International Journal of Scientific & Engineering Research Volume 8, Issue 5, May-2017
ISSN 2229-5518

A Comparative Study: MongoDB vs MySQL

Mr. Sushil Soni ,Mr. Mayuresh Ambavane ,Mr. Shamal Ambre , Mr. Shirshendu Maitra

Abstract-The relational database has been the foundation of enterprise applications for decades, and since MySQL's release in 1995 it has been a popular and inexpensive resource. Yet with the explosion in the volume and variety of data, recently non-relational database technologies like MongoDB have emerged to address the needs of new applications. MongoDB is not only used for new applications but also to augment or replace existing relational infrastructure.

In this paper we will try to show case a comparative study of non-relational databases and relational databases. We mainly emphasis our presentation on one application of the NoSQL database technology, known as MongoDB, and make a comparison with another application of relational databases, known as MySQL, and thus justifying why MongoDB is more efficient than MySQL. We will also present the benefits of using a non-relational database compared to a relational database. A comparison criterion includes theoretical differences, characteristics, limitation, integrity, distribution, system requirements, and architecture, query and insertion times.

Index Terms— MySQL, MongoDB, NoSQL, RDBMS

1 INTRODUCTION

A few years back an application normally only used to have thousands of users to tens of thousands of users in the most extreme case, nowadays there are applications that have millions of users and who are connected day-and-night, year in and year out. It is important to use an appropriate database, which supports simultaneous connection of hundreds of thousands users.

Relational databases are globally used in most of the applications and they have good performance when they handle a limited amount of data. To handle a large volume of data like internet, multimedia and social media the use of traditional relational databases is ineffective. To overcome this problem the "NO SQL" term was introduced. The NoSQL term was used by Carlo Strozzi in year 1998 and refers to non relational databases, term which was later reintroduced in 2009 by Eric Evans. Nowadays, the term has received another meaning, namely "Not Only SQL", which is a lenient variant of defining the term, compared to its previous significance, the anti-relational.

NoSQL, is not a tool, but a methodology composed of several interdependent and competing tools. The primary benefit of a NoSQL database is that, unlike a relational database it is able to handle unstructured data such as documents, email, multimedia and social media efficiently. Non relational databases do not use the RDBMS principles (Relational Database Management System) and don't store data in tables, schema isn't fixed and have very simple data model. Instead, they use identification keys and data can be found from the keys assigned.

There are four strategies for storing data in a non-relational database, as shown in, and they are as follows:

1. Key-Value - Key-Value databases are conceptual distributed dictionaries and don't have a predefined schema; they are schema less. The key can be synthetic or self-generated, and the value is able to be anything: string, JSON, BLOB and others.

2. Document - MongoDB is the most popular document based databases. They are flexible in the type of content because they don't have a predefined schema. Conceptually, they work well with documents of many different types such as JSON, BSON, XML and BLOBs. Basically they represent only a specialization of key-value databases. A document is written or read using a key. Besides the range of capabilities Key-Value, document based databases add different opportunities to find documents based on their content.

3. Column/ Field – Databases from BigTable category, like HBase and Hypertable are columnar type and should have a predefined schema. Data is stored in cells grouped in columns, and the columns are logically grouped into groups of columns. Hypothetically, they can contain an unlimited number (limited depending on the application) of columns that can be generated at runtime or at schema definition.

4. Graph-Oriented – This strategy can help complex data queries which are also performed in an approximately smaller interval of time compared to other databases using the strategies proposed above.

Also, non-relational databases provide high flexibility at insertion or deletion of an attribute from the database because they don't have a fixed

database schema. Based on the requirement of the application, we can make use of different type of NoSQL database and each NoSQL database has its own functions, data model and architecture options of the database depends on the application.

In this paper we concentrate on one of the NoSQL technologies, namely MongoDB, and make a comparison with MySQL to highlight why MongoDB is more capable than MySQL. In addition, we will present the benefits of using a non-relational database in a various application, using MongoDB as the NoSQL database.

2 OVERVIEW OF MongoDB

MongoDB is a schema less document-oriented database. The name MongoDB comes from "humongous". The database is written in C++ and is intended to be scalable. The primary reason for moving away from relational model is to make scaling easier. The fundamental idea is to replace the concept of a "row" with a more flexible model; the "document". By making use of embedded documents and arrays, this perspective makes it possible to represent complex hierarchical relationship with a single record. MongoDB is also schema free i.e. a document's keys are not predefined or fixed

MongoDB provides high performance, high operability, high availability, and easy scalability. MongoDB works on fundamental idea of collection and document.

Database: - Database is a physical, real-time container for collections. Each and every database gets its own unique set of files on the file system. A particular MongoDB server generally has multiple databases.

Collection: - Collection is a set of MongoDB documents. It is similar to an RDBMS table. A collection operates within a single database. Collections don't enforce a schema. Documents within a collection can have many different fields. Generally, each and every document in a collection is of similar or related motive.

Document: - A document is a set of key-value pairs. Document have dynamic schema. Dynamic schema means that the documents in the same collection don't need to have the exact same set of fields or columns or structure, and common fields in a collection's documents may hold many different types of data.

Data Design in MongoDB database holds a set of collections. A collection has no pre-defined schema such as tables, and stores data as documents. BSON (objects like binary encoded JSON) are used to store documents. A document is a set of fields that can be thought of as a row or tuple in a collection. It can contain complex structures like lists, or even document. All documents have an ID field, which is used as a primary key (field which uniquely identifies each document) and each collection can contain any type of document, but queries and indexes can only be made against one collection. MongoDB supports indexing over embedded objects and arrays thus have a special feature for arrays called "multikeys". This capability allows using an array as index, which can then be used to search documents by their associated tags. Figure 1 shows the structure of MongoDB.

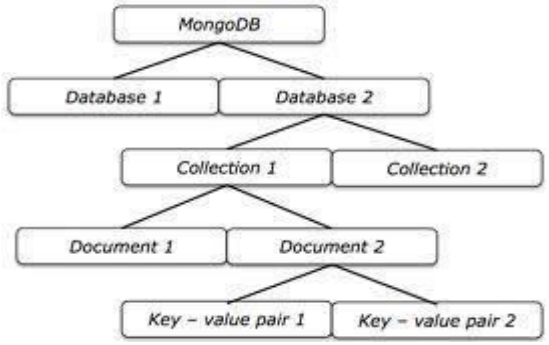

Figure 1: STRUCTURE OF MongoDB

MongoDB has its own query language named Mongo Query Language. To get certain documents from a db collection, a query document is created containing the fields that the desired documents must match. For example,

Insert Command

db.users.insert ({ user id:"xyz123", age: 34, status:"X"})

Select Command

db.users.find ({ status:"X", age: 34})

Delete Command

db.users.remove ({ status:"X"})

Drop Command

db.users.drop ()

3 COMPARISON OF MongoDB & MySQL

As per the detailed review of several papers, a comparative study is made between MongoDB and MySQL based on their fundamental concept and commands used for different operations.

A. Based on Terms / Concept

RDBMS	MongoDB
Database	Database

Table	Collection
Row/Tuple	**Document**
Column	Field
Table Join	Embedded Documents
Primary Key (explicitly)	Primary Key (Default key _id provided by MongoDB implicitly)
Group by	Aggregation
Fixed schema	Schema less

B. Based on Schema Statements

SQL schema	MongoDB schema
CREATE Command	
CREATE TABLE teachers(t_id Varchar(30), age Number, status char(1), PRIMARY KEY(id))	db.teachers.insert({t_id:"abc123",age:55, status : "A"})
DROP Command	
DROP TABLE teachers	db.teachers.drop()
INSERT Command	
INSERT INTO teachers(t_id , age, status) VALUES ("a123",45,"A")	db.teachers.insert({t_id:"a123",age:45,status:"A"})
SELECT Command	
SELECT t_id, status, age FROM teachers	db.teachers.find({ },{t_id:1, status:"B",age:45 })
DELETE Command	
DELETE FROM teachers WHERE status ="D"	db.teachers.remove({ status:"D" })

C. Based on Performance

Several other research paper authors have performed testing and thus have compared MongoDB with MySQL database. They have performed testing by using the textbook management system. The given graph shows the result of testing. In performance testing, the authors have inserted 100 to 50,000 textbooks information into database. The cost time for MongoDB and MySQL were recorded as shown in figure. Two important factors for which MongoDB was preferred over MySQL are:

Insertion Speed

From the graph, we can see that MongoDB spends less time than MySQL, for a large amount of information as shown in figure2. It leaves MongoDB 30× to 50× faster than MySQL as sown in figur3.

Number of Parallel Clients 5 Time in seconds

Basic Insert	Total Rows	Rows / client	SQL Time	Mongo Time	Sql Ops/sec	Mongo Ops/sec
several columns	100	20	0.19	0.011	526	9,091
600 bytes per row	1,000	200	1.8	0.02	556	50,000
	5,000	1,000	9	0.25	556	20,000
	25,000	5,000	100	1.5	250	16,667
	50,000	10,000	270	2.5	185	20,000

Figure 2: INSERTION SPEED COMPARISONS

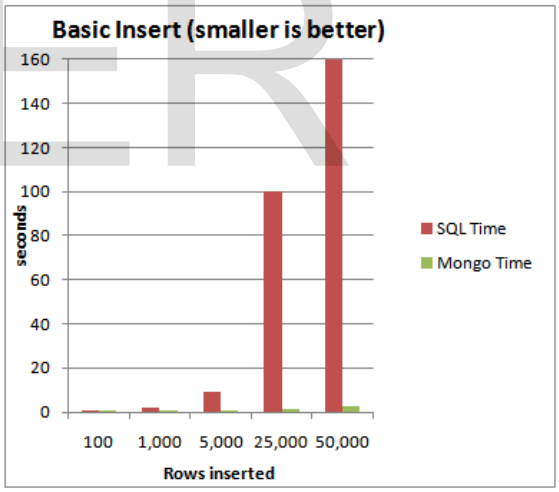

Figure 3: INSERTION TIME FOR MySQL AND MongoDB

- **Query Speed**

In the figure 4, it calculates the time to get the data out of the database. MongoDB leads MySQL with almost 3× performance as shown in figure 5, 6. But MongoDB spends much more time on problem solving as well as the post maintenance issues and is not easier than MySQL. Thus from above comparison; it proves that for large amount of data MongoDB is preferred over MySQL.

Number of Parallel Clients 5 Time in seconds

	Total Rows	Rows / client	SQL Time	Mongo Time	Sql Ops/sec	Mongo Ops/sec
Basic Query	50	10	0.1	0.08	500	625
with index	500	100	0.38	0.1	1,316	5,000
	5,000	1,000	2.8	1.2	1,786	4,167
	25,000	5,000	14	4	1,786	6,250
	50,000	10,000	28	10.4	1,786	4,808

Figure 4: QUERY SPEED COMPARISONS

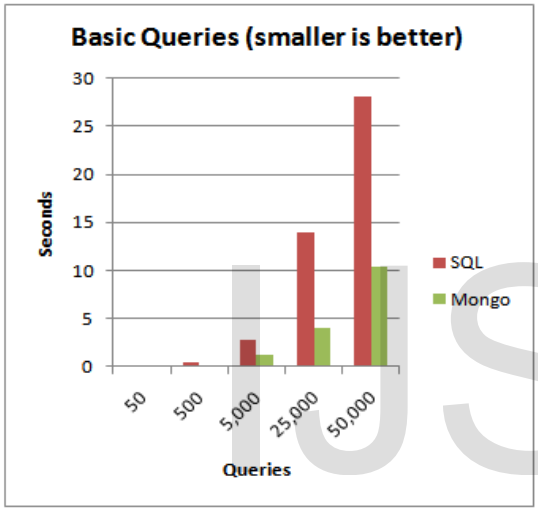

Figure 5: BASIC QUERY TIME FOR MySQL AND MongoDB

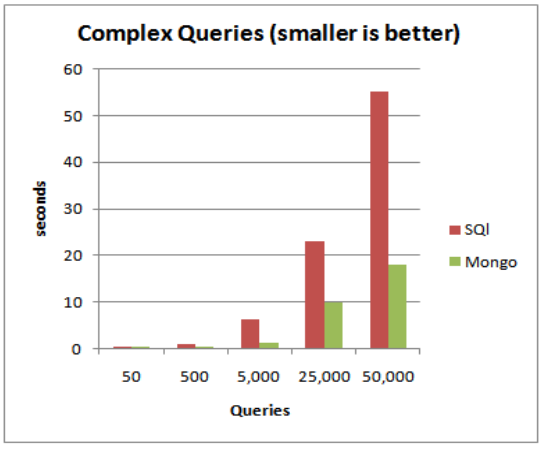

Figure 6: COMPLEX QUERY TIME FOR MySQL AND MongoDB

4 CONCLUSION

RDBMS won't go away, they're still definitely needed. But the storage requirements for the new generation of applications are largely different from legacy applications. We can choose MongoDB instead of MySQL because of two factors, ease of use and performance. We conclude that if your application is data intensive and stores lots of data, queries lots of data, and usually lives and breathes by its data, then you'd better do that efficiently or have resources (i.e. money) to burn. Lastly, the report concludes by proposing a database integration method by using a middleware between the two layers. In this method, application does not have to consider about the complexity of underlying database layer there data distribution and storage. They have to use the basic SQL query language to get result from the database and all the format conversion rules will be done by the Metadata. The system was proposed because MongoDB has newly come into existence, whereas the standard SQL language has been over years and, therefore if we merge the two we can use the features of both the database. Although, NoSQL has the advantage of horizontal expansion, but for complex SQL requests, it cannot support them very well. For the Query based on KEY/ VALUE and massive data storage requirements, NOSQL is a good choice.

5 REFERENCES

[1] SQL Support over MongoDB using Metadata Sanobar Khan*, Prof.Vanita Mane**

[2] Bogdan George Tudorica, Cristian Bucur, "A Comparison between several NoSQL Databases with comments and notes"

[3] Jing Han, Haihong E, Guan Le, "Survey on NoSQL Database", IEEE 978-1-4577-0208-2,2011

[4] Neal Levitt, "Will NoSQL Databases Live Up to Their Promise?" IEEE Computer Society, vol.43, no.2, pp.12-14, Feb.2010

[5] A Comparative Study: MongoDB vs. MySQL Conference Paper · June 2015 DOI: 10.13140/ RG.2.1.1226.7685

[6] A comprehensive comparison of SQL and MongoDB databases by Rajat Aghi, Sumeet Mehta, Rahul Chauhan, Siddhant Chaudhary and Navdeep Bohra