International Journal of Enterprise Computing and Business Systems
ISSN (Online) : 2230-8849
Volume 2 Issue 2 July 2013
International Manuscript ID : ISSN22308849-V2I2M3-072013

NOSQL - NOTONLY SQL

Dr. S. George
University of New Zealand

Abstract

A NoSQL database provides a mechanism for storage and retrieval of data that uses looser consistency models than traditional relational databases. Motivations for this approach include simplicity of design, horizontal scaling and finer control over availability. NoSQL databases are often highly optimized key–value stores intended for simple retrieval and appending operations, with the goal being significant performance benefits in terms of latency and throughput. NoSQL databases are finding significant and growing industry use in big data and real-time web applications. NoSQL systems are also referred to as "Not only SQL" to emphasize that they do in fact allow SQL-like query languages to be used.

ACID vs BASE NoSQL cannot necessarily give full ACID guarantees. Usually eventual consistency is guaranteed or transactions limited to single data items. This means that given a sufficiently long period of time over which no changes are sent, all updates can be expected to propagate eventually through the system.[*citation needed*].

Contents

History

Carlo Strozzi used the term *NoSQL* in 1998 to name his lightweight, open-source relational database that did not expose the standard SQL interface. Strozzi suggests that, as the current NoSQL movement "departs from the relational model altogether; it should therefore have been called more appropriately 'NoREL'.

Eric Evans (then a Rackspace employee) reintroduced the term *NoSQL* in early 2009 when Johan Oskarsson of Last.fm wanted to organize an event to discuss open-source distributed databases. The name attempted to label the emergence of a growing number of non-relational, distributed data stores that often did not attempt to provide atomicity, consistency, isolation and durability guarantees that are key attributes of classic relational database systems.

Taxonomy

There have been various approaches to classify NoSQL databases, each with different categories and subcategories. Because of the variety of approaches and overlappings regarding the nonfunctional requirements and the feature-set it could be difficult to get and maintain an overview of the nonrelational database market. Nevertheless, the most basic classification that most would agree is one based on the data model. A few of these and their prototypes are:

- **Column**: Hbase, Accumulo
- **Document**: MongoDB, Couchbase
- **Key-value** : Dynamo, Riak, Redis, Cache, Project Voldemort
- **Graph**: Neo4J, Allegro, Virtuoso

Classification based on data model

Stephen Yen in his blog post "A yes for NoSQL taxonomy" suggests the following:[citation needed]

Term	Matching Database
KV Store	keyspace Flare SchemaFree RAMCloud Oracle NoSQL Database(OnDB)
KV Store - Eventually	Dynamo Voldemort Dynomite SubRecord Mo8onDb Dovetaildb

Term	Matching Database
consistent	
KV Store - Ordered	TokyoTyrant Lightcloud NMDB Luxio MemcacheDB Actord
KV Cache	Memcached Repcached Coherence Infinispan EXtremeScale JBossCache Velocity Terracoqua
Tuple Store	Gigaspaces Coord ApacheRiver
Object Database	ZopeDB DB40 Shoal
Document Store	CouchDB Couchbase MongoDB Jackrabbit XML-Databases ThruDB CloudKit Prsevere Riak-Basho Scalaris
Wide Columnar Store	Bigtable Hbase Cassandra Hypertable KAI OpenNeptune Qbase KDI

Classification based on feature

Ben Scofield categorized NoSQL databases based on nonfunctional categories ("(il)ities") plus a rating of their feature coverage:[citation needed]

Data Model	Performance	Scalability	Flexibility	Complexity	Functionality
Key-Value Stores	high	high	high	none	variable (none)
Column Store	high	high	moderate	low	minimal
Document Store	high	variable (high)	high	low	variable (low)
Graph Database	variable	variable	high	high	graph theory
Relational Database	variable	variable	low	moderate	relational algebra.

It has been suggested that this article be merged into *Comparison of structured storage software*. (Discuss) *Proposed since March 2011.*

Examples

Document store

Main articles: Document-oriented database and XML database

The central concept of a document store is the notion of a "document". While each document-oriented database implementation differs on the details of this definition, in general, they all assume that documents encapsulate and encode data (or information) in some standard formats or encodings. Encodings in use include XML, YAML, and JSON as well as binary forms like BSON, PDF and Microsoft Office documents (MS Word, Excel, and so on).

Different implementations offer different ways of organizing and/or grouping documents:

- Collections
- Tags
- Non-visible Metadata
- Directory hierarchies

Compared to relational databases, for example, collections could be considered as tables as well as documents could be considered as records. But they are different: every record in a table has the same sequence of fields, while documents in a collection may have fields that are completely different.

Documents are addressed in the database via a unique **key** that represents that document. One of the other defining characteristics of a document-oriented database is that, beyond the simple key-document (or key–value) lookup that you can use to retrieve a document, the database will offer an API or query language that will allow retrieval of documents based on their contents. Some NoSQL document stores offer an alternative way to retrieve information using MapReduce techniques, in CouchDB the usage of MapReduce is mandatory if you want to retrieve documents based on the contents, this is called "Views" and it's an indexed collection with the results of the MapReduce algorithms.

Name	**Language**	**Notes**
BaseX	Java, XQuery	XML database
Cloudant	Erlang, Java, Scala, C	JSON store (online service)
Clusterpoint	C++	XML, geared for Full text search
Couchbase Server	Erlang, C, C++	Support for JSON and binary

Name	Language	Notes
		documents
Apache CouchDB	Erlang	JSON database
djondb	C++	JSON, ACID Document Store
ElasticSearch	Java	JSON, Search Engine
eXist	Java, XQuery	XML database
Jackrabbit	Java	Java Content Repository implementation
IBM Lotus Notes and Lotus Domino	LotusScript, Java, IBM X Pages, others	MultiValue
MarkLogic Server	XQuery, Java, REST	XML database with support for JSON, text, and binaries
MongoDB	C++, C#, Go	BSON store (binary format JSON)
Oracle NoSQL Database	Java, C	
OrientDB	Java	JSON, SQL support
Sedna	XQuery, C++	XML database
SimpleDB	Erlang	online service
OpenLink Virtuoso	C++, C#, Java, SPARQL	middleware and database engine hybrid

Graph

This kind of database is designed for data whose relations are well represented as a graph (elements interconnected with an undetermined number of relations between them). The kind of data could be social relations, public transport links, road maps or network topologies, for example.

Main article: Graph database

Name	Language	Notes
AllegroGraph	SPARQL	RDF GraphStore

Name	**Language**	**Notes**
IBM DB2	SPARQL	RDF GraphStore added in DB2 10
DEX	Java, C++, .NET	High-performance Graph Database
FlockDB	Scala	
InfiniteGraph	Java	High-performance, scalable, distributed Graph Database
Neo4j	Java	
OpenLink Virtuoso	C++, C#, Java, SPARQL	middleware and database engine hybrid
OrientDB	Java	
Sones GraphDB	C#	
OWLIM	Java, SPARQL 1.1	RDF graph store with reasoning
VelocityGraph	C#	Fully Tinkerpop Blueprints compliant. Scalable hybrid Object Database and Graph Database

Key–value stores

Key–value stores allow the application to store its data in a schema-less way. The data could be stored in a datatype of a programming language or an object. Because of this, there is no need for a fixed data model. The following types exist:

KV - eventually consistent

- Apache Cassandra
- Dynamo
- Hibari
- OpenLink Virtuoso
- Project Voldemort
- Riak

KV - hierarchical

- GT.M
- InterSystems Caché

KV - cache in RAM

- memcached
- OpenLink Virtuoso
- Oracle Coherence
- Redis
- Tuple space
- Velocity
- IBM WebSphere eXtreme Scale
- JBoss Infinispan

KV - solid state or rotating disk

- Aerospike
- BigTable
- CDB
- Couchbase Server
- Keyspace
- LevelDB
- MemcacheDB (using Berkeley DB)
- MongoDB
- OpenLink Virtuoso
- Tarantool
- Tokyo Cabinet
- Tuple space
- Oracle NoSQL Database

KV - ordered

- Berkeley DB
- FoundationDB
- IBM Informix C-ISAM

- InfinityDB
- MemcacheDB
- NDBM

Object database

Main article: Object database

- db4o
- GemStone/S
- InterSystems Caché
- JADE
- NeoDatis ODB
- ObjectDB
- Objectivity/DB
- ObjectStore
- ODABA
- OpenLink Virtuoso
- Versant Object Database
- WakandaDB
- ZODB

Tabular

- Apache Accumulo
- BigTable
- Apache Hbase
- Hypertable
- Mnesia
- OpenLink Virtuoso

Tuple store

- Apache River
- OpenLink Virtuoso

- Tarantool

Triple/Quad Store (RDF) database

- Meronymy SPARQL Database Server
- Virtuoso Universal Server
- Ontotext-OWLIM
- Apache JENA
- Oracle NoSQL database

Hosted

- Freebase
- OpenLink Virtuoso
- Datastore on Google Appengine
- Amazon DynamoDB
- Cloudant Data Layer (CouchDB)

Multivalue databases

- Northgate Information Solutions Reality, the original Pick/MV Database
- Extensible Storage Engine (ESE/NT)
- OpenQM
- Revelation Software's OpenInsight
- Rocket U2
- D3 Pick database
- InterSystems Caché
- InfinityDB

Cell database

- Boardwalk

References

1. Lith, Adam; Jakob Mattson (2010). "Investigating storage solutions for large data: A comparison of well performing and scalable data storage solutions for real time extraction and batch insertion of data" (PDF). Göteborg: Department of Computer Science and Engineering, Chalmers University of Technology. p. 70. Retrieved 12 May 2011. "Carlo Strozzi first used the term NoSQL in 1998 as a name for his open source relational database that did not offer a SQL interface[...]"
2. "NoSQL Relational Database Management System: Home Page". Strozzi.it. 2 October 2007. Retrieved 29 March 2010.
3. "NoSQL 2009". Blog.sym-link.com. 12 May 2009. Retrieved 29 March 2010.
4. Mike Chapple. "The ACID Model".
5. http://djondb.com
6. http://tinman.cs.gsu.edu/~raj/8711/sp13/djondb/Report.pdf
7. http://undefvoid.blogspot.com/2013/03/meeting-with-djondb.html
8. Sandy (14 January 2011). "Key Value stores and the NoSQL movement". http://dba.stackexchange.com/questions/607/what-is-a-key-value-store-database: Stackexchange. Retrieved 1 January 2012. "Key–value stores allow the application developer to store schema-less data. This data usually consists of a string that represents the key, and the actual data that is considered to be the value in the "key–value" relationship. The data itself is usually some kind of primitive of the programming language (a string, an integer, or an array) or an object that is being marshaled by the programming language's bindings to the key–value store. This structure replaces the need for a fixed data model and allows proper formatting."
9. Marc Seeger (21 September 2009). "Key-Value Stores: a practical overview". http://blog.marc-seeger.de/2009/09/21/key-value-stores-a-practical-overview/: Marc Seeger. Retrieved 1 January 2012. "Key–value stores provide a high-performance alternative to relational database systems with respect to storing and accessing data. This paper provides a short overview of some of the currently available key–value stores and their interface to the Ruby programming language."
10. "Riak: An Open Source Scalable Data Store". 28 November 2010. Retrieved 28 November 2010.
11. Tweed, Rob; George James (2010). "A Universal NoSQL Engine, Using a Tried and Tested Technology" (PDF). p. 25. "Without exception, the most successful and well-known of the NoSQL databases have been developed from scratch, all

within just the last few years. Strangely, it seems that nobody looked around to see whether there were any existing, successfully implemented database technologies that could have provided a sound foundation for meeting Web-scale demands. Had they done so, they might have discovered two products, GT.M and Caché.....*"

12. JBoss Infinispan

13. https://www.google.com/patents/US7383272?dq=Method+and+system+for+versioned+sharing,+consolidation+and+reporting+information&hl=en&sa=X&ei=wmUFUuyKBsv0iwLmvoCgBA&ved=0CDQQ6AEwAA