Bases de données NO SQL et SIG : d'un existant restreint à un avenir prometteur

Christian Carolin, *AXES Conseil*
carolin@axes.fr - http://www.axes.fr

Les bases de données relationnelles constituent désormais le support de stockage usuel des données géographiques. Les formats propriétaires SIG, même de grande diffusion, ont maintenant vocation à un usage local ou à être exploités comme format d'échange.

Un nouveau type de bases de données, apparu ces dernières années, répond à des besoins de souplesse de formalisation, de volumétrie croissante de données, de moindre structuration initiale de l'information, de plus haute performance et disponibilité. Ce type de base est nommé NO SQL, porté notamment par la vague *Big Data*. L'interprétation provocatrice « *Pas de SQL* » se passe de commentaires, l'acronyme signifiant à l'origine *Not Only SQL*.

On lit tout et son contraire à propos des bases *NO SQL*. *A priori*, le monde SIG, qui répète depuis plusieurs décennies qu'il faut structurer l'information, créer des modèles normés, que la qualité des résultats dépend de la complétude de l'information, ne serait pas concerné par ce type de base. On constate néanmoins que des fonctions *NO SQL* dédiées SIG existent et que des éditeurs fournissent déjà des solutions dans ce domaine.

Big Data constitue un vecteur de la croissance volumique de l'information numérique. *NO SQL* y répond en constituant l'une des solutions qui permet de répondre au traitement de données de masse non- ou semi-structurées. Les données non- ou semi-structurées représentent environ 80 % des données numériques stockées à l'échelle mondiale et, au moment où le monde de la géomatique est confronté à l'explosion des volumes d'information, des solutions qui permettent potentiellement d'y faire face ne peuvent être ignorées.

Principes et un peu de théorie…

Il est possible de caractériser les bases *NO SQL* par les principes suivants :

▶ **L'absence de modèle unique :** *NO SQL* répond à un besoin satisfait par la mise en œuvre de divers outils et implémentations. Le modèle relationnel n'est pas utilisé, les solutions d'interrogation de la base sont limitées. Les SGBDR exploitent en théorie le paradigme formalisé par les règles de CODD (*http://itsy.co.uk/ac/0405/sem3/44271_ddi/lec/3_coddsrules.htm*), mais il n'existe rien d'équivalent en *NO SQL*. C'est le besoin qui déterminera le choix d'une solution parmi la typologie présentée plus loin ;

▶ **Le théorème CAP :** les bases *NO SQL* représentent une application concrète du théorème CAP d'Eric Brewer. Celui-ci dit qu'il est impossible, sur un système distribué, de garantir en même temps les trois contraintes suivantes :

- Cohérence (*Consistency*) : tous les noeuds d'un système voient exactement les mêmes données au même moment ;
- Haute disponibilité (*Availability*) : Les données sont toujours disponibles ;
- Tolérance au partitionnement (*Partition Tolerance*) : une

panne partielle doit permettre de maintenir les données accessibles, notamment grâce au principe de partitionnement.

Le théorème CAP admet que seuls deux des trois principes énoncés peuvent être réalisés simultanément (source : *http://www.julianbrowne.com/article/viewer/brewers-cap-theorem*) ;

▶ **L'anti-abstraction :** depuis de nombreuses années, les éditeurs (SIG ou autres) se sont efforcés de dissocier les couches physiques (matériels et infrastructures) des couches logiques (logiciel), afin de faciliter le portage et une approche multi-plateformes. Les implémentations *NO SQL* d'envergure opèrent exactement à l'inverse de cette démarche, en exploitant les capacités intrinsèques des architectures, à grand renfort de techniques et d'algorithmes connus (*sharding*, *hashcode*, etc.), afin d'obtenir des performances optimales. Au vu du couplage fort entre l'infrastructure, les couches *middleware* et le logiciel de base *NO SQL* proprement dit, le paramétrage abstractif pour l'implémentation d'une base volumineuse multi-serveurs/clusters ne présente donc pas grand intérêt ;

▶ **Map/Reduce :** *Map/Reduce* est une technique de programmation permettant de manipuler de grandes quantités de données en les distribuant dans un cluster de nœuds, afin d'effectuer des calculs et des traitements. Pour simplifier, *Map* représente l'invocation de la fonction et la recherche de la donnée, *Reduce* produit le résultat agrégé. L'implémentation la plus célèbre est certainement le *framework Hadoop*, développé par la fondation *Apache*. *Map/Reduce* constitue un élément essentiel de l'exploitation des données d'une base *NO SQL*.

Quelques constats :

▶ **Des outils de manipulation de données limités :** pas ou peu de fonctions relationnelles, pas de langage de requêtes performant, des interfaces d'administration sommaires ; cette technologie présente tous les défauts de jeunesse possibles. Néanmoins, la communauté *OpenSource* notamment, est très active sur ce domaine et les outils progressent de mois en mois, sans toutefois atteindre encore les capacités fonctionnelles connues dans le monde SGBDR ;

▶ **Et la qualité transactionnelle ?** Clairement, la plupart des bases *NO SQL* sont difficilement ACID (Atomicité, Cohérence, Isolation et Durabilité). Ceci n'est pas directement lié à leurs propriétés intrinsèques, mais, sur des bases volumineuses, à la notion d'environnement distribué. Dans un contexte SI ou SIG, il faut donc évaluer le risque que peut poser l'insuffisance éventuelle de sécurité transactionnelle ;

▶ **L'existant normatif SIG :** une déclinaison géographique du format JSON (de plus en plus utilisé) existe : *GeoJSON*. Elle permet de définir des données répondant au triptyque classique point/ligne/polygone - à noter que sa définition n'est pas pilotée par l'OGC. *GeoJSON* a sa propre extension topologique (*TopoJSON*).

Typologie des bases NO SQL

Une classification, désormais classique, est résumée ci-après :

▶ **Bases de données clé-valeur :** il s'agit de tables organisées en fonction d'un « *hashage* » de la clef, qui constitue le point d'entrée unique aux données (valeurs). La répartition est fonction de l'algorithme utilisé pour le *hash*. Cette répartition a vocation à être mise en œuvre sur des bases volumineuses ;

▶ **Bases de données orientées documents :** il s'agit d'une implémentation évoluée du modèle clef-valeur, permettant l'ajout et la suppression de champs et l'indexation de champs d'un document. L'utilisateur va disposer d'un *id* interne attribué automatiquement, charge à lui de construire un identifiant fonctionnel (exemple : numéro de parcelle, n° de tronçon de voie, etc.). La structuration des documents est réputée normée, le plus souvent via une notation *Javascript* (JSON) ou XML ;

▶ **Bases de données orientés colonnes :** c'est un système de colonnes dynamiques, triées et assemblées en familles de colonnes. Ces bases peuvent atteindre plusieurs millions de colonnes. Elles sont peu adaptées à des mises à jour fréquentes, le mode ajout est privilégié. Les possibilités d'interrogation sont assez limitées. L'usage est donc réservé à des stockages conséquents visant des millions d'utilisateurs (par exemple : *Facebook*) ;

▶ **Bases de données orientées graphes :** souvent issu de la théorie des graphes, ce type de base est fondé sur la gestion de relations complexes entre objets. Le modèle de graphe permet de représenter des réseaux physiques et logiques et de manipuler des notions liées par exemple, à la connexion ou à la

proximité. Il s'agit probablement de la base conceptuellement la plus complexe à mettre en œuvre.

Le besoin

NO SQL constitue typiquement une réponse à des besoins que les bases de données relationnelles ne peuvent pas aisément satisfaire, ou répondent à des conditions techniques et financières peu favorables :

- **Extensibilité (scalability) :** l'ajout marginal d'un enregistrement ou d'un set d'enregistrements ne doit pas dégrader, à ressources constantes, le fonctionnement nominal au-delà d'un niveau proportionnel acceptable ;

- **Résilience :** le théorème CAP confirme le maintien opérationnel des données d'une base partiellement déficiente ;

- **Performance :** le modèle entité/relation est, dans sa déclinaison physique, porteur de complexité et de consommation de ressources. Chacun sait par exemple, que les jointures de tables volumineuses d'un SGBDR coûtent cher en termes de ressources machine. Ce mode de liaison entre tables n'existe pas en *NO SQL* ;

- **Souplesse de structuration :** le mode de mémorisation de la donnée sous forme de colonnes ou de documents, permet d'ajouter dynamiquement des données qui structurent la base.

L'offre

L'*Open Source* constitue la grande majorité des solutions du marché. Des éditeurs commerciaux se positionnent néanmoins sur cette technologie. Le tableau ci-après présente les principales solutions du marché et les acteurs qui les ont mises en œuvre.

Types de base	Bases	Implémentations célèbres
Clé / valeur	Redis, Riak, Voldemort	Linkedin
Colonnes	BigTable, HBase, Cassandra, DynamoDB	Amazon, Google, Facebook
Document	MongoDB, CouchDB, Couchbase, RavenDB	Expedia, SAP, Salesforce
Graphe	Neo4J, Oracle Graph	WalMart, Accenture, Boomberg, eBay

Et Oracle ?

Acteur commercial emblématique de la base de données, **Oracle** se devait de proposer une solution. Celle-ci existe (*Oracle NoSQL Database* et *Oracle Graph*). En version 12, ces différents produits sont intégrés/connectés au moteur de sa base données relationnelle, bénéficiant ainsi de ses propriétés ACID et de ses fonctions géospatiales (cartouche spatiale sur la version *Enterprise*).

PostGreSQL/PostGIS

Dans une version récente, *PostGreSQL* exploite nativement le format JSON. *PostgreSQL* fournit désormais (version 9.4) des solutions d'indexation et des fonctions et des opérateurs pour manipuler les données JSON.

NO SQL et SIG

Jusqu'à présent, le monde SIG ne semblait pas excessivement motivé par l'exploitation de cette technologie. Des *topics* sur les forums dédiés présentent des expériences, mais ils sont peu nombreux, datent souvent de plusieurs années et nombre de liens et URL sont périmés. Néanmoins, des implémentations significatives ont été réalisées, utilisant tout ou partie de la technologie, afin de répondre à des besoins particuliers.

Quelques éditeurs SIG ont développé des connecteurs ou des API leur permettant d'interfacer leur solution à une base *NO SQL* :

- **MongoDB :** *Mongo* est un acteur emblématique des bases de données *NO SQL* de type document. *MongoDB* a son propre vocabulaire : en équivalent SGBDR, une base est plus ou moins un schéma, une collection correspond (approximativement) à une table. *MongoDB* est multi-instances, supporte le *sharding*, permet la création d'index, dispose d'un interpréteur de requête, permettant des opérations classiques (expressions régulières, booléens.) sur des données typées. Les actions sur une base se font par défaut en mode ligne de commande, mais des GUIs de qualité variable existent (*Ghengis*, *UMongo*…). La maîtrise de *Javascript* et de la notation JSON est souhaitable, même en utilisant des interfaces graphiques d'administration.

Attention, le codage est quasi-indispensable, les pilotes pour les principaux SIG en mode client lourd étant soit inopérants, soit obsolètes.

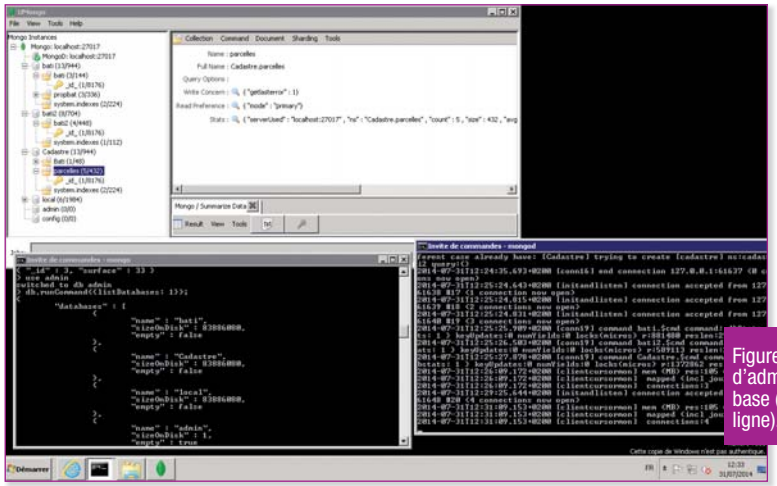

Figure 1 : Écrans d'administration d'une base (GUI et en mode ligne).

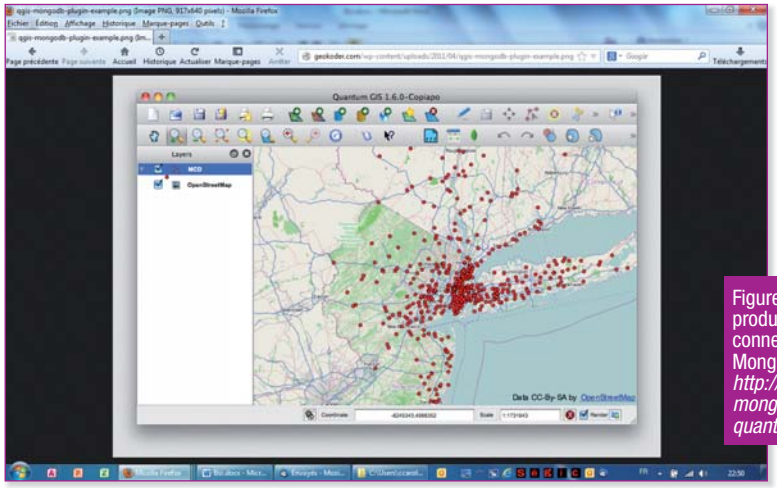

Figure 2 : Carte produite avec QGis connecté à une base MongoDB. *http://geokoder.com/mongodb-plugin-for-quantum-gis.*

▶ **Hadoop :** un des principaux acteurs du marché SIG a développé des outils permettant de réaliser des analyses spatiales sur des données gérées via le célèbre *framework Hadoop*, en appliquant *Map/Reduce* sur des données géographiques. Concrètement, cela permet de filtrer et agréger des traitements et opérations sur des millions, voire milliards d'enregistrements de données spatiales et d'obtenir rapidement une représentation cartographique qui aurait demandé des heures de traitement SQL dans un environnement relationnel ;

▶ **Geocouch :** Issu de la base *NO SQL CouchDB* et de la société *Couchbase*, qui a développé le serveur éponyme, *Geocouch* est une extension spatiale de ces deux produits *NO SQL* orientés document. *GeoCouch* est à *CouchDB/Base* ce que *PostGIS* est à *PostGreSQL*. *CouchDB* est accessible via une API REST. L'interface graphique est nommée *Futon*.

▶ **NEO4J :** *Neo4J* est une des principales bases *NO SQL* orientée graphe. Comme nombre d'autres solutions d'origine *Open Source*, il existe une version entreprise payante (*Neo technology*). Un langage de requête (CQL) permet d'interroger la base constituée. *Neo4J* dispose d'un module spatial, intégrant les principales fonctions de recherche topologique, de routage, de création d'indexes spatiaux. Il ne s'agit pas de création cartographique, mais d'interfacer un SIG avec les fonctions géographiques de *Neo4J*. Lorsqu'une solution SIG dispose nativement de ces fonctions, *Neo4J* perd en partie de son intérêt. À noter que cette base dispose de propriétés ACID.

Que faut-il attendre du NO SQL en géomatique ?

Quelques réflexions peuvent être dégagées des éléments précédents :

▶ **Pas de substitution au monde relationnel :** il semble impossible à ce jour d'envisager le remplacement des SGBDR par des bases *NO SQL* pour gérer des données géographiques. Néanmoins, des volumétries importantes de données géographiques peuvent être stockées (coordonnées d'objets localisés, par exemple) ainsi que des données attributaires de structures et contenus variables, et exploitées avec *NO SQL* (calculs de surfaces, de volumes, via *Map/Reduce*…) dans des conditions de haute performance et des contraintes limitées. *NO SQL* peut donc constituer un complément appréciable sur un projet SIG exploitant une volumétrie significative de données localisées ;

▶ **Une offre en devenir :** la jeunesse de la technologie se concrétise par la profusion d'acteurs et le manque de standards ; la rationalisation du marché n'a pas encore eu lieu. Des implémentations très significatives de *Google* entre autres (*BigTable* est utilisé pour *Google Earth*, par exemple), permettent déjà de confirmer son intérêt, mais des projets plus modestes peuvent également bénéficier de ce type de technologie, à des coûts acceptables ;

▶ **Le maintien de la modélisation des données géographiques :** *NO SQL* ne signifie en aucun cas que la modélisation devient inutile. Le manque de fondamentaux conceptuels pour la formalisation de ces bases ne dispense pas de

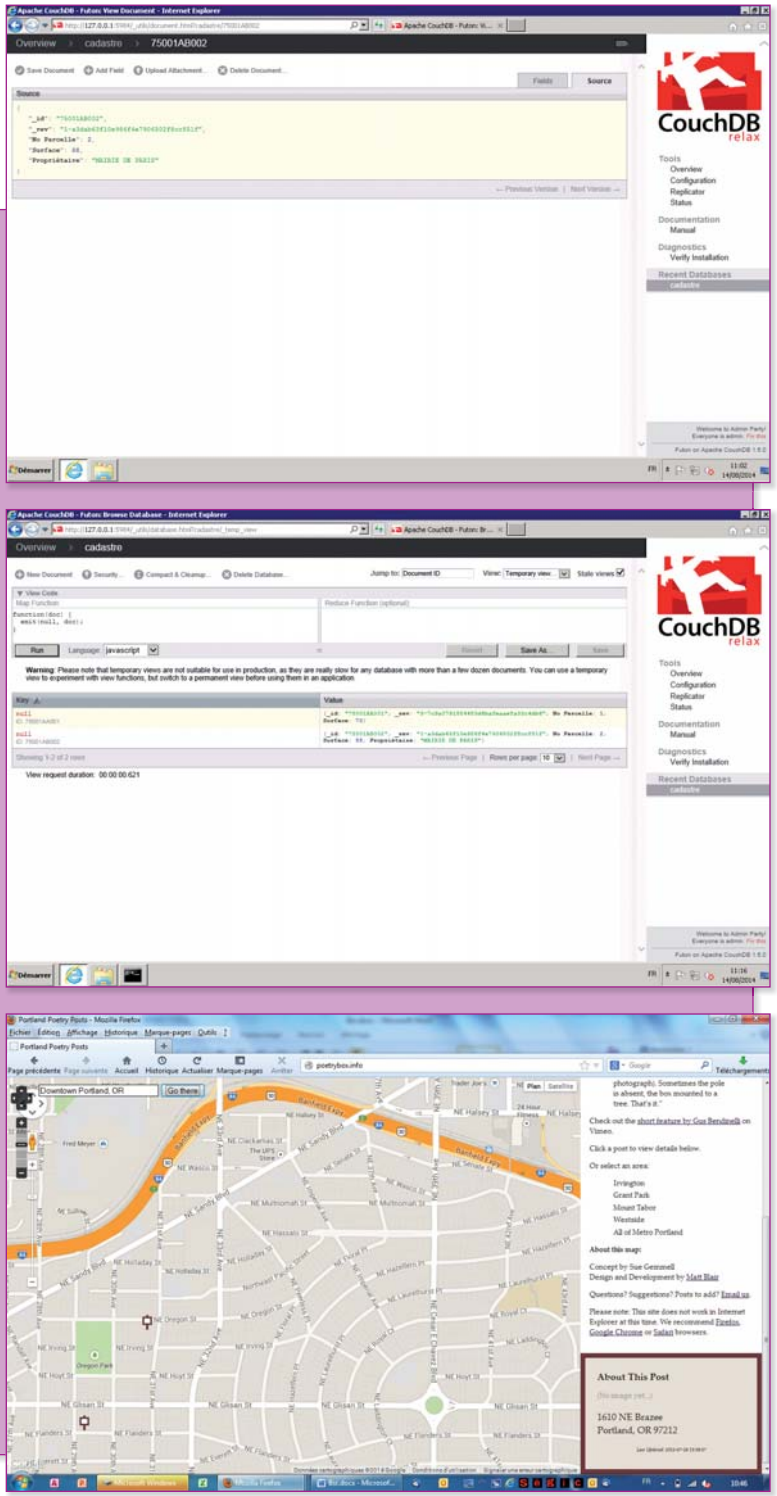

Figure 3 : exemple de construction d'une base parcellaire simple (interrogation via *Map/Reduce*). *http://poetrybox.info* (développement GWT avec des données stockées via *GeoCouch*).

concevoir en amont une structuration de principe des objets, faute de quoi les performances ne seront pas à la hauteur des attentes. Utilisée de façon raisonnable et mesurée, la facilité de construction d'une base peut néanmoins aider à définir une solution cible, via une approche dichotomique entre la conception et l'implémentation ;

▶ **Le choix d'une solution piloté par le besoin et non par l'offre :** retenir une solution *NO SQL* passe impérativement par l'étude préalable du besoin. Une base de données colonnes pour une faible volumétrie ou le choix d'une base documents sans réflexion préalable de structuration des données constituent autant de facteurs qui conduiront à l'échec. On ne se lance pas dans la réalisation d'un projet *NO SQL* sans avoir une idée précise de l'objectif à atteindre. La puissance de ce type de système s'exprime notamment par des techniques telles que le partitionnement. Oubliez donc *NO SQL* pour gérer 500 Mo de données…

▶ **Une maîtrise technique nécessaire :** il est difficile d'envisager une implémentation *HBase* ou *Cassandra*, par exemple, sans solides compétences sur les infrastructures. De même, la compréhension du fonctionnement *NO SQL* est certainement facilitée par une connaissance préalable des SGBDR. Enfin, « *parler JSON dans le texte* » est souvent indispensable et les nostalgiques du mode commande en ligne retrouveront ici la possibilité de manipulation de l'information, là où les GUI disponibles restent souvent limitées…

▶ **Une évolution technique SIG en corrélation avec la technologie SI :** le monde SIG suit les technologies les plus récentes utilisées dans le cadre des systèmes d'information. Les anciens formats propriétaires disparaissent progressivement ou sont exploités de façon marginale, la « *néo-géographie* » exploite désormais *Javascript*, *JSON*, REST, et met le focus sur l'usage et la simplification de l'accès à la cartographie. *NO SQL* devrait, à moyen terme, faire partie des solutions possibles répondant à cette évolution.

À suivre dans l'avenir le *NewSQL*, dont l'approche « *Best of Breed* » doit apporter le meilleur des deux mondes : la puissance d'une architecture distribuée associée aux fonctionnalités apportées par les SGBDR. |