# SQL vs NoSQL: A Performance Comparison

Ruihan Wang
University of Rochester
ruihan.wang@rochester.edu

Zongyan Yang
University of Rochester
zyang46@ur.rochester.edu

## Abstract

We always hear some statements like 'SQL is outdated', 'This is the world of NoSQL', 'SQL is still used a lot by most of companies.' Which one is accurate? Has NoSQL completely replace SQL? Or is NoSQL just a hype? SQL (Structured Query Language) is a standard query language for relational database management system. The most popular types of RDBMS(Relational Database Management Systems) like Oracle, MySQL, SQL Server, uses SQL as their standard database query language.[3] NoSQL means Not Only SQL, which is a collection of non-relational data storage systems. The important character of NoSQL is that it relaxes one or more of the ACID properties for a better performance in desired fields. Some of the NOSQL databases most companies using are Cassandra, CouchDB, Hadoop Hbase, MongoDB. In this paper, we'll outline the general differences between the SQL and NoSQL, discuss if Relational Database Management Systems is a thing of past, and also compare the speed performance of SQL and NoSQL databases, such as BerkeleyDB, MongoDB and MySQL.
**Keywords:** RDBMS, NoSQL, ACID properties, MongoDB, Oracle, Key-value stores,

## 1. Introduction

SQL stands for Structured Query Language, invented as a standard high-level interface for most of databases, usually used as DDL and DML for the management of relational database management system (RDBMS). Databases based on the relational model include MySQL, MS-SQL Server, Oracle database and so on, each of them supports SQL as the query language. NoSQL, which stands for Not Only SQL, however is a non-relational database management system. Leading NoSQL databases include MongoDB, Cassandra, CouchDB, HBase, etc. With the current popularity of "Big Data", NoSQL databases were pioneered and improved a lot by top internet companies like Amazon, Google and LinkedIn. The main difference between non-relational data model and the traditional one is, the non-relational model is designed for processing huge amount of data in a second, with relatively low consistency requirement. As a consequence, it relaxes the ACID constraints provided by many relational database systems, in exchange for the improvement of performance. For the discussion of our paper, we will narrow the SQL vs NoSQL topic down to the comparison and performance advantage of NoSQL over SQL based on models and theorems, together with the limitations that result from those factors.

## 2. ACID Properties and CAP Theorem

### 2.1. ACID Properties

We need to refer the ACID properties[12]:

**Atomicity**
A transaction is an atomic unit of processing; it should either be performed in its entirety or not performed at all.

**Consistency preservation**
A transaction should be consistency preserving, meaning that if it is completely executed from beginning to end without interference from other transactions, it should take the database from one consistent state to another.

**Isolation**
A transaction should appear as though it is being executed in iso- lation from other transactions, even though many transactions are execut- ing concurrently. That is, the execution of a transaction should not be interfered with by any other transactions executing concurrently.

**Durability or permanency**
The changes applied to the database by a com- mitted transaction must persist in the database. These changes must not be lost because of any failure.

### 2.2. CAP Theorem

For a distributed database, the CAP theorem states that it's impossible to simultaneously provide more than two out of the following three guarantees:

**Consistency**
Every read receives the most recent write or an error

**Availability**
Every request receives a (non-error) response – without guarantee that it contains the most recent write

**Partition tolerance**
The system continues to operate despite an arbitrary number of messages being dropped (or delayed) by the network between nodes

Based on CAP theorem, different database picks different combination of consistency, avalability, and partition tolerence:

- **CA**: Relational Database
- **CP, AP**: Non-Relational Database

## 3. NoSQL and Non-Relational Database

### 3.1. Clarification of NoSQL

The concept of NoSQL is not well defined and can refer to many different databases. Typically, there are four kinds of NoSQL databases, namely key-value stored, document stored, column stored and graph stored. We will give a brief introduction to each of them below.

| | SQL | NoSQL |
|---|---|---|
| Model | Relational. Store data in a table. | Non-relational. Store data in JSON Documents, key-value pairs, columns, or graphs. |
| Data | Every record has the same attribute | Offer flexibility that each record may have different attributes. |
| | Good for structured data | Good for semi-structured data |
| Schema | Fixed schema | Dynamic or flexible schema |
| Transactions | Support ACID transactions | Depends on different solution |
| Consistency & Availability | Strong consistency enforced, prioritized over availability and performance | Consistency, availability, and performance can be trade to meet the need of application (CAP therorem) |
| Performance | Insert and update performance depends on disk's IO speed | Performance can be maximized by reducing consistency |
| Scale | Typical scale Vertically | Typical scale Horizontally |

**Figure 1.** SQL versus NoSQL

| Key | Value |
|---|---|
| /test/10/float | 4.5 |
| /test/10/string | "Hello World" |

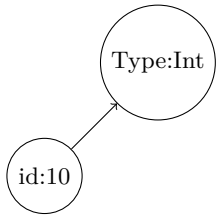
(a) key-value store

```
{
_id : 10
floatVal : 4.5
stringVal : 'Hello World'
}
```

(b) Document store

| Columns | Value |
|---|---|
| id | 10; ... |
| floatVal | 4.5; ... |
| stringVal | "Hello World"; ... |

(c) Column store




(d) Graph store

**Figure 2.** Data in different NoSQL databases

#### 3.1.1. Key-value Stored

Key-value database is the easiest one to understand among those four. The database manage system simply stores data in an unstructured dictionary (or hash table). The key in each records is the same as the primary key in SQL databases, while the value is an array of data. Typical key-value databases are BerkeleyDB, LevelDB and Redis.

#### 3.1.2. Document Stored

The main idea of document-oriented database is the notion of document. Although each database implementation differs on the definition of this, they are all assume the data are encapsulated in some standard formating. One of the most popular document-oriented database, MongoDB, uses the JSON format as its storage; while other databases like BaseX or YAMLDB use XML and YAML as their storage.

#### 3.1.3. Column Stored

A column-oriented database store tables by columns, instead of by rows. This difference is mainly handled by the database manage systems, meaning a column-oriented database is generally compatible with traditional row-based database; both of them can use SQL to load data and perform query. Some example of column-oriented databases are SAP HANA, Amazon Redshift and Sybase IQ.

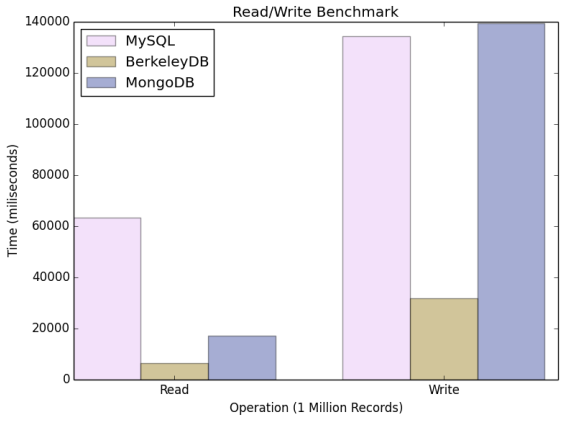



**Figure 3.** read/write time (less is better)

#### 3.1.4. Graph stored

A graph database is a database that store data with nodes, edges and properties. Nodes represent entities like people or companies, while edges and properties represent relationships between entities. With the notion of graphs, we can model data naturally and store them in logic level. Examples of graph databases are AllegroGraph and gStore.

## 4. Performance

### 4.1. Configuration

To evaluate the performance of relational databases and non-relational databases, we are using a benchmark tool from stssoft[11] to test the read and write speeds among relational database (MySQL), key-value storage (Berkeley DB), and document storage (MongoDB). The test platform configuration is:

- OS: Microsoft Windows 10 Pro 64 bit
- CPU: Intel (R) Core (TM) i5-3210M CPU @ 2.50GHz
- RAM: DDR3 8GB 1333MHz
- Storage: SAMSUNG MZ7PC128HAFU-000H1
- Database Settings: InsertsPerQuery: 5000; Indexing Technology: BTree;
- Test Data: 1,000,000 Records, 100% randomness

### 4.2. Results

As we can see in 3 and 4, NoSQL are much faster than traditional SQL database in terms of read and write speed, especially in key-value storage like Berkeley DB, which means less waiting time in scenarios such as online transactions.

Acute reader may noticed that in terms of writing speed, MySQL is slower than MongoDB in at beginning, but gradually become faster than MongoDB when data collection become huge. This may because the pre-heat of the DBMS system, but more experiments still needed to conclude a reason.

Based on the results, we can conclude that in NoSQL database different types of operation will lead to various

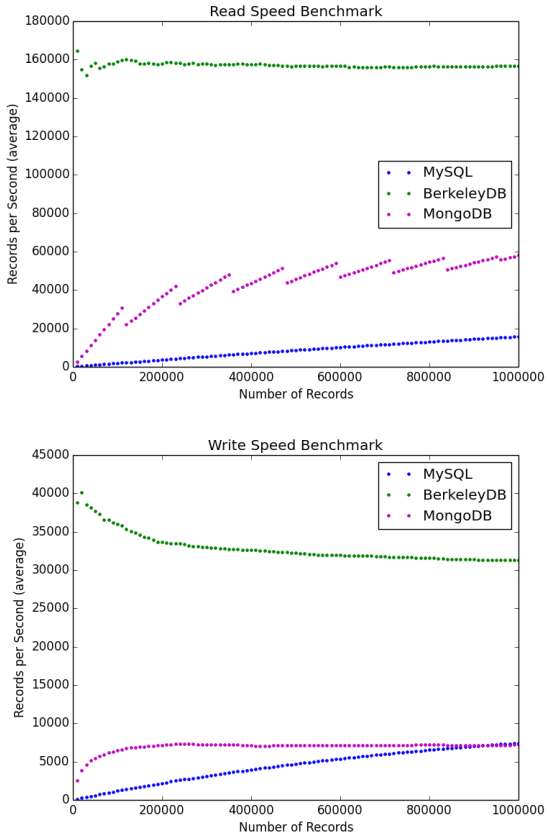



**Figure 4.** read/write speed

performance[2]. Although not all NoSQL databases perform better than SQL database, we can conclude that NoSQL databases are generally faster than SQL databases.

## 5. Conclusion

In a word, it's hard to simply say SQL and NoSQL which one is better. Both SQL and NoSQL have advantages and disadvantages under different situations. SQL databases, for the reason that they are relation oriented, have advantages of vertical scalability and strong consistency. As consistency is prioritized in SQL databases, the database manage system has to do lots of work to maintain the consistent state, which will definitely compromise the performance. NoSQL databases, as they are designed to be flexible and fast, have less constraints than SQL by reducing the overhead of consistency. As for flexibility, NoSQL can store data in several types like objects (documents or key-value pair) distributedly. As for speed, NoSQL is generally faster than SQL, especially for key-value storage in our experiment; On the other hand, NoSQL database may not fully support ACID transactions, which may result data inconsistency.

## References

[1] International Journal of Advanced Research in Computer Science and Software Engineering, 'SQL and NoSQL Databases' by Vatika Sharma, Meenu Dave.

[2] IEEE Pacific RIM Conference on Communications, Computers, and Signal Processing ' A performance comparison of SQL and NoSQL databases' by Li Yishan, Manoharan Sathiamoorthy

[3] 'Comparative Study of SQL & NoSQL Databases' by Supriya S. Pore, Swalaya B. Pawar

[4] Rick Cattell. 2011. Scalable SQL and NoSQL data stores. SIGMOD Rec. 39, 4 (May 2011), 12-27.

[5] Han J, Haihong E, Le G, et al. Survey on NoSQL database[C] Pervasive computing and applications (ICPCA), 2011 6th international conference on. IEEE, 2011: 363-366.

[6] S. H. Aboutorabi, M. Rezapour, M. Moradi and N. Ghadiri, "Performance evaluation of SQL and MongoDB databases for big e-commerce data," 2015 International Symposium on Computer Science and Software Engineering (CSSE), Tabriz, 2015, pp. 1-7.

[7] 'Next Generation databases: NoSQL, NewSQL, and Big Data' by Guy Harrison

[8] 'NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence' by Pramod J. Sadalage, Martin Fowler

[9] 'Adopting NoSQL Databases Using a Quality Attribute Framework and Risks Analysis' by Hilda Mackin, Gonzalo Perez and Charles C. Tappert

[10] https://en.wikipedia.org/wiki/CAP_theorem

[11] https://stssoft.com/benchmark

[12] 'Fundamentals of Database Systems 7th Edition' by Ramez Elmasri, Shamkant B. Navathe