LE NO-SQL, EMOI ET MOI ET MOI

ou comment se poser les bonnes questions quand on se demande à quoi ça sert…

Présenté par Guillaume SUEUR
DécryptaGéo 2017

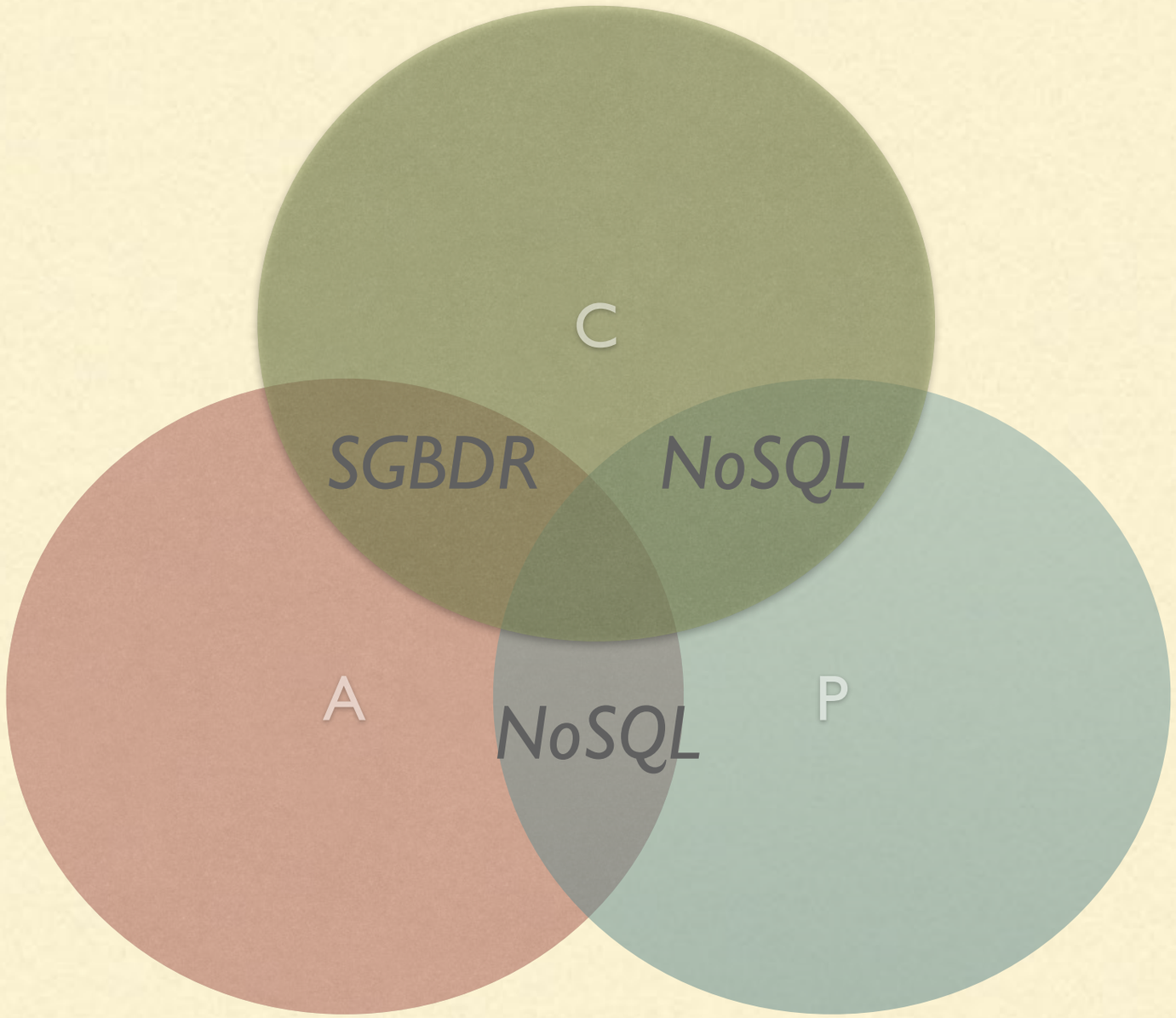
A L'ORIGINE DU NOSQL…

- Un théorème célèbre des systèmes informatiques oppose **trois notions fondamentales** et précise qu'un système distribué ne peut en **combiner que deux.**
  Cela s'applique aux bases de données :

SGBDR

- **A**vailibility (**Disponibilité**) : chaque client peut toujours lire et écrire dans la base
- **C**onsistency (**Cohérence**) : tous les clients ont la même vue de la donnée (ACID)
- **P**artition Tolerance (**Tolérance** à la partition) : le système peut fonctionner sur plusieurs noeuds physiques simultanés.

A L'ORIGINE DU NOSQL…



A L'ORIGINE DU NOSQL…

- L'émergence de besoins **exponentiels** en matière de **stockage** (plusieurs To) a nécessité des modèles **distribués**.

- Les bases NoSQL ne sont **pas des bases de données relationnelles,** elles mettent l'accent sur la **scalabilité** et les **performances**.

- Google (BigTable), Amazon (Dynamo), Facebook (Cassandra, puis HBase), Sourceforge (MongoDB), Ubuntu One (CouchDB), Hadoop…

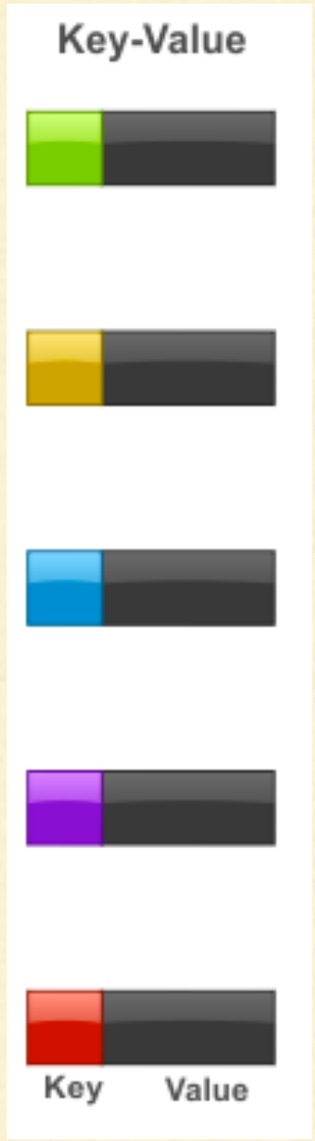
- La première convention « NoSQL » a lieu en juin 2009 et baptise par ce terme ces nouvelles technologies.

UTILISER NOSQL

- NoSQL = **Not Only SQL**, donc Not Only Structured Query Language
- Des bases de données manipulables via des interfaces **HTTP** (mais pas de manière standardisée comme le SQL)
- Des serveurs organisés en **cluster** de noeuds interconnectés.
- Des bases de données qui échangent généralement au format **JSON**
- HTTP + JSON les rendent très facilement intégrable en environnement web.

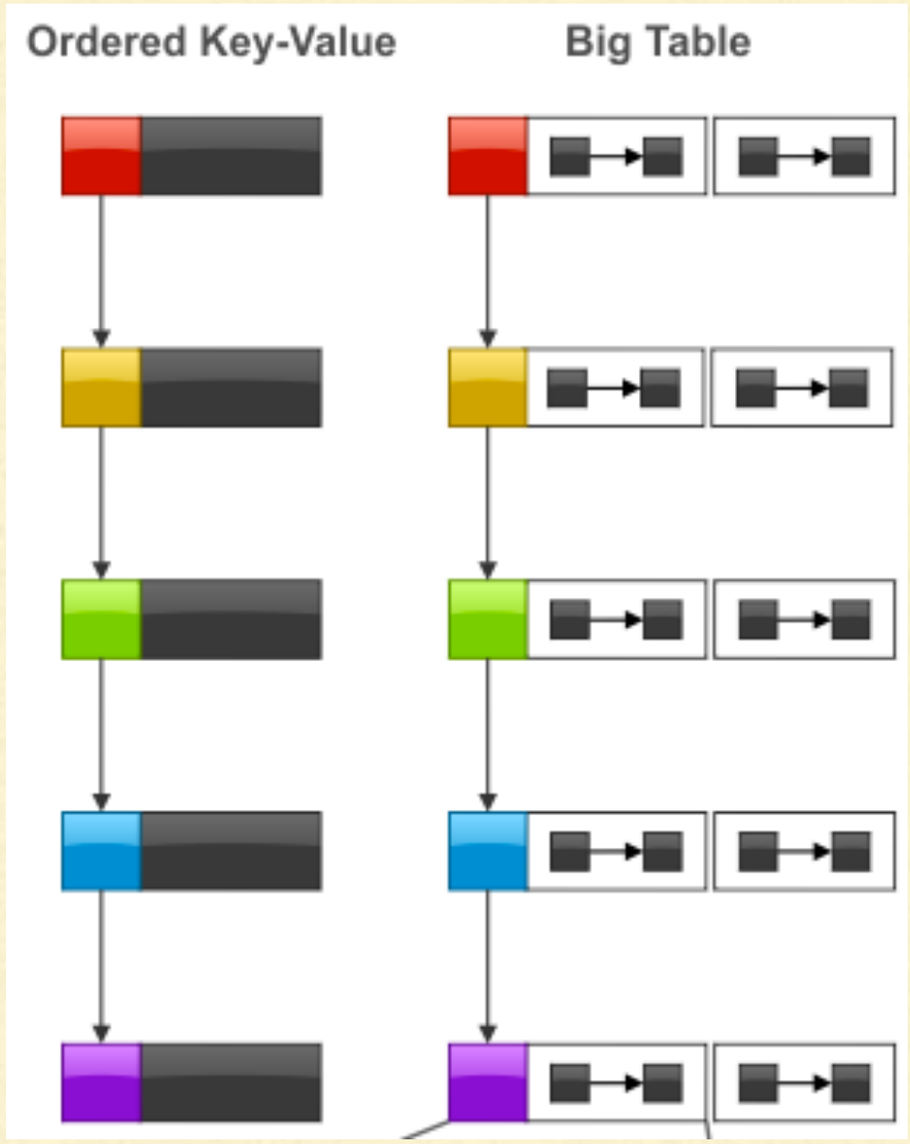
IL Y A PLUSIEURS NOSQL

- Orienté clés-valeur
  - Permet de stocker des couples (ID, data) la donnée pouvant être à peu près n'importe quoi (texte, fichier, photo…)
  - N'a pas vocation à permettre de croiser les données entre elles
  - REDIS, BerkeleyDB, MemcacheDB



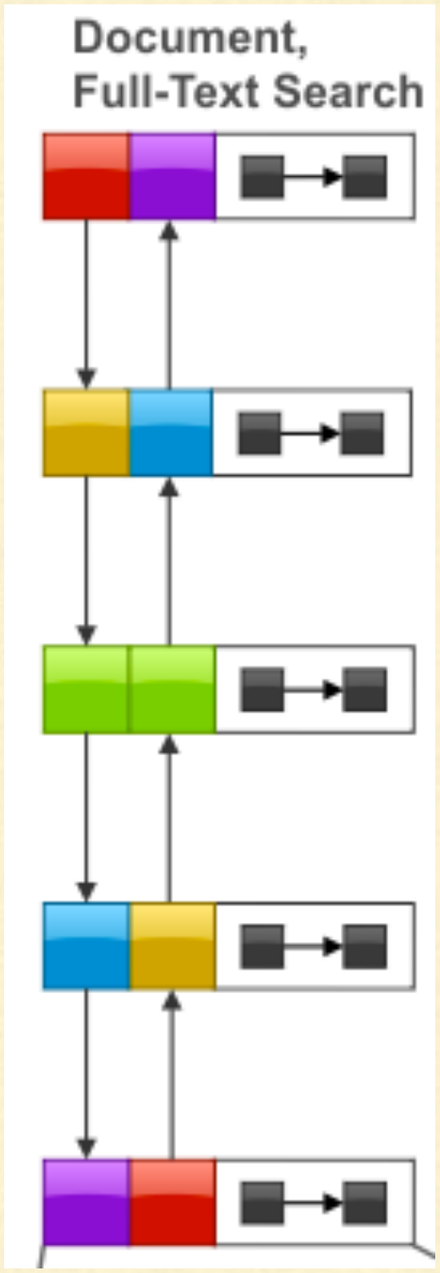
IL Y A PLUSIEURS NOSQL

- Orienté colonne (extension du modèle précédent vers clé-valeur**s**)
  - Comme une table classique, mais le nombre de colonnes est dynamique, déterminé par la structure des enregistrements insérés.
  - Facilite les traitements sur de gros volumes de données (datamining, analyses diverses…)
  - BigTable (Google), Accumulo, HBase, Cassandra, Hypertable



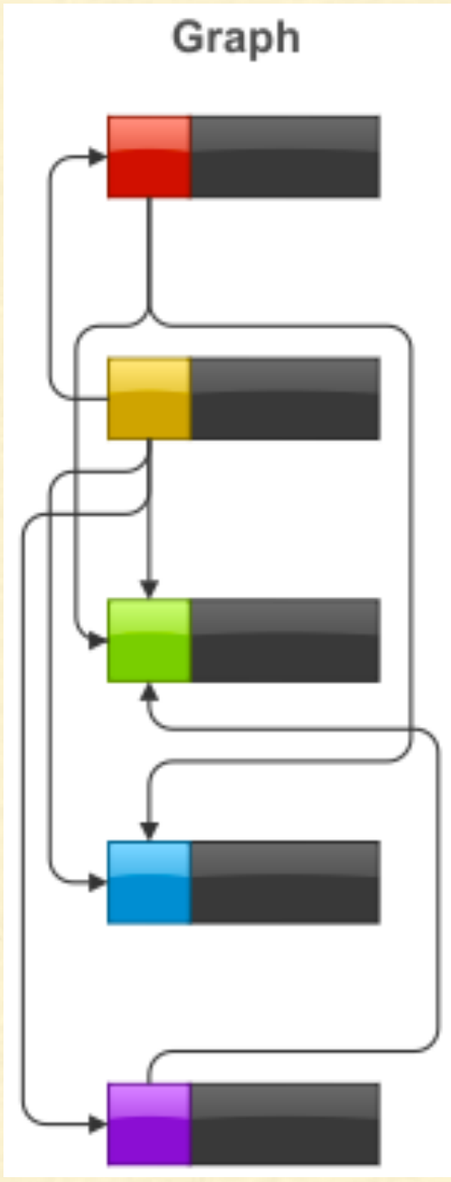
IL Y A PLUSIEURS NOSQL

- Orienté Document
  - Permet de stocker des structures complexes sans schéma prédéterminé
  - Mais la notion de schéma reste essentielle pour indexer et retrouver les informations pertinentes.
  - CouchDB, MongoDB, ElasticSearch…



IL Y A PLUSIEURS NOSQL

- Orienté Graphe
  - Permet de modéliser des relations entre entités basées sur la théorie des graphes.
  - Utilisé par les réseaux sociaux pour mettre en contact avec les amis, les achats susceptibles de vous intéresser…
  - Neo4J



Stop following me, you fucking freaks!
Key-Value
Ordered Key-Value
Big Table
Document,
Full-Text Search
Graph
SQL
Key
Value

QUAND RECOURIR AU NOSQL ?

- Quand il y a une problématique de scalabilité :

- Quand on ne sait pas les questions auxquelles il faudra répondre :

- Quand on se sait pas ce qu'on va stocker mais qu'on veut pouvoir l'exploiter quand-même :

- Fichiers divers
- Structures variables (plusieurs versions d'une norme)
- Schémas complexes

QUAND RECOURIR AU NOSQL ?

- D'une certaine manière, le NoSQL devient utile quand on ne sait pas

  - Combien de données…

  - Quelles données…

  - Quels usages…

NOSQL ET INFORMATION GEOGRAPHIQUE

- **Mécanismes d'indexation et de recherche** fréquents mais rarement **optimisés** (types simples, bbox, opérations simples : intersection, inclusion…)
  - **MongoDB** : cartouche spatiale comprenant les différents types d'objets, et permettant les requêtes spatiales d'inclusion, intersection et proximité.
  - **ElasticSearch** propose les différents types géographiques, et les opérateurs INTERSECTS, DISJOINT, WITHIN, CONTAINS
  - **CouchDB** dispose d'une cartouche spatiale GeoCouch, mais aux fonctionnalités limitées.
- Pas de **systèmes de coordonnées**, EPSG:4326 pour tout le monde !
- Pas pour le **traitement** de données SIG, mais certainement plutôt pour la **publication**.

NOSQL ET PUBLICATION DE SERVICES GEOGRAPHIQUES

- **GeoMesa** est un projet récent (pas forcément encore bien cuit…) reposant sur **Accumulo** et disposant d'un plugin **GeoServer**.

- **FME** permet d'alimenter des bases NoSQL à partir du format **JSON/GeoJSON** attendu par celles-ci.

- **GDAL/OGR** a des drivers R/W pour MongoDB, CouchDB et ElasticSearch. Ca permet donc théoriquement d'exploiter ces serveurs depuis **MapServer**.

NOSQL ET PUBLICATION DE SERVICES GEOGRAPHIQUES

- Au-delà des outils, d'autres problèmes se posent dès qu'il s'agit de publication…

- Les standards OGC requièrent implicitement un modèle de **base de données relationnelle** (voir standard SOS pour les capteurs par ex.) qu'il peut être compliqué d'implémenter en NoSQL.

- Les spatial datasets INSPIRE reposent aussi sur une approche relationnelle forte et des **structures de tables strictes** le tout au format **XML**…

NOSQL ET PUBLICATION DE SERVICES GEOGRAPHIQUES

- Le monde **OpenData** correspond mieux aux préoccupations NoSQL
- Pas de schéma **prédéfini**
- Volume à stocker **inconnu** a priori
- Besoin de **disponibilité** car public plus large

NOSQL ET PUBLICATION DE SERVICES GEOGRAPHIQUES

- Pour se lancer il faut se poser deux questions :
  Ma base doit-elle être tolérante à la partition (scalable) ?
  Ma base contient-elle peu de relations ?

Si vous répondez non à une, prenez PostgreSQL.

- Pour qui se lance, il faut alors réfléchir en amont au(x) :
  - **Type de système** NoSQL (KV, colonne, document, graphe)
  - **Produit** lui-même, car ils peuvent être forts différents
  - Modalités d'**alimentation** et d'**exploitation** de la base de données (puisque pas de standard comme le SQL).

MERCI POUR VOTRE ATTENTION

DES

QUESTIONS ?