NoSQL storage and management of geospatial data with emphasis on serving geospatial data using standard geospatial web services

Pouria Amirian, Adam Winstanley, Anahid Basiri

Department of Computer Science, National University of Ireland Maynooth
Tel. (+353) 1 7083853 Fax (+353) 1 7083848
amirian@cs.nuim.ie, adam.winstanley@nuim.ie, anahid.basiri@nuim.ie

KEYWORDS: BIG GEOSPATIAL DATA, NOSQL, GEOSPATIAL WEB SERVICE

1. Introduction

Today a huge amount of geospatial data is being created, collected and used more than ever before. The ever increasing observations and measurements of geo-sensor networks, satellite imageries, point clouds from laser scanning, geospatial data of Location Based Services (LBS) and location-based social networks has become a serious challenge for data management and analysis systems. Traditionally, Relational Database Management Systems (RDBMS) were used to manage and to some extent analyse the geospatial data. Nowadays these systems can be used in many scenarios but there are some situations when using these systems may not provide the required efficiency and effectiveness. In these situations, NoSQL solutions can provide the efficiency necessary for applications using geospatial data. It is important to differentiate between the physical way a NoSQL product is implemented, and the interfaces, coding and access methods they use for the abstraction of data. This paper provides an overview of the major types of NoSQL solutions, their advantages and disadvantages and the challenges they present in managing geospatial data. Then the paper elaborates on serving geospatial data using standard geospatial web services with a NoSQL database as a backend.

2. Why Use a NoSQL Database for Management of Geospatial Data?

It has been more than thirty years since relational DBMSs became the major solution for storing all kinds of data in many types of applications. They manage data using relational algebra and relational calculus as their theoretical foundation. They use tables, relationships, keys and Structured Query Language (SQL) to perform all sorts of functions with data. They usually are the best solutions when the schema of data is fixed and predefined. In other words, RDBMSs are ideal solutions to managing structured data.

RDBMS can be effectively used in many common GIS workflows. Since they support transaction and locking features, they provide robust backends for enterprise GIS systems. Usually geospatial data have a fixed schema and in most cases they are not used in isolation. That is a join of two or more datasets and connecting data through spatial operations is needed in most GIS workflows. For this reason managing fixed schema geospatial data and using them in GIS workflows can usually be done effectively through RDBMS systems.

However, today unstructured and semi-structured data are becoming mainstream. Observation and measurements from diverse kinds of sensors, geospatial data of Location Based Services (LBS) and social networks are three examples of semi-structured and unstructured data in the geospatial world. In addition, there are also some methods that generate massive geospatial data, for example remote sensing (RS) and laser scanning. In

some situations when high availability and scalability is needed, RDBMSs provide very expensive solutions for managing such data even if it has a fixed schema. For example, in an Automatic Vehicle Location (AVL) application, an RDBMS cannot easily handle the possibly massive read and write operations. In these situations NoSQL databases can provide a highly accessible and scalable way of managing semi-structured or unstructured geospatial data efficiently.

In addition, some problems arise with RDBMS systems when scalability is needed by adding more servers and technologies to bind them together. In addition with more load on the system, vertical partitioning, denormalization and removal of the relational constraints comes into play. In summary, to achieve high scalability in RDBMS systems the normalized relational model of data storage has to be compromised and deviated from relational model.

3. What is a NoSQL Database?

The NoSQL DBMSs are a broad class of DBMSs identified by non-adherence to the RDBMS model. In general, the most significant difference is the lack of SQL. There are different types of NoSQL databases, each with distinct set of characteristics but they all can deal with large amounts of (semi-structured and unstructured) data and are able to support a large set of read and write operations. For this reason NoSQL and relational models are not in contrast with each other rather they complement each other. The most widely accepted taxonomy of NoSQL databases are: key-value, document, tabular or columnar and graph.

The key-value database is the simplest type of NoSQL databases. As the name implies, this type of database stores schema-less data using keys. The key is usually a string and the stored values can be any valid type such as a primitive programming data type (string, integer etc) or a BLOB (Binary Large Object) without any predefined schema. It provides a simple API to access stored data (Fowler and Sadalage). In most cases, this type of NoSQL database solution provides very little functionality beyond key-value storage. There is no support for relationships (Fowler and Sadalage). In terms of concurrency, they usually provide eventual consistency but don't provide optimistic concurrency especially in highly scalable environments. Queries are just limited to accessing values using keys but since there is one request to access the value, the queries are executed quickly. Transactions are limited to a single key. The database contains no semantic model. In other words, the client is in charge of interpreting and understanding values.

Key-value databases can be utilized to store geospatial data but their complexity hinders spatial searches especially for polylines and polygons. For this reason, it needs to be spatially indexed for fast data retrieval which, in most cases, gives lower performance than a relational database. Many researchers and developers recommend using indexing techniques such as grids or tiles (Schwartz), quadkeys, space filling curves (Fuller) and similar approaches. In summary, key-value data stores are ideal for inserting, deleting and searching huge amount of data items using their unique identifiers (keys). If spatial searches are needed however they are not good solutions.

A document database in its simplest form is a key-value database in which the database understands its values (Hecht and Jablonski). In other words, values inside the database are based on predefined formats such as XML, JSON or BSON (Binary JSON). This feature of document databases provides many advantages over key-value databases. Instead of putting too much logic in the application tier, many operations can be done by the database itself. Queries in this type of NoSQL databases are quite flexible and some document databases

even have their own query languages. Similar to key-value databases, there is no need to adhere to a predefined schema to insert data. There is only limited support for relationships and joins as each document is stand alone. However, more concurrency options, such as optimistic concurrency and eventual consistency are available (Tiwari). Transaction integrity is supported for one document or document fragment.

The document databases can be used for managing geospatial data more effectively than key-value databases. Since geospatial data inside the document database can be retrieved using flexible queries, they can be used for storing and managing geospatial data in multiple use cases. In fact many document databases support geospatial data natively or through extensions (Schutzberg). Some applications of LBS such as proximity queries can be efficiently implemented using these document databases. As mentioned before, relationships and joins are not supported the way they are supported in relational databases. Often in common GIS workflows relationships and joins have to be used. Special kinds of document database can partially handle relationships. Native XML databases can be configured to enforce adherence to set of predefined XML schemas. In addition to all the advantages of document databases, native XML databases are able to utilize many XML technologies to provide further functionality. For example, they can use XQuery and XPath to perform various queries and create flexible result sets, they can make use of XPointer to reference other documents thus modelling a relationship and they can use XSD and RelaxNG to enforce schema validation. Geography Markup Language (GML), as a standard mechanism for storing, modelling and exchanging geospatial data, is an XML-based grammar and so Native XML databases are an ideal choice for managing geospatial data in GML format.

The tabular or columnar (or column family) database stores data in columns. This is in contrast to the row-oriented format in relational databases (Tiwari). The column is the smallest unit of data and it is a triplet that contains a key, value and timestamp (Hecht and Jablonski). Columnar databases store all values beside the name of the columns and stores null values simply by ignoring the column. Usually, related columns compose a column-family. All the data in a single column family will be stored on the same physical set of files (Xiang, Hou and Zhou). This feature provides higher performance for search, data retrieval and replication operations. A super column is a column that contains other columns but it cannot contain other super columns (Tiwari). Most columnar databases use a distributed file-system to store data to disk and so provide a horizontally scalable system. In fact columnar databases are designed to run on a large number of machines. Queries in this type of NoSQL databases are limited to keys and in most cases they don't provide a way to query by column or value. By limiting queries to just keys, columnar databases ensure that procedure to find the machine containing actual data is quite fast. There is no join capability and, as in other types of NoSQL databases, there is limited support for transactions.

Columnar databases are ideal for storing huge amounts of data when high availability is needed. Similar to document databases, there are many columnar databases which support the management and simple analysis of geospatial data. Any GIS related application which needs heavy data insertion and fast data retrieval with simple queries can efficiently make use of columnar databases. As an example, an Automatic Vehicle Location (AVL) application which needs to track the location of many vehicles simultaneously can store the incoming data from vehicles in a columnar database and respond to queries efficiently.

As the name implies graph databases are based on graph theory and employ nodes, properties and edges as their building blocks. In the graph databases various nodes might have different properties. The graph databases are well suited for data which can be modelled as networks

such as road networks, social networks, biological networks and semantic webs. Their main feature is the fact that each node contains a direct pointer to its adjacent node and no index lookups are necessary for traversing data. As a result they can manage huge amount of data as long as the data do not require expensive join operations. Some of graph databases support transactions in the way that relational databases support them (Perdue). In other words the graph database allows the update of a section of the graph in an isolated environment, hiding changes from other processes until the transaction is committed.

Geospatial data can be modelled as graphs. Since graph databases support topology natively, topological relationship between geospatial data can be easily managed by this type of NoSQL databases. In most GIS workflows, topological relationships play a major role. In addition, since graph databases are ideal for managing data with evolving schema, they can be effectively used in Volunteered Geographic Information (VGI) and crowd sourcing applications. Also, graph databases are the best choice for managing huge linear networks (such as roads) and for routing and navigation applications.

4. Using an SQL Database in a NoSQL Manner

In an inter-organizational and intra-organizational business intelligence environment with many software infrastructures, data models, data exchange patterns and policies, interoperability is a major concern. It is common practice to resort to Service Oriented Architecture (SOA) and Service Orientation (SO) principles to provide the interoperability needed for all components participating in every work process that has potential for reusability. These services are often implemented using web service technologies. In geospatial workflows, reusable services are usually implemented based on Open Geospatial Consortium (OGC) specifications.

However, geospatial services based on OGC specification and web services are not the same (Amirian, Alesheikh and basiri, Sample, Shaw, Tu and Abdelgurfi) but both are in line with the idea of SOA. As part of an enterprise architecture and closely related to the quality of service parameters of SOA, the most effective and efficient solution to manage data must be selected and implemented. In many situations, NoSQL databases are ideal solutions which provide high availability and horizontal scalability with less effort and complexity. However, the most important barrier to adapting a NoSQL solution is the lack of organisational agility. In a survey conducted by Information Week, 44% of business IT professionals had not heard of NoSQL (Babcock).

In other words, many organizations for different reasons are not agile enough to change their software infrastructure and switch to NoSQL databases. In this situation, one of the best strategies to gradually adapt NoSQL solutions is to use RDBMSs in a NoSQL fashion. This means using tables to store data, denormalize and vertical partitioning of data inside tables.

6. Acknowledgements

Research presented in this paper was funded by a Strategic Research Cluster grant (07/SRC/I1168) by Science Foundation Ireland under the National Development Plan. The authors gratefully acknowledge this support.

References

Fowler, M. and Sadalage, P. 2012. NoSQL distilled: a brief guide to the emerging world of polyglot persistence. Addison-Wesley.

Schwartz, J. 2009. Tiles of BingMap. http://msdn.microsoft.com/en-us/library/bb259689.aspx

Fuller, A. 2010, Space filling curves. http://www.google.com/intl/ga/events/io/2010/sessions/next-gen-queries-appengine.html

Hecht, R. and Jablonski. S. 2011. NoSQL Evaluation A Use Case Oriented Survey. International Conference on Cloud and Service Computing, 336-341.

Tiwari, S. 2011. Professional NoSQL. Wrox publication.

Schutzberg, A. 2011. NoSQL databases: What Geospatial Users Need to Know.

Chang, F., Dean, J., Ghemawat, S., Hsieh, W. and Gruber. R. 2006. Bigtable: A distributed storage system for structured data. seventh symposium on operating system design and implementation.

Xiang, P., Hou, R. and Zhou, Z. 2010. Cache and consistency in NoSQL. 3rd IEEE international conference on computer science and information technology, 117-120.

Amirian, P., Alesheikh, A. and Basiri, A. 2010. Standard-based, interoperable services for accessing urban services data, Computer Environment and Urban Systems, 34 (4), 309-321.

Sample, J., Shaw, K., Tu , S. and Abdelguerfi, M. 2008. Geospatial Services and Applications for the Internet, SpringerScience.

Perdue, T. 2011. http://newtech.about.com/od/databasemanagement/a/Nosql.htm

Babcock, C. 2010. http://www.informationweek.co.uk/software/information-management/surprise-44-of-business-it-pros-never-he/227500077?queryText=nosql+survey

Biography

Pouria Amirian *holds a PhD of Geospatial Information Systems (GIS). He is a PostDoctoral Research Fellow in the Department of Computer Science at National University of Ireland Maynooth on the Strategic Research in Advanced Geotechnologies (StratAG) project. His areas of interest includes spatial interoperability, spatial databases, NoSQL, Service Oriented Architecture and Web Services Technologies.*

Adam Winstanley *is Head of Computer Science at NUI Maynooth and co-PI on StratAG. He is also a Senior Research Associate in the National Centre for Geocomputation in Maynooth*

Anahid Basiri *is a PostDoctoral Research Fellow in the Department of Computer Science, NUI Maynooth. Her current research interests include navigation services, uncertainty in GIS and spatio-temporal objects.*