Cours bases de donneés documentaires et distribuées (NoSQL)

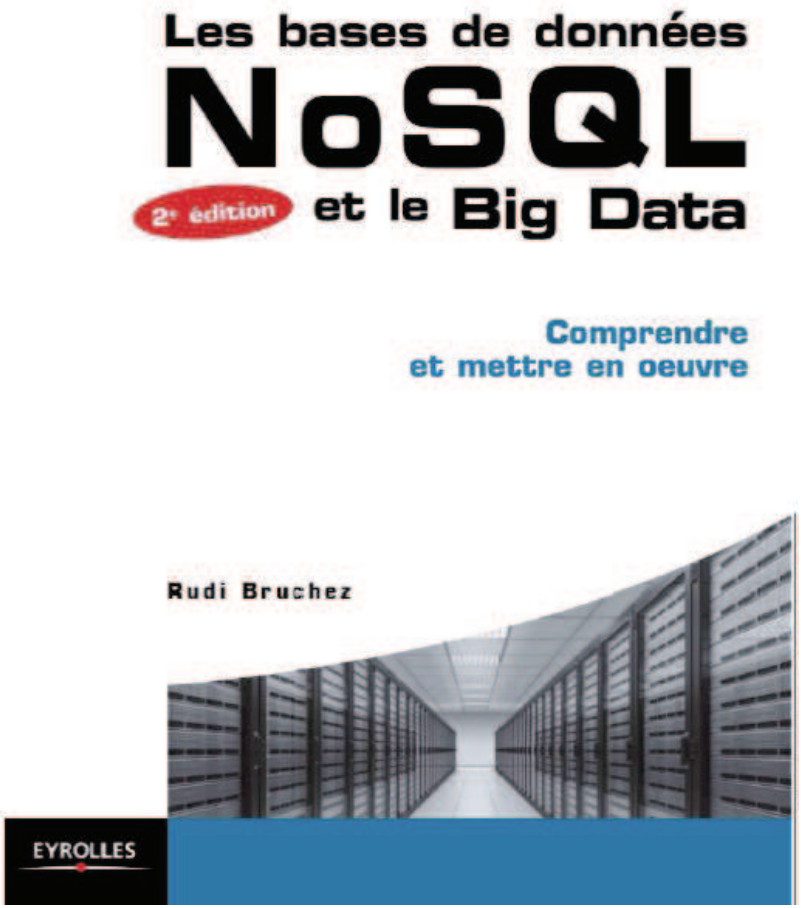
par Dr. Mouna Ben Ishak

Mastère Professionnel -M1- Big data in e-commerce

IHEC Carthage

2020

Références de base



Plan

Chapitre I : Introduction aux bases de données NoSQL

Chapitre II : Les principales bases de données NoSQL

Chapitre III : Les bases de données documentaires
(Cas pratique MongoDB)

Chapitre I

Introduction aux bases de données NoSQL

I.1. Histoire des systèmes de gestion de bases de données

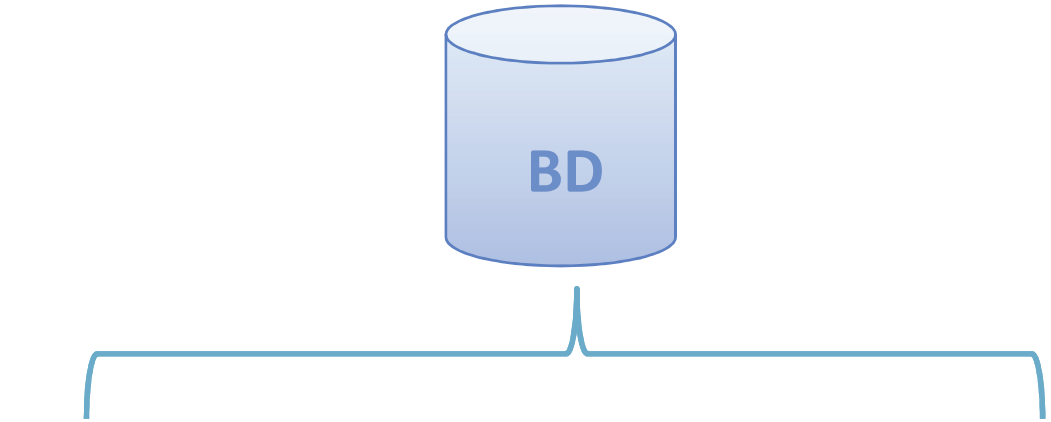
I.2. Des SGBDR au NoSQL

I.3. Les principales typologies de bases de données NoSQL

I.4. Questions récapitulatives

Qu'est ce qu'une base de données ?

Une base de données est un ensemble de données rassemblées et stockées d'une manière organisée afin d'en faciliter l'exploitation.



Stocker

- Maintenir les données sur des mémoires de masse

Exploiter

- Interroger
- Mettre à jour

Qu'est ce qu'une base de données ?

SGBD

La gestion et l'accès à une base de données sont assurés par un ensemble de programmes qui constituent le système de gestion de base de données (SGBD).

- Un SGBD est caractérisé par le modèle de description des données qu'il supporte.
- Les données peuvent être manipulées soit par un langage spécifique de manipulation des données soit par des langages de programmation.
  - Le langage de manipulation de données permet de retrouver et de manipuler des données.
  - Ce langage doit permettre l'obtention des réponses aux requêtes en un temps raisonnable.

Qu'est ce qu'une base de données ?

Schéma de données

- Le schéma ou modèle de données illustre la structure logique d'une base de données.
- Il renseigne sur les caractéristiques de chaque type de donnée et spécifie les relations et les contraintes qui déterminent comment les données peuvent être stockées et manipulées.
- Il existe plusieurs types de modèles de données (hiérarchique, réseau, relationnel, objet, graphe, etc.).

Le modèle hiérarchique (les années 1950)

- Le modèle hiérarchique est historiquement la première forme de modélisation de données sur un système informatique.
- On l'appelle modèle hiérarchique à cause de la direction des relations qui s'établissent uniquement du parent vers les enfants, ce qui forme un arbre hiérarchique.

Limite

Il ne peut y avoir qu'un seul arbre et les relations ne peuvent se dessiner que du parent vers l'enfant.

Codasyl, Cobol et le modèle réseau (les années 1960)

- En 1959, le Codasyl (Conference on Data Systems Languages (=Conférence sur les langages de systèmes de traitement de données)) conduit à la création d'un consortium dont l'objectif était de développer des standards de gestion de données.
- On a établi un langage d'interrogation de données appelé COBOL (COmmon Business Oriented Language) qui a fait l'objet d'une norme ISO et qui a évolué en plusieurs versions.
- COBOL est un langage *navigationnel* : chaque pointeur est posé et maintenu sur une entité, appelée entité courante, et il se déplace selon des besoins vers d'autres entités. Quand on travaille avec plusieurs entités, chaque entité a son pointeur maintenu sur un article.

Codasyl, Cobol et le modèle réseau (les années 1960)

- Après avoir établi le langage de traitement de données, les membres du Codasyl ont établi un standard de structuration des données qu'on a nommé *le modèle de données réseau*.

Limite

Le problème majeur des systèmes de gestion de bases de données réseau était leur langage de requête, complexe et navigationnel (COBOL).

Edgard Frank Codd et le modèle relationnel (les années 1970)

- Edgar Frank Codd est un informaticien britannique qui a travaillé comme programmeur pour IBM aux États-Unis en 1948.
- Après l'obtention du diplôme de doctorat en Computer Science, Il a commencé à travailler comme chercheur aux laboratoires de recherche d'IBM à San Jose, en Californie au milieu des années 1960 où il a élaboré l'organisation des données selon un modèle basé sur la théorie mathématique des ensembles.
- Les investissements effectués par la société pour le système IMS (hiérarchique) n'ont pas incité IBM à s'intéresser à ce nouveau modèle.

Edgard Frank Codd et le modèle relationnel (les années 1970)

- Un an plus tard, Codd a publié un article intitulé *A Relational Model of Data for Large Shared Data Banks* dans Communications of the ACM, revue de l'association pour la machinerie informatique (Association for Computing Machinery).
- L'article pose les bases du modèle relationnel en indiquant les bases mathématiques et algébriques de relations.

Edgard Frank Codd et le modèle relationnel (les années 1970)

- Le modèle de Codd est un système de relations basé uniquement sur les valeurs des données et la manipulation de ces dernières à l'aide d'un langage de haut niveau implémentant une algèbre relationnelle.
- En 1974, le laboratoire de San Jose a développé un prototype, le System R, pour expérimenter les concepts proposés par Codd. On a développé un langage de manipulation de données nommé Sequel (Structured English Query Language).
- Le nom du langage Sequel était ensuite remplacé par SQL (Structured Query Language).

Edgard Frank Codd et le modèle relationnel (les années 1970)

- A l'université de Berkeley, Eugene Wong et Michael Stonebraker se sont inspirés des travaux de Codd pour bâtir un système nommé Ingres, qui a évolué plus tard à PostgreSQL, Sybase et Informix.
- Larry Ellison s'est aussi inspiré des travaux de Codd et de System R pour développer son moteur de bases de données Oracle.
- IBM a transformé son System R en un vrai moteur commercial, nommé SQL/DS, qui est devenu DB2 par la suite.

Le modèle relationnel (à partir des années 1980)

- Le modèle relationnel a remplacé toutes les autres formes de structuration de données.
- Le succès du modèle relationnel est dû non seulement aux qualités du modèle lui même mais aussi aux optimisations faites au niveau du stockage ce qui a permis la réduction de la redondance des données.

Le modèle relationnel (à partir des années 1980)

- En 1985 Edgar Codd a publié deux articles dans le magazine ComputerWorld *Is Your DBMS Really Relational ?* et *Does Your DBMS Run By the Rules ?* qui résument les caractéristiques de son modèle connues par *les règles de Codd* et qui sont au nombre de treize.
- L'ensemble de ces règles indique la voie à suivre pour les systèmes de gestion de bases de données relationnelles.
  $\Rightarrow$ Elles ne sont jamais totalement implémentées, à cause des difficultés techniques que cela représente.

Les spécificités des SGBDR

- Le modèl relationnel fixe un certains nombre d'opérations de relations entre les tables :
  - Un système de jointure entre les tables permettant de construire des requêtes complexes faisant intervenir plusieurs tables.
  - Un système d'intégrité référentielle permettant de s'assurer que les liens entre les tabels sont valides.
- la plupart des moteurs de SGBDR sont transactionnels ce qui leur impose le respect des contraintes *Atomicity, Consistency, Isolation et Durability* communément appelées les contraintes **ACID**.

les contraintes ACID

- **Atomicity (Atomicité) :** Cela signifie que les mises à jour de la base de données doivent être atomiques, c'est-à-dire qu'elles doivent être totalement réalisées ou pas du tout. Par exemple, sur 5000 lignes qui doivent être modifiées au sein d'une même transaction, si la modification d'une seule échoue, alors la transaction entière doit être annulée.
  $\Rightarrow$ C'est primordial, car chaque ligne modifiée peut dépendre du contexte de modification d'une autre, et toute rupture pourrait engendrer une incohérence des données de la base.

les contraintes ACID

- **Consistency (Cohérence) :** Cela signifie que les modifications apportées à la base doivent être valides, en accord avec l'ensemble de la base et de ses contraintes d'intégrité.
  Si un changement enfreint l'intégrité des données, alors la transaction doit être interdite.

les contraintes ACID

- **Isolation (Isolation) :** Cela signifie que les transactions lancées au même moment ne doivent jamais interférer entre elles, ni même agir selon le fonctionnement de chacune. Par exemple, si une requête est lancée alors qu'une transaction est en cours, le résultat de celle-ci ne peut montrer que l'état original ou final d'une donnée, mais pas l'état intermédiaire. Les transactions doivent donc s'enchaîner les unes à la suite des autres, et non de manière concurrentielle.

les contraintes ACID

- **Durability (Durabilité) :** Cela signifie que toutes les transactions sont lancées de manière définitive. Une base ne doit pas afficher le succès d'une transaction, pour ensuite remettre les données modifiées dans leur état initial.
  Pour ce faire, toute transaction est sauvegardée dans un fichier journal, de sorte que si un problème survient empêchant sa validation complète, la transaction pourra être correctement terminée lors de la disponibilité du système.

Le passage au XXIe siècle et les limites du modèle relationnel

L'émergance du Big Data

- Le passage au XXIe siècle était marqué par une multiplication exponentielle des volumes de données par certaines entreprises notamment celles en rapport avec Internet : données scientifiques, réseaux sociaux, opérateurs téléphoniques, bases de données médicales, agences nationales de défense du territoire, indicateurs économiques et sociaux, etc.
- Des volumes de données qui se comptent pétaoctets (100 000 téraoctets) et c'est ce que les anglo-saxons ont appelé le Big Data.

Le passage au XXIe siècle et les limites du modèle relationnel

Le Big Data : définition

l'ensemble des tachnologies dédiées aux traitements de données de forte volumétrie et qui répondent aux 4V :

- Volume important des données.
- Variété des informations hétérogènes peu ou non structurées.
- Vélocité exigée dans le traitement due à la fréquence de création et/ou de collecte.
- Variabilité permettant de s'adapter au format changeant des données.

Possibilité de rajouter un 5èmeV :

- Véracité des données qui est un élément important à intégrer dans un processus de traitement des données.

Le passage au XXIe siècle et les limites du modèle relationnel

Ces données gigantesques ont incité le passege d'un système centralisé à un système distribué pour la gestion de données.

La gestion et le traitement de ces données hétérogènes et volumineuses sont considérés comme un nouveau défi de l'informatique, et les moteurs de bases de données relationnelles traditionnels, hautement transactionnels, semblent totalement dépassés.

Le passage au XXIe siècle et les limites du modèle relationnel

Les SGBDRs et les systèmes distribués

Le problème d'adaptation des SGBDR aux environnements distribués était levé par les fournisseurs de services en ligne (Yahoo ; Google, etc.) et les acteurs du web social (Facebook, Twitter, LinkedInd, etc.) qui ont présenté les besoins suivants :

- **Scaling des données :** Capacité à répartir les données entre un nombre important de machines afin d'être en mesure de stocker de très grands volumes de données.
- **Scaling des traitements :** Capacité à distribuer les traitements sur un nombre de machines important afin d'être en mesure d'absorber des charges très importantes.
- **Distribution de données :** Le stockage de données sur plusieurs datacenters afin d'assurer une continuité de service en cas d'indisponibilité de service sur un datacenter.

Le passage au XXIe siècle et les limites du modèle relationnel

Système centralisé vs système distribué

- Système centralisé : tout est localisé sur la même machine et accessible par le programme
  - Système logiciel s'exécutant sur une seule machine.
  - Accédant localement aux ressources nécessaires (données, code, périphériques, mémoire ...)
- Système distribué : une définition parmi d'autres (Andrew Tannenbaum)
  - Ensemble d'ordinateurs indépendants connectés en réseau et communiquant via ce réseau.
  - Cet ensemble apparaît du point de vue de l'utilisateur comme une unique entité.

Le passage au XXIe siècle et les limites du modèle relationnel

- Dans un système centralisé les contraintes ACID sont aisées à garantir.
- Dans un système distribué, il est nécessaire de distribuer les traitements de données entre différents serveurs.
  - Difficulté de maintenir les contraintes ACID à l'échelle du système distribué entier tout en maintenant des performances correctes.
  - Sur la plupart des SGBD relationnels, il convient de s'assurer en permanence que les données liées entre elles sont placées sur le même noeud du serveur. Lorsque le nombre de relations au sein d'une base augmente, il devient de plus en plus difficile de placer les données sur des noeuds différents du système.

Le passage au XXIe siècle et les limites du modèle relationnel

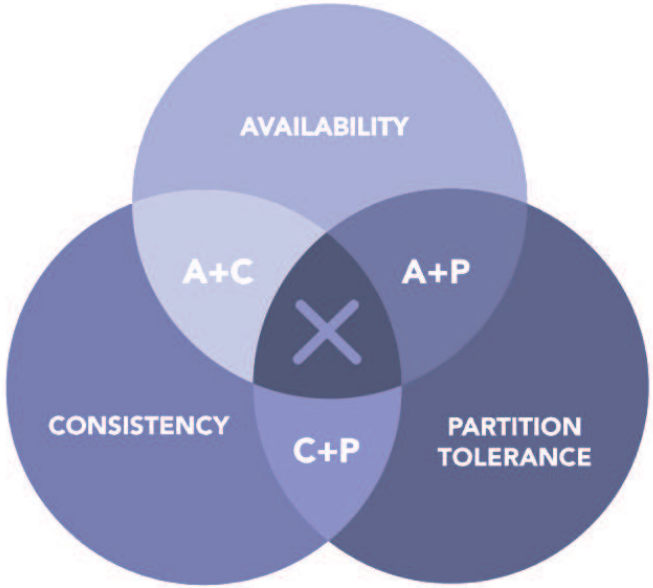
Les systèmes distribués et le théorème de CAP

le théorème de CAP (Consistency-Availibility-Partition tolerance). énnonce trois grandes propriétés pour les systèmes distribués :

- **Consistency ou Cohérence :** tous les noeuds du système voient exactement les mêmes données au même moment.
- **Availability ou Disponibilité :** la perte d'un noeuds n'empêche pas les survivants de continuer à fonctionner correctement.
- **Partition tolerance ou Résistance au partitionnement :** aucune panne moins importante qu'une coupure totale du réseau ne doit empêcher le système de fonctionner correctement.

Le passage au XXIe siècle et les limites du modèle relationnel

Le théorème de CAP stipule qu'il est impossible d'obtenir ces trois propriétés en même temps dans un système distribué et qu'il faut donc en choisir deux parmi les trois.



Les bases de données relationnelles implémentent les propriétés de Cohérence et de Disponibilité (système AC).

Le passage au XXIe siècle et les limites du modèle relationnel

Autres points faibles des SGBDRs

Soit le modèle relationnel suivant :

- Livre(NumLiv, Titre, Prix, #NumAut)
- Auteur(NumAut, Nom, Prénom, Adresse)
  - Dans ce schéma un livre est écrit par un et seul auteur et un auteur est à l'origine de 0 à n livres.
  - Supposons que ce schéma a donné lieu à une base de données relationnelle (MySQL, Oracle, etc.) et qu'au bout de quelques semaines d'utilisation la base contienne 100000 livres et 110000 auteurs.

Le passage au XXIe siècle et les limites du modèle relationnel

Autres points faibles des SGBDRs

- Supposons maintenant qu'un résumé est fourni par les maisons d'édition pour chaque nouveau livre et qu'il faut stocker cette information dans la base car elle facilitera le développement d'un futur magasin en ligne en autorisant des recherches complexes par mots-clés. Que faut-il faire ?
  - Il faut ajouter un champ dans la table Livre qui contiendra 100000 livre dont le champs résumé ne sera pas rensigné : Ce champs ne sera renseigné qu'à partir du tuple 1000001.

Limite

Les bases de données relationnelles ne supportent pas les structures changeante. $\Rightarrow$ Besoin de déclarer au préalable l'ensemble des champs représentant un objet.

Problématique

Les SGBDR règnent en maîtres pour le stockage et la manipulation de données depuis plus que 20 ans...mais

- ne sont pas adaptés aux environnements distribués requis par les volumes gigantesques de données.
- sont incapables de gérer des structures hétérogènes et/ou changeantes.

Solution

Not only SQL (NoSQL), nommé aussi NoRel et inventé par Carl Strozzi en 2009.

- C'est l'ensemble de technologies de stockage et de gestion de données hétérogènes et volumineuses.
- Ce sont des approches qui viennnent compléter les outils existants (i.e. SGBDR) afin de combler leurs faiblaisses.

NoSQL versus SQL

Les Bases de données NoSQL

- sont adaptées aux systèmes distribués.
- sont généralement des systèmes CP (Cohérent et Résistant au partitionnement) ou AP (Disponible et Résistant au partitionnement).

NoSQL versus SQL

- La plupart des SGBD de la mouvance NoSQL ont été construits en faisant fi des contraintes ACID quitte à ne pas proposer de fonctionnalités transactionnelles.
- La majeure partie des outils développés dans le cadre de la mouvance NoSQL permettent l'utilisation d'objets hétérogènes apportant comparativement une bien plus grande flexibilité dans les modèles de données ainsi qu'une simplification de la modélisation.

Des solutions NoSQL ? ...Pourquoi pas une solution unique ? !

Pourquoi plusieurs solutions NoSQL ?

- Tout comme les SGBD relationnels ne sont pas la réponse unique aux problématiques de base de données, on ne doit pas considérer qu'un seul autre outil sera en mesure d'apporter une solution au caractère universel.
- La mouvance NoSQL ne suggère pas l'abandon total de SQL mais se traduit plutôt par *Not Only SQL* et non *No SQL*.
- Il existe une diversité importante d'approches. que nous classons en quatre grandes catégories : Paradigme clé / valeur, Bases orientées colonnes, Bases documentaires et Bases orientées graphes.

Le modèle clé-valeur / Key-value

Principe

Le modèle clé-valeur est un modèle basique où chaque objet est identifié par une clé unique constituant la seule manière de le requêter.
⇒ adapté à la gestion de caches ou pour un accès rapide aux informations.

- Une BD NoSQL de type clé-valeur fonctionne comme un grand tableau associatif et retourne une valeur dont on ne connaît pas la structure à partir d'une clé.
- Les données sont simplement représentées par un couple clé-valeur.
- La valeur peut être une simple chaîne de caractères, ou un objet sérialisé.

Le modèle orienté colonne

Principe

Le modèle orienté colonne permet de disposer d'un très grand nombre de valeurs sur une même ligne, de stocker des relations one-to-many et d'effectuer des requêtes par clé.
$\Rightarrow$ adapté au stockage de listes : messages, posts, commentaires,...

- Les données sont stockées par colonne et non par ligne.
- Le modèle est proche d'une table dans un SGBDR mais le nombre de colonnes :
  - est dynamique.
  - peut varier d'un enregistrement à un autre ce qui évite de retrouver des colonnes ayant des valeurs NULL.

Le modèle orienté document

Principe

Le modèle orienté document permet de gérer des collections de documents, composés chacun de champs et de valeurs associées, ces dernières peuvent être requêtées.
$\Rightarrow$ adapté au stockage de profils utilisateur.

- Les BD NoSQL documentaires sont basées sur le modèle clé-valeur tels que les documents représentent les valeurs.
- Les documents n'ont pas de schéma, mais une structure arborescente : ils contiennent une liste de champs, un champ a une valeur qui peut être une liste de champs.

Le modèle orienté graphe

Principe

Le modèle orienté graphe permet de gérer des relations multiples entre les objets.
$\Rightarrow$ adapté à la modélisation, le stockage et la manipulation de données très complexes liées par des relations variées(issues de réseaux sociaux par exemple).

- Les BD graphe permettent une représentation des données basée sur la théorie des graphes.
- Elles s'appuient sur les notions de noeuds, de relations et de propriétés qui leur sont rattachées.

Exemples de SGBD NoSql

Modèle	Clé-Valeur	Colonne	Document	Graphe
Complexité du modèle	+	++	++	+++
Produits logicels	Voldemort Redis Riak	Hbase Cassandra HyperTable	MongoBD CouchBD Couchbase	Neo4j OrientDB DEX InfiniteGraph

1) Pourquoi on trouve plusieurs typologies de bases de données NoSQL ?
2) Quelles sont les principales typologies des bases de données NoSQL ?
    a. Donner un exemple de SGBD pour chaque représentation.
    b. Donner un domaine d'application approprié pour chaque représentation.
3) Donner trois atouts du modèle relationnel.
4) Donner trois atouts des bases de données NoSQL.

5) Est ce que les contraintes ACID sont garanties par les bases de données NoSQL ? Justifier la réponse
6) Enoncer le théorème CAP. Faites l'illustration à l'aide d'un exemple de votre choix.
7) Quelles sont les propriétés du théorèmes CAP garanties par les bases de données relationnelles ? Pourquoi ?