CHAPITRE 1

LE NOSQL

CHAPITRE 2

SQL VS NOSQL

CHAPITRE 3

ÉCHANGE ENTRE SQL ET NOSQL

BIBLIOGRAPHIE

[1] Scriptol. Nosql. *Scriptol* http://www.scriptol.fr/programmation/nosql.php, 2011.

[2] Robert Rees. Nosql, no problem : An introduction to nosql databases. http://www.thoughtworks.com/articles/nosql-comparison.

[3] Michaël Figuière. Nosql europe : Tour d'horizon des bases de données nosql. http://blog.xebia.fr/2010/04/21/nosql-europe-tour-dhorizon-des-bases-de-donnees-nosql/, 21 avril 2010.

[4] Marwan KHELIF. Nosql : premiers pas avec cassandra. http://www.mkhelif.fr/2010/10/01/nosql-premiers-pas-avec-cassandra.html, 2010.

[5] Michaël Figuière. Nosql europe : Bases de données orientées graphe et neo4j. http://blog.xebia.fr/2010/05/03/nosql-europe-bases-de-donnees-graphe-et-neo4j/, 3 mai 2010.

[6] Michaël Figuière. Nosql europe : Bases de données clé-valeur et riak. http://blog.xebia.fr/2010/04/26/nosql-europe-bases-de-donnees-cle-valeur-et-riak/, 26 avril 2010.

[7] Michaël Figuière. Nosql europe : Bases de données orientées documents et mongodb. http://blog.xebia.fr/2010/04/30/nosql-europe-bases-de-donnees-orientees-documents-et-mongodb/, 30 avril 2010.

[8] Robert Sheldon. Sql server index basics. http://www.simple-talk.com/sql/learn-sql-server/sql-server-index-basics/, 25 November 2008.

[9] Dries Buytaert. Nosql and sql. http://buytaert.net/nosql-and-sql, 4 Decembre 2009 - 13 :05.

[10] Ted Dziuba. I can't wait for nosql to die. http://teddziuba.com/2010/03/i-cant-wait-for-nosql-to-die.html, Jeudi 4 mars 2010.

[11] Rick Cattell. Scalable sql and nosql data stores. *SIGMOD Record*, December 2010 (Vol. 39, No. 4).

[12] Peter Membrey Eelco Plugge and Tim Hawkins. *The Definitive Guide to MongoDB*. Apress, 2010.

[13] SHEN Shu GU Yunhua and ZHENG Guansheng. Application of nosql database in web crawling. *International Journal of Digital Content Technology and its Applications*, Volume 5, Number 6, June 2011.

[14] Gavin Terrill. Neo4j - an embedded, network database. http://www.infoq.com/news/2008/06/neo4j/, 16 février 2010.

[15] The top 10 ways to get to know neo4j. *Neo4j Blog* http://blog.neo4j.org/2010/02/top-10-ways-to-get-to-know-neo4j.html, 16 février 2010.

[16] Cours de bases de données relationnelles. www.cmi.univ-mrs.fr/~campioni/documents/BD/BD-relationnelles.pdf.

[17] Daniel Bartholomew. Sql vs. nosql. http://www.linuxjournal.com/article/10770, 1er septembre 2010. Article apparu dans Linux Journal, Magazine mensuel spécialisé dans GNU/-Linux et les logiciels libres.

[18] Michael Stonebraker. Newsql vs nosql for new oltp, michael stonebraker, voltdb. http://www.slideshare.net/Dataversity/newsql-vs-nosql-for-new-oltp-michael-stonebraker-voltdb, 24 Août 2011. Conférence de Michael Stonebraker, pionnier des SGBD et directeur technique chez VoltDB, sur sa solution NewSQL.

[19] Site officiel de MySQL. http://www.mysql.com/why-mysql/marketshare/. Visité le 29 février 2012.

[20] Brad Arrington. Mysql is not acid compliant. http://www.remotedbaexperts.com/Blog/2010/11/mysql-is-not-acid-compliant/, 22 novembre 2010. Exemple d'exécution de MySQL avec des résultats prouvant qu'il n'est pas ACID.

[21] Site officiel de Memcached. http://memcached.org/about, visité le 25 février 2012.

[22] Blog officiel de MongoDB. http://blog.mongodb.org/post/434865639/state-of-mongodb-march-2010, mars 2010. visité le 25 février 2012.

[23] Site officiel de 10gen. http://www.10gen.com/customers, visité le 25 février 2012. Donne des explications sur les motivations de ses différents clients.

[24] Wiki officiel de la fondation Apache. http://wiki.apache.org/cassandra/CassandraUsers, 18 décembre 2010. visité le 25 février 2012.

[25] Site officiel de l'entreprise neotechnology. http://neotechnology.com/customers/. visité le 25 février 2012.

[26] Serge Leblal avec IDG NS. Newsql pour combiner le meilleur de sql et nosql. http://www.lemondeinformatique.fr/actualites/lire-newsql-pour-combiner-le-meilleur-de-sql-et-nosql-34475-page-3.html, 25 août 2011. visité le 1er mars 2012.

[27] Neal Leavitt. Will nosql databases live up to their promise ? *IEEE Computer Society 0018-9162*, 26 janvier 2010.

[28] Michael Stonebraker. Errors in database systems, eventual consistency, and the cap theorem. http://cacm.acm.org/blogs/blog-cacm/83396-errors-in-database-systems-eventual-consistency-and-the-cap-theorem/fulltext, 5 avril 2010.

[29] Site officiel de MySQL. http://www.mysql.com/customers/. Visité le 05 mars 2012.

[30] Documentation officielle de mongoDB. www.mongodb.org/display/DOCS/. Visité le 05 mars 2012.

[31] Site du framework spring. http://www.springsource.org/features/data-access, visité le 7 mars 2012.

[32] Site officiel de MongoDB. http://www.mongodb.org/display/DOCS/SQL+to+Mongo+Mapping+Chart, visité le 7 mars 2012.

Première partie

Annexes

ANNEXE A

LES PROPRIÉTÉS ACID

Les propriétés ACID permettent à un SGBD d'effectuer des transactions. Par transaction, il faut comprendre une suite d'opérations qui font passer la BDD d'un état antérieur à un état postérieur. Les états intermédiaires entre les états avant la transaction et après la transaction ne sont pas visibles. Ses propriétés sont les suivantes :

Atomicité : la suite d'opérations constituant une transaction est indivisible. La transaction est entièrement effectuée ou pas du tout. Il y a annulation de toute la transaction lorsqu'une des opérations échoue. S'il est question de modifier une série de valeurs et qu'une modification échoue alors toutes les valeurs déjà modifiées reprennent leurs anciennes valeurs.

Cohérence : quelque soit l'opération effectuée, la base doit garder un état cohérent. Toute transaction qui viole par exemple une règle d'intégrité échoue. Après la fusion de deux tables, les entrées doivent toutes avoir des identités différentes. Si ce n'est pas le cas alors la fusion n'est pas effectuée.

Isolation : chaque transaction est isolée de sorte à ce qu'elle est seule peut voir les modifications pendant son exécution. Toute transaction enclenchée en parallèle d'une autre voit la version des données antérieure à celle-ci. Il existe 4 niveaux d'isolation définis dans le standard ANSI/ISO SQL :

1. Uncommited read ou lectures des données non validées. Ce niveau est le niveau d'isolation le plus léger. Avec un tel niveau d'isolation le système se comporte comme s'il n'y en avait pas. Les modifications apportées par une transaction non validée sont visibles.
2. Commited read ou lecture des seules données validées. Les modifications lors d'une transaction ne sont visibles que lorsqu'elle termine. Cependant lors d'une transaction une donnée peut changer sans que la transaction en cours en soit responsable. Ceci peut arriver, par exemple, lorsqu'une transaction en parallèle termine et que ses modifications sont validées.
3. Repeatable read ou lecture répétée. Ce niveau fonctionne comme le niveau précédent à la seule différence que durant son exécution, une transaction ne voit pas les mises à jour effectuées par d'éventuelles transactions qui se sont exécutées en parallèle. D'où « repeatable read » pour souligner que pendant une transaction, une donnée aura toujours la même valeur en lecture si elle n'est pas modifiée par la transaction elle-même. Cependant tout nouveau rajout validé de données au système par une transaction qui a terminé est visible par toute autre transaction en cours.

4. Serializable ou sérialisable. Ce niveau est le niveau d'isolation le plus poussé. Le système se comporte comme s'il n'y avait qu'une transaction à la fois. Pendant son exécution une transaction ne voit ni les mises à jour, ni les rajouts de données au système des autres transactions. D'où « serializable » pour mettre en relief le caractère d'exécution en série des transactions plutôt qu'en parallèle.

Durabilité : dès lors qu'une transaction est validée, aucune défaillance du système ne pourra conduire à l'annulation de celle-ci. Les modifications liées à une transaction validée perdurent et ne sont jamais remises en cause.

ANNEXE B

ILLUSTRATION DES REPRÉSENTATIONS NOSQL

ANNEXE C

LE THÉORÈME DE BREWER OU LE THÉORÈME CAP

Jusqu'en 2002, le théorème n'était qu'une conjecture de énoncée par Brewer.

Théorème C.0.1 (de Brewer) *Dans un environnement distribué, un système ne peut être doté qu'au plus de deux des trois propriétés suivantes :*

- *Cohérence (Consistency)*
- *Disponibilité (Availability)*
- *Résistance au morcellement (Partition Tolerance)*

Cohérence : toute opération effectuée sur un nœud est automatiquement visible sur tous les autres nœuds du système. Si j'écris sur un noeud et lis sur un autre, je dois Disponibilité : Résistance au morcellement :