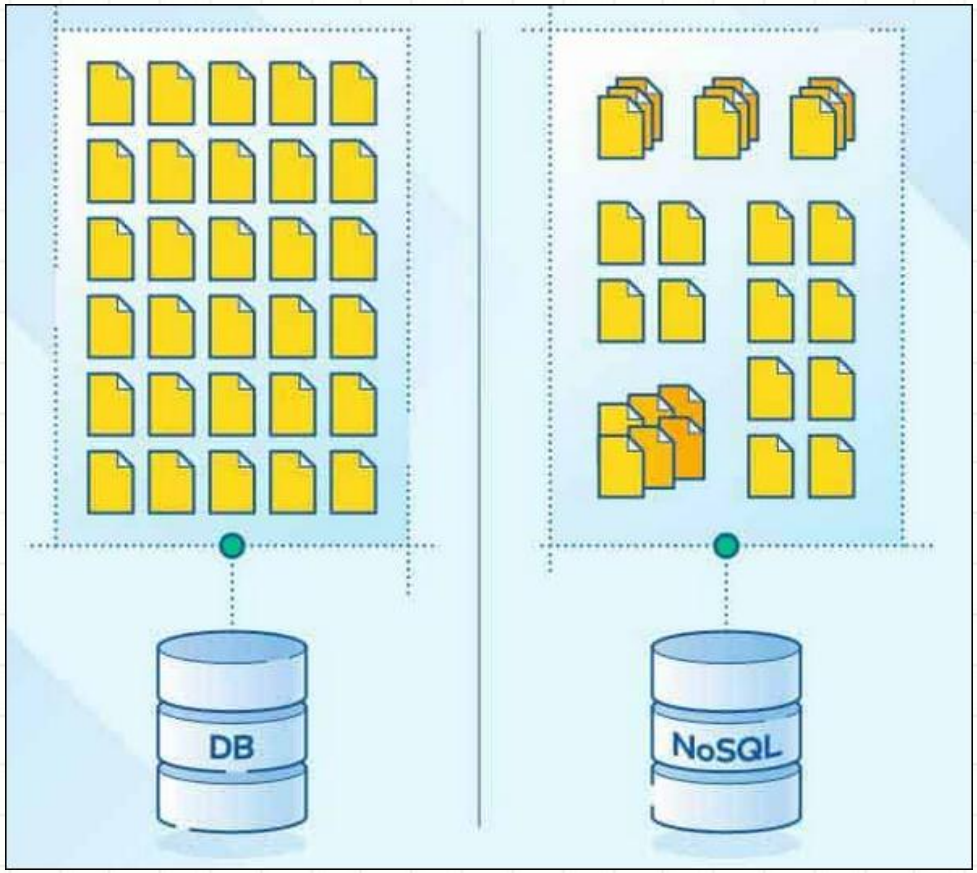


Graphe

Clé-Valeur



Une différence essentielle entre les bases de données NoSQL et les bases de données relationnelles traditionnelles réside dans le fait que NoSQL est une forme de stockage non structuré.



Cela signifie que NoSQL n'a pas de structure de table fixe comme celle trouvée dans les bases de données relationnelles.

Les Caractéristiques NoSQL

Les Caractéristiques NoSQL

- Structure de données proches des utilisateurs, développeurs
- Sérialisation
- Tables de hachage
- Parallélisation des traitements
- Réplication
- Stockage réparti
- Images
- Documents
- Données structurées et non structurées
- Interfaces depuis les langages classiques
- Protocoles d'accès aux données
- Priorité au traitement du côté client
- JSON

Types de bases de données NoSQL

Types de bases de données NoSQL

Bases de données de documents

Ces bases de données associent généralement chaque clé à une structure de données complexe qui s'appelle un document. Les documents peuvent contenir des paires de tableaux de clés ou des paires clé-valeur ou même des documents imbriqués.

Magasins de valeurs clés

Chaque élément est stocké sous la forme d'une paire de valeurs clés. Les magasins de valeur clé sont les plus simples parmi les bases de données NoSQL.

Mémoires à colonnes larges

Ces types de bases de données sont optimisés pour les requêtes sur de grands ensembles de données, et au lieu de lignes, ils stockent des colonnes de données ensemble.

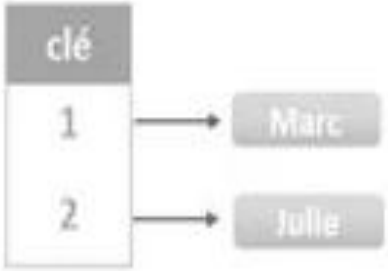
Base de donnée orientée graph

Ces DB stockent des informations sur les graphiques, les réseaux, tels que les connexions sociales.

Les bases NoSQL

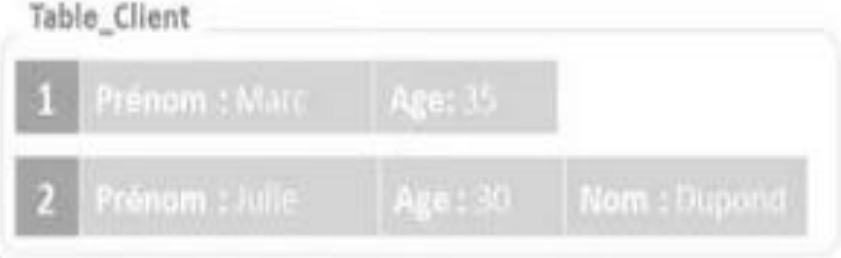
Clé/Valeur

ne permettent de stocker que des couples [clé, valeur].Cette valeur peut être une simple chaine de caractère comme un document, ou encore un objet beaucoup plus complexe.



Orientées Colonnes

sont très proches des SGBDR, on y retrouve le principe de « table », mais elles présentent deux grosses différences : les colonnes sont dynamiques et l'historisation de la données se fait à la valeur et non à la ligne.



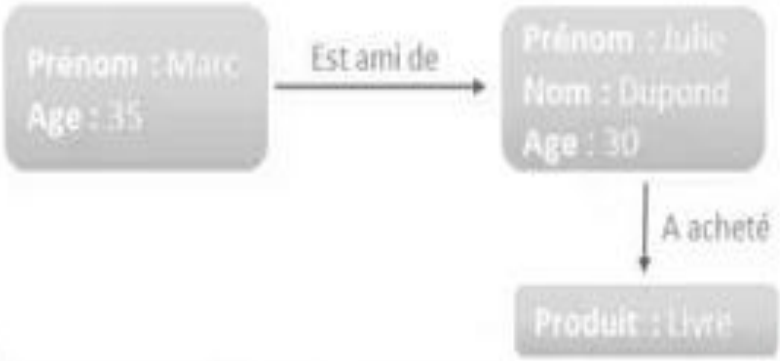
NoSQL

Orientées Documents

reposent également sur le paradigme [clé, valeur], mais la valeur, dans ce cas, est un document (JSON ou XML) sur lequel il est possible requêter.

Orientées Graphes

stockent les données en se basant sur la théorie des graphes. Elles s'appuient sur la notion de nœuds et de relations.



Théorème CAP

Théorème CAP

- «Une théorie soumise par Brewer en 2000. Elle a été reprise en 2003 par Seth Gilbert et Nancy Lynch du MIT qui l'ont redéfinie et elle prit le nom de théorème CAP. Cette dernière théorie stipule que dans le cadre d'un système distribué, plus spécifiquement dans l'utilisation d'une application web, une base de données ne peut pas garantir ces trois attributs en même temps. »

- Le théorème énonce donc qu'il est impossible pour un système distribué de fournir les trois propriétés suivantes à la fois :

- Consistency
- Availability
- Partition Tolerence

- ✓ **Consistency**: Tous les noeuds du système **voient les mêmesdonnées** au même moment quelques soient les modifications ;
- ✓ **Availability**: Les requêtes d'écriture et de lecture sont toujours satisfaites, donc il y a **disponibilité** pour la lecture et l'écriture ;
- ✓ **Partition tolerance**: La seule raison qui pousse un système à l'arrêt est la coupure totale du réseau. Autrement dit si une partie du réseau n'est pas opérationnelle, cela n'empêche pas le système de répondre. Le système **tolère** même une **partie** du réseau.

Théorèmes ACID et BASE

Principes ACID et BASE

- **Hormis le théorème CAP ci-haut énoncé, nous pouvons aussi parler de deux principes qui sont alors liés à la répartition des données et qui sont à la base des architectures actuelles des systèmes de gestion de bases de données, notamment les systèmes du type NoSQL.**

- **ACID et BASE représentent deux principes de conception aux extrémités opposées du spectre cohérence-disponibilité.**

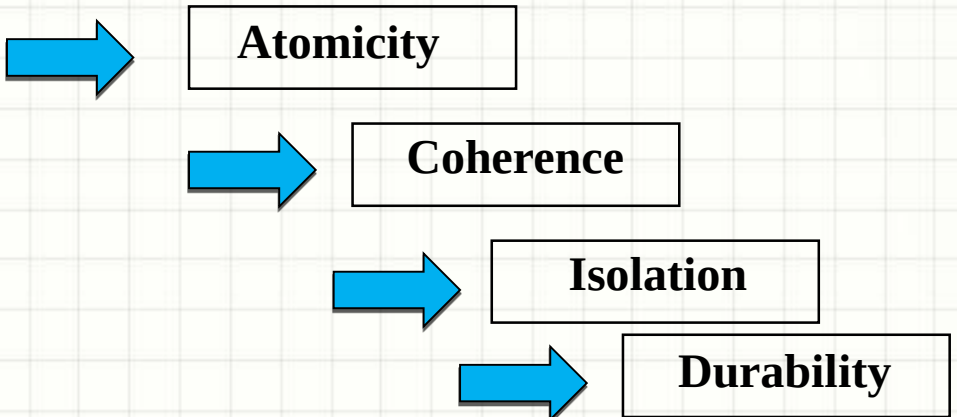
  Les propriétés ACID se concentrent sur la cohérence et sont l'approche traditionnelle des bases de données.

  Le principe BASE était créé à la fin des années 90 pour saisir les concepts émergents de la haute disponibilité et rendre explicite à la fois le choix et le spectre. Les systèmes étendus modernes et à grande échelle, y compris le Cloud, utilisent une combinaison des deux approches.

- **Bien que les deux acronymes soient plus mnémoniques que précis, l'acronyme BASE (étant le second apparu) est un peu plus délicat : Basically Available, Soft state, Eventually consistent (Simplement disponible, état souple, finalement consistant). Soft state et Eventual consistency sont des techniques qui fonctionnent bien en présence de partitions réseau et donc améliorent la disponibilité.**

- **La relation entre CAP et ACID est plus complexe et souvent incomprise, en partie parce que les C et A d'ACID représentent des concepts différents des mêmes lettres dans CAP et en partie parce que choisir la disponibilité affecte seulement certaines des garanties ACID**

- **Les quatre propriétés ACID sont :**



- ✓ **Atomicity** : Tout système bénéficie d'opérations atomiques. Quand l'objectif est la disponibilité, toutes les parties de la partition doivent toujours utiliser des opérations atomiques. De plus, des opérations atomiques de plus haut niveau (celles qu'impliquent ACID) simplifient la restauration ;
- ✓ **Coherence**: Dans ACID, le C signifie qu'une transaction préserve toutes les règles des bases de données, telles que les clés uniques. En comparaison, le C de CAP ne se réfère qu'à une cohérence de copie unique, un sous-ensemble strict de la cohérence ACID. Plus généralement, maintenir des invariants durant les partitions peut être impossible, d'où le besoin de bien choisir quelles opérations interdire et comment ensuite restaurer les invariants ;
- ✓ **Isolation** : L'isolation est au coeur du théorème CAP : si un système nécessite l'isolation ACID, il peut opérer sur au plus une partie durant une partition réseau. La possibilité de sérialiser les transactions nécessite des communications en général et échoue durant les séparations réseau. Des définitions plus faibles de l'exactitude sont viables durant les partitions en compensant durant la phase de restauration ;

Avantages et désavantages des bases de données NoSQL

Les bases de données NoSQL présentent des nombreux avantages par rapport aux bases de données traditionnelles:

- Contrairement aux bases de données relationnelles, les bases de données NoSQL sont basées sur des paires clé-valeur
- Certains types de stockage de bases de données NoSQL incluent différents types de stockages, tels que les stockages de colonnes, de documents, de valeurs de clé, de graphiques, d'objets, de XML et d'autres modes d'entrepôt de données.
- Lorsque vous travaillez avec des bases de données NoSQL, qu'elles soient de source ouverte ou qu'elles soient propriétaires, l'extension est plus simple et moins coûteuse que travailler avec des bases de données relationnelles. La raison c'est qu'une processus d'extensibilité horizontale est effectuée et la charge est répartie sur tous les nœuds. Au lieu d'une extensibilité verticale, plus courant dans les systèmes de bases de données relationnelles.
- On pourrait dire que l'implémentation de bases de données NoSQL de source ouverte est rentable. Puisqu'ils n'ont pas besoin de frais de licence et peuvent fonctionner sur du matériel économique.
- Quelques types de stockage de bases de données NoSQL incluent les entrepôts de colonnes, de documents, de valeurs de clé, de graphiques, d'objets, de XML et d'autres types d'entrepôt de données.

Bien entendu, les bases de données NoSQL ne sont pas parfaites et ne constitueront pas toujours le choix idéal.

La plupart des bases de données NoSQL ne prennent pas en charge les fonctions de fiabilité, qui sont soutenues par les systèmes de bases de données relationnelles. Ces caractéristiques de fiabilité peuvent être résumées comme suit: « atomicité, consistance, isolement et durabilité. » Cela signifie également que les bases de données NoSQL, qui ne supportent ces fonctionnalités, offrent une certaine cohérence en termes de performance et d'extensibilité.

L'incompatibilité avec les requêtes SQL est l'une des complexités trouvées dans la plupart des bases de données NoSQL. Cela signifie qu'un langage de requête manuelle est nécessaire, ce qui rend les processus beaucoup plus lents et complexes.

Afin de soutenir les fonctionnalités de fiabilité et de cohérence, les développeurs doivent implémenter leur propre code, ce qui ajoute une complexité supplémentaire au système.

Cela pourrait limiter le nombre d'applications sur lesquelles nous pouvons compter pour effectuer des transactions sécurisées et fiables, telles que des systèmes bancaires.

NoSQL vs. Les bases de données relationnelles

Ce tableau offre une brève comparaison entre les fonctionnalités de NoSQL et celles des bases de données relationnelles :

Feature	NoSQL Databases	Relational Databases
Performance	High	Low
Reliability	Poor	Good
Availability	Good	Good
Consistency	Poor	Good
Data Storage	Optimized for huge data	Medium sized to large
Scalability	High	High (but more expensive)

Conclusions

Conclusions

NoSQL commercialise des fonctions de fiabilité et de cohérence pour un processus de performance et d'extensibilité extrême. Cela en fait une solution spécialisée, car le nombre d'applications pouvant dépendre des bases de données NoSQL est encore limité. Ainsi, bien que la spécialisation puisse être peu flexible, si vous voulez un travail spécialisé, rapide et efficace, le plus indiqué sera NoSQL.

Merci