ISSN 2278-3091

International Journal of Advanced Trends in Computer Science and Engineering, Vol. 3 , No.2, Pages : 16 - 20 (2014)
Special Issue of ICACE 2014 - Held on March 10, 2014 in Hotel Sandesh The Prince, Mysore, India

Implementing Information System Using MongoDB and Redis

Yogesh Punia[1], Rinkle Aggarwal[2]
[1]Department of C.S.E Thapar University Patiala, India, yogesh.punia66@gmail.com
[2]Department of C.S.E Thapar University Patiala, India, raggarwal@thapar.edu

Abstract : Demand for speed, manageability, distributed, open-source, agile development with schema-less databases, easier horizontal scalability leads to fixing of broken parts in MySql or Oracle by introducing next generation databases also termed as NoSQL. These are modern web-scale databases which are introduced to handle emerging applications such as social network analysis, bioinformatics network analysis and semantic Web analysis where a wide variety of data is to be processed which needs continuous witness with a quick increase. A critical challenge these days is to have an effective management and analysis of data at a large-scale. This paper makes an effort to use a combination of NoSQL databases to replace traditional database like Oracle applied to information management system, comparing the traditional database system with combination of MongoDB and Redis and presents the performance comparison of these two schema.

Key words : *MongoDB; Oracle; Redis; MySQL; NoSQL; RDBMS*

INTRODUCTION

Traditional database has been serving from early 70's whether it is about storing large amount of data or fetching data from multiple tables through complex joins. However it started lacking with the rise of web 2.0 as the volume of data stored about users has piled up to layers as for instance in social networking sites facebook. Google and Amazon found it very challenging when system crawls while millions of users lookup against tables having millions of rows of data. Much effective management and analysis of data at a large-scale poses critical challenge. These days, big data has attracted a lot of attention from industry and academia [1]. Relational database face tough competition as it does not suit well to complex programming structures consisting of complex data types and data of hierarchical nature. Their first and foremost advantage is that, not in a way as relational databases, they handle unstructured data such as e-mails, word-processing files, social media and multimedia efficiently [2]. Complex objects consisting objects and further lists inside them cannot be mapped directly to a single row in a single table. In coding point of view it does not map well as it is difficult to handle the syntax of SQL along with the client side code for accessing SQL database. In order to overcome all these problems a range of new databases has been introduced(currently 150) having categories like Wide Column Store/Column Families, Document store, Key Value/Tuple Store, Graph Databases, Multimodal Databases, Object Databases, Grid & Cloud Database Solutions, XML Databases, Multidimensional Databases, Multivalue Databases, Event sourcing and other.

In this paper, the relational database used is Oracle which is a popular open source database with small, fast and low-cost features. NoSQL database system uses MongoDB, which is a cross-platform document -oriented database system, characterized by mass data storage and good query performance. MongoDB is merged with Redis which is another key-value store NoSQL database. It is combined with MongoDB for fast retrieval of data. REmote DIctionary Server is short termed as Redis. Section I of this paper explains concepts of MongoDB in detail and section II introduces the features of Redis and need of merging it with MongoDB. The implementation of Clients' Insurance Policy Information Management System based on MongoDB and Redis. Section III compares the performance of Clients' Insurance Policy Information Management System using Oracle and MongoDB. In section IV future work is explained.

MongoDB

A. Non-Relational Database

MongoDB (from "hu**mongo**us") is an open-source document database and the leading NoSQL database. Written in C++ [6]. NoSQL refers to non-relational databases promises to handle large volumes of structured and unstructured data. Unlike relational databases they NoSQL databases do not require schemas to be defined beforehand. These fits perfectly fine with the agile technologies as each time a new feature is developed, the schema of the database is needed to be changed. The data-structure of non-relational database is not fixed. This allows the insertion of data without a predefined schema.

This ultimately proves helpful in making significant changes in applications in real-time, with no interruption in service, leading to faster development, reliable code integration and lessens the amount of time by database administrators. NoSQL also referred as “Not only SQL” to emphasize that these also support SQL like queries. MongoDB is a document-oriented database released in 2009. It holds a set of collections; a collection holds a set of documents. A document further is a set of key-value pairs. These documents have a dynamic schema which means that documents in the same collection do not need to have the same set of fields or structure. The key feature for which MongoDB is used is its flexibility which is understood as that the data is stored in JSON documents as shown in fig 1 below:

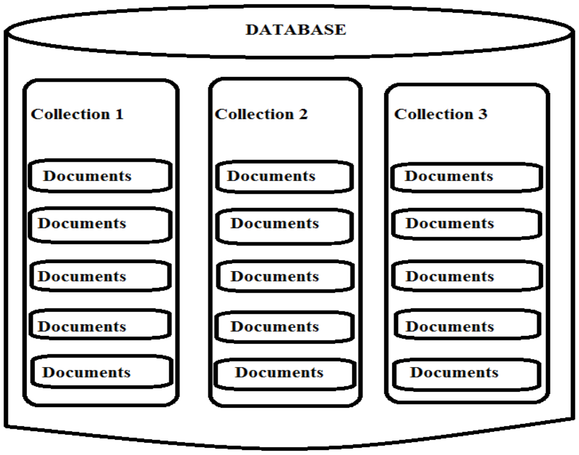

Fig 1 MongoDB database.

B. *Key feature of MongoDB:JSON*

These provides a rich data model that flawlessly maps to native programming language, and the dynamic schema makes it easier to unfold your data model as compared to a system with compelled schema RDBMS. MongoDB is chosen for evaluation because it can manage small data as well as large data efficiently.

I. Redis:Key-value Store Database

Redis (REmote DIctionary Server) is an open source, advanced key-value store. The key feature of Redis is that it can store more complex data types and atomic operations like appending to a string; incrementing the value in a hash; pushing into a list; getting the member with highest ranking in a sorted set; computing set union, intersection, difference can be defined against these data types. It is in-memory but persistent on disk which means data does not disappear on system restart. Redis holds the whole dataset in memory. In-memory data set here means that data may be swapped out when not in use for long or not needed in future much. It is also termed as data structure server since keys may contain strings, lists, hashes, sets, and sorted sets.

In [3] the author had explained the key-value implementation of database. He made an approach to persist the objects under race and failure conditions. As Redis is well famous for speed and scalability so it is widely used with cloud storage. As cloud web applications demand fast request and retrieval of data. Whereas relational database could have been proved a bottleneck for this requirement of speed. RDBMS had fixed schema and if they had to be used in future then there may be a lot of space wastage with lots of unused columns. Whereas KVS had been really simple while designing where values may be stored anytime using a unique key. Those values could be used later using that key. Also known as hash addressing.

RDBMS have various serious advantages over KVS: such as they are well tested, also have good management tools, apart from this have programming patterns available. The Key-values on the other hand are fairly newly introduced, each of them is having different approaches and these excel only in a very small segment [4]. Author in [7] has discussed about differences between the traditional SQL database and newly emerging technologies coming up with the name NoSQL like MongoDB, Redis. NoSQL had earlier misunderstood as anti-SQL but actually NoSQL is that class of database management system which is different from the traditional relational databases where data is not stored using fixed table schemas. The purpose of newly designed databases is to serve as database system for huge web-scale applications where they have proved to be more efficient.

Redis is used when data is not meant to be lost and is small enough to fit in memory. Data is to be stored and cleaned up carefully. It may begin to suffer performance problems once it exceeds the memory limit. It operates as a single process and single threaded. Thus, a single Redis instance cannot perform parallel execution of tasks. Also this database cannot be used where durability of data is required. Data may get lost in case of any incidents, so its usage is limited.

IMPLEMENTATION SCHEME

A. *NoSQL Data Model*

The data model which is used here is based on MongoDB and Redis. Using NoSQL provides the benefit of storing data in schema less structure. As per the system's requirements; the basic information which is stored in MongoDB is about Insurance Policy's, client's and location's information. Insurance_ID, Total_Installments, Installments_left can be set into the basic information of Insurance with embedded documents in collection. Client's embedded documents contains the information of client personal details like name, address, insurance_id. Location is one other document which contains the location as per insurance id.

B. *System Implementation*

This work is done using Java language, MongoDB is used to store the data. This data is then made to split according to the location of the person. This data is made available at the different call centers at different locations. The categorized data according to the location is retrieved with the help of Redis and this is done for the fast access and operations on data. Thus two databases are merged in order to store and retrieve. MongoDB is used for large-scale data to be stored in schema-less structure and Redis is used for fast retrieval of data.

i. Establishing a Connection:

C. *To connect Java programs with MongoDB database, drivers are installed and environment variables are set. Afterwards Equations*

Number equations consecutively with equation numbers in parentheses flush with the right margin, as in (1). First use the equation editor to create the equation. Then select the development kit of MongoDB is imported in the application program. Following are the basic CRUD commands used while implementation. To connect to MongoDB databases, the key code is as followed:

Mongo mong=new Mongo(“DBServer”,27017)//connecting with the database server name and port number

DB db=mong.getDB("final");//open database, 'final' is the name of the database

i. Inserting the data

At first, Java program establishes the connection with database then passes the SQL statements to program. The data is inserted into database through java server pages (jsp). MongoDB database make use of object-oriented thinking for implementing the system, first, accordingly "insurance" to create a DBCollection object col, and then create a BasicDBObject object obj, add insurance information to the object col, and finally inserting data into database.

```
DBCollection col=db.getCollection("insurance");
BasicDBObject document=new BasicDBObject();
                        document.put("name", name);
       document.put("phone", phone);
       document.put("mail", mail);
       document.put("address", address);
       document.put("location", location);
       document.put("insurance_id", insurance_id);
                        col.insert(doc);
```

ii. Query the data

The data stored in MongoDB is inserted, updated and deleted using: col.update(document,newdocument) ;
col.remove(searchQuery);

The data is updated and deleted by searching through the documents using cursor:

```
DBCursor cursor =table.find(searchQuery);
while(cursor.hasNext()){
   //processing the data cursor.Next();
}
```

iii. Making connection with Redis

The data stored using MongoDB is splitted according to the attribute 'Location' and stored in Redis. To make connection with Redis following code is required:

```
static Jedis jedis = new Jedis("DBServer",6379);
{
        try{
      Jedis.connect()
            }
   catch(JedisConnectionException
System.out.println("can not connect");}
}
```

iv. Inserting data into Redis:

The data is inserted with the following code

```
            jedis.set(insurance,address);
            mean "approximately" or "effectively." Do not
            use the word "issue" as a euphemism for
            //insurance is key, address is value
```

ii. Retrieving the data using Redis

The data is categorized as per location and sent to the required destination, the information is retrieved at a faster pace at the client's demand.

searchQuery.put("location",location);

PERFORMANCE COMPARISON

To compare performance of MongoDB with Oracle following implementation of Oracle is needed. Same data has been saved, updated, deleted and retrieved by firing the queries in the next sub-section.

a) Oracle data model

The performance testing is done in comparison to Oracle which is a structured relational database. It uses common SQL language. SELECT, UPDATE, DELETE are the commonly used data manipulation statements. Modern relational databases are established based on the relational model of data proposed by E.F. Codd [6]. The data is stored in tables. Afterwards to split the data joins are applied which take time to retrieve the data, whereas in MongoDB it is a matter of only a small fraction of time as compared to time taken by oracle. As the number of records are added on a larger level this difference in time taken is a great slot. In MongoDB, for the operations like adding new records, updating and deleting the existing records, functions are used.

b) Traditional system implementation

In Oracle, the creation of table is done using CREATE statement:

```
CREATE TABLE info_insurance(
name varchar 2(50),
phone number(10),
mail varchar2(20),
address varchar2(50),
location varchar2(20),
insurance_id varchar2(10) not null
);
```

For operation like deletion of data, following query is fired when no other table is related to this one.

DROP TABLE info_insurance;

To insert data into Oracle databases, a new record is saved into database by using INSERT command. As for instance, following code is written:

INSERT INTO info_insurance (name, phone, mail, address, location, insurance_id) VALUES ('Ram',9183929295,'54/4 GandhiMarg', 'delhi', 'I904");

To update into Oracle databases either the entire table or part of an entry following statement is used:

```
UPDATE info_insurance
Set address='23 Mall Road'
WHERE location='Noida';
```

a) Comparing with MongoDB in insertion of data

A bunch of data is inserted both in MongoDB and in Oracle as well in order to check the amount of data taken in case of insertion in milliseconds. Table 1 shows the number of milliseconds for each bunch of inserts.

Table 1-Inserting time (in milliseconds)

No. of Records	**Oracle DB**	**MongoDB**
10	31	800
100	46	4
1000	1346	40
10000	8766	678
100000	84678	4321
1000000	884567	56789

Fig 2 shows that the amount of time spent while insert operation in Oracle database is comparatively high. It is put in the form of graphs as shown below:

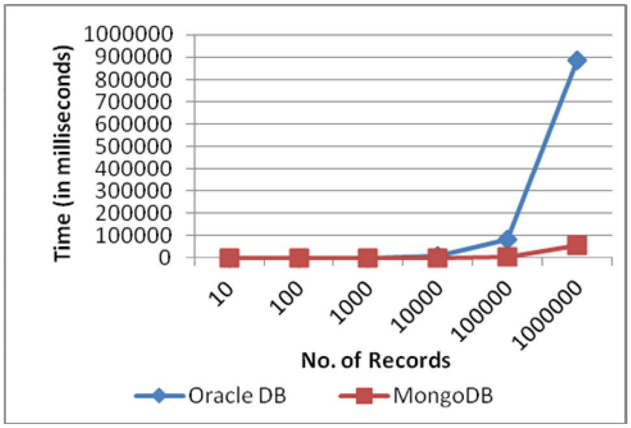

Fig 2 Inserting data

MongoDB is efficient for large insertion of large amount of data. For a small bunch of data it took a little too long to insert records. Oracle database on the other hand deals very nice for small amount of data but as the number of records is increasing, the time spent is bigger.

MongoDB and Redis are combined that is data is further segregated and stored on Redis database for fast retrieval. It is mandatory to show how the performance would have been affected if Redis has not been used and data would have been retrieved from MongoDB only.

Table 2-MongoDB vd Redis

No. of Element s	**Mongo Read**	**Redis Read**	**Mongo Write**	**Redis write**
10	5	2	8	5
100	11	13	34	8
1000	93	38	153	31
10000	980	238	1394	220
50000	5218	958	8713	979

The following graph depicts the comparison between read an write operations in MongoDB and Redis:

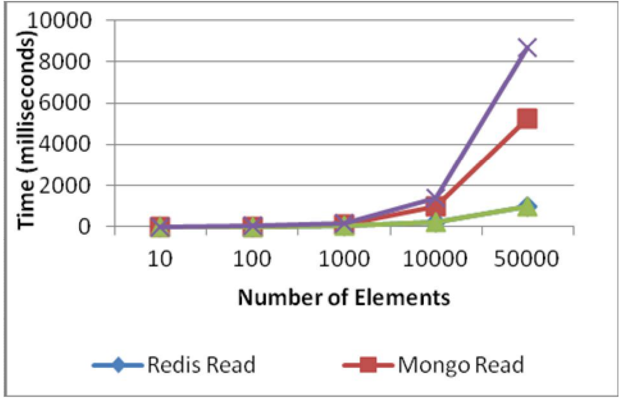

Fig 3 **MongoDB vs Redis**

CONCLUSION

Bigdata has been implemented using combination of MongoDB and Redis at the backend. This is done in order to do fast transactions of data that is fast insertion and retrieval of data. Both MongoDB and Redis are scalable and Redis is used for its speed as it used KVS (Key Value store) whereas MongoDB is used for its flexibility, scalabilty, and efficient performance. With the help of MongoDB complex data into one field. An array or an object or a reference could be stored in a field in MongoDB. Also the deletion time in MongoDB was constant where as it generally increases in case of Oracle[6]. MongoDB is able to store large amount of data in schema-less structure and it is scalable. It is quite flexible in nature. It is easy to deploy and copy database from one server to other by using import-export tools. Complex data may be stored into a single field like an array or a reference.

Oracle is a bit complex database, it has relations between tables and tables have a fix structure. These relations may be one to one or many to many. These relations may be helpful to join tables and create complex queries.

The problem with Oracle is replication, if the database is to be copied, it is quite difficult. Even if the tools are considered for this purpose, they are not fast enough. It is very much slower than MongoDB. MongoDB is still lack behind Redis as shown above.

FUTURE SCOPE

The present implementation can be used to design more complex applications using some map-reducing functions like to aggregate result functions. Also some plugins could be developed as it is an open source community. Various other combinations of different types of databases can be applied to implement the above mentioned management system and can be compared in terms of performance and reliability.

REFERENCES

[1] Changqing Ji; Yu Li; Wenming Qiu; Awada, U.; Keqiu Li, "Big Data Processing in Cloud Computing Environments," *Pervasive Systems, Algorithms and Networks (ISPAN), 2012 12th International Symposium on* , vol.,

[2] Leavitt, N., "Will NoSQL Databases Live Up to Their Promise?," *Computer* , vol.43, no.2, pp.12,14, Feb. 2010 doi: 10.1109/MC.2010.58

[3] Persisting Objects in Redis Key-Value Database Matti Paksula University of Helsinki, Department of Computer Science Helsinki, Finland

[4] Armbrust, M., Lanham, N., Tu, S., Fox, A., Franklin, M., and Patterson, D. A. Piql: A performance insightful query language for interactive applications. First Annual ACM Symposium on Cloud Computing (SOCC)

[5] implement textbook management system instead of MySQL," *Communication Software and Networks (ICCSN), 2011 IEEE 3rd International Conference on* , vol., no., pp.303,305, 27-29 May 2011

[6] http://www.mongodb.org/

[7] E.F. Codd E. F. Codd. 1970. A relational model of data for large shared data banks. ***Commun. ACM*** 13, 6 (June 1970), 377-387. DOI=10.1145/362384.362685 http://doi.acm.org/10.1145/362384.362685