Distributed Databases: SQL vs NoSQL

Seda Unal, Yuchen Zheng
April 23, 2017

1 Introduction

Distributed databases have become increasingly popular in the era of big data because of their advantages over traditional databases. In this project, distributed databases are investigated from a relational versus non-relational perspective introducing SQL and NoSQL and discussing their advantages and disadvantages for distributed databases.

A distributed database is a database where portions of the database are stored in multiple physical locations and processing is distributed among multiple database nodes. SQL, Structured Query Language, is the standard language for relational database management systems. In a relational database, relations/tables are used to match data and the resulting group is called a Schema. The need to store, process and analyze the unstructured data led to the development of schema-less alternatives to SQL, namely NoSQL known as not only SQL.

In this project, a thorough comparison between SQL and NoSQL for distributed databases is provided on many aspects from model features, data integrity and flexibility to scalability.

2 Distributed Databases

A distributed database is a collection of multiple, logically related databases distributed over a computer network [1]. It has advantages such as improved performance, speed and resource efficiency over a traditional centralized database system. However, it also has disadvantages such as increased complexity and difficulty to maintain data integrity.

A distributed database system allows applications to access data from local and remote databases. With the data stored on multiple computers, users can access and modify the data in a network simultaneously. DDBMS (distributed database management system) control all database servers and maintain the consistency of the global database in a distributed database. Distributed databases use a client-server-node architecture to process information requests, in which client is an application that requests information from a
server, a server is the software that manages a database, and a node in a distributed database system can be either a client or a server.

In the recent years preference has gradually moved to distributed database systems with motivations like the distributed nature of organizational units, support for both OLTP (online transaction processing) and OLAP (online analytical processing) and database recovery [2, 3]. Distributed databases have advantages over centralized databases like reliability, higher speed, low communication cost. Minimal disruption on database from failures indicates a distributed database is much reliable as the entire system will not come to a halt because of minor mistakes. Data stored on multiple computers, which does not require sending everything to a central computer to process, guarantees faster responses for users. Lastly, data in distributed databases is located locally where it is mostly used, which minimizes the communication costs for data manipulation [3, 4].

3 SQL vs NoSQL in Distributed Databases

SQL and NoSQL are both great inventions to keep data storage and retrieval optimized and smooth with one having some advantages over the other in certain scenarios [5, 8, 7].

Consider a social platform where users engage with each other in terms of posts associated with images, audio, video, comments, links to websites, etc. Using SQL a query with many joins is necessary to retrieve the content. In addition to the complexity of the data, consider stream of posts dynamically loading. With SQL, many queries and joins will be necessary to complete the task.

A solution to this problem can be using JSON, as it is the supported dynamic data format. Another approach, from a nonrelational perspective would be using NoSQL. NoSQL simplifies the approach for this specific scenario.

Due to its simplicity, the use of NoSQL has grown with the social media platforms in order to successfully handle the growing need of IoT (Internet of Things).

3.1 SQL

SQL (Structured Query Language) is a programming language widely used to manage data in relational databases. A relational database is a set of tables containing data fitted into predefined categories. Users can access data from the database without knowing the structure of the database table.

Scalability in relational SQL databases is an important issue since the database has to be distributed on to multiple servers and handling tables across different servers can be problematic. However, with Google announcing F1, a SQL database that is trivial to scale up, to run at the core of AdWords business; SQL is guaranteed to be available for distributed databases [8].
3.2 NoSQL

Non-relational databases have been around since the late 1960s, but the major shift to NoSQL databases happened only in the last decade, when Amazon introduced its Dynamo distributed NoSQL system [2].

A NoSQL database provides a mechanism for storage and retrieval of data which is modeled in means other than the tabular relations used in relational databases.

3.2.1 NoSQL Databases

There has been a booming of all kinds of data in recent years, a big portion of which is no longer structured data which SQL is good at handling. NoSQL is exactly the tool to deal with these unstructured and semi-structured data.

A NoSQL database environment is, simply put, a non-relational and largely distributed database system that enables rapid, ad-hoc organization and analysis of extremely high-volume, disparate data types [9].

Schema-less NoSQL databases follow basically all advantages of distributed databases and have become the alternative to relational databases. What does it mean by saying NoSQL is schema-less? It means for any data we put into NoSQL databases, we do not need to predefine a rigid schema like one of those in relational databases. The format of the data can even be changed at any time, without disruption on anything, which provides great application flexibility, and a faster speed than relational databases.

NoSQL also has great horizontal scalability. It automatically spreads data over many servers, which can be added or removed from the data layer. Another reason is that NoSQL does not follow ACID (atomicity, consistency, isolation, durability), which relational databases have. ACID makes it harder to have a good horizontal scalability.

Other features like replication (which maintains availability) and integrated caching (which improves read performance) also make NoSQL databases distinct from relational databases.

3.2.2 Types of NoSQL Databases

Among many types of NoSQL databases, there are four most common ones: Document database, which stores document-oriented information (semi-structured data); Column store, which stores data tables as sections of columns of data, rather than as rows of data; Key-value store, which stores data with an indexed key and a value; Graph database, which stores data whose relations are represented as a graph [9]. A brief comparison of these databases in in Figure 1 in terms of their performance, scalability, flexibility, complexity and functionality.

Let’s check an example here. We choose MongoDB (document), Cassandra (column), Riak (key-value), Neo4j (graph) from the types we just mentioned to see how they are similar or different from each other in Data distribution mechanism and Distributed data processing support.
Table 1: Comparison of Four Representative NoSQL Databases [10]

<table>
<thead>
<tr>
<th>Data distribution mechanism</th>
<th>MongoDB</th>
<th>Cassandra</th>
<th>Riak</th>
<th>Neo4j</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key-range and hash based distribution.</td>
<td>Key-range and hash based distribution</td>
<td>Hash based distribution</td>
<td>master-slave cluster</td>
<td></td>
</tr>
<tr>
<td>Distributed data processing support</td>
<td>Aggregation pipeline, MapReduce</td>
<td>Hadoop, MapReduce</td>
<td>MapReduce</td>
<td>Hadoop</td>
</tr>
</tbody>
</table>

3.3 NoSQL vs SQL

NoSQL supporting a nonrelational, flexible, dynamic and horizontally scalable databases is the frequently selected programming language for distributed database systems. SQL, on the other hand, enforces strict schema, structured data, strong consistency and vertical scalability. Depending on the needs, the best fit between SQL and NoSQL can be decided. In distributed systems, due to the scalability problems of SQL, mostly NoSQL is preferred. However, there is evidence that scalability in SQL can be achieved with clustered hierarchical distributed databases [8].

4 Conclusion

In the last decade, with the introduction of big data, IoT and increasing user activity in social platforms have forced the backbone in database technology to be distributive. Distributed databases have better reliability, higher speed and low communication cost compared to the traditional databases.

SQL and NoSQL are both great inventions to keep data storage and retrieval optimized and smooth. There are advantages and disadvantages of both programming languages in different applications. However, it would be fair to state that both are good and a better usage might depends on the situation and the needs.
References


[10] Xiaoming Gao, Investigation and Comparison of Distributed NoSQL Database Systems, Indiana University