MySQL and NoSQL database comparison for IoT application

Sharvari Rautmare
Dept. of E & TC
Sinhgad College of Engineering,
Pune, India.
sharvari318@gmail.com

Dr. D. M. Bhalerao
Dept. of E & TC
Sinhgad College of Engineering,
Pune, India.
dmbhalerao@sinhgad.edu

***Abstract*— Internet of Things (IoT) concept has been around in tech world for few years now. IoT focuses on connection of number of smart devices. In near future, IoT will have applications in various domains and these applications are going to produce tremendous amount of data. With the continuous generation of heterogeneous data, problem arises to store, transfer & manage the data efficiently. Traditional database systems used Structured Query Language (SQL) database which has supported all the user requirements along with simplicity, robustness, flexibility, scalability, performance. But the main limitation they are facing is their static schema which is making RDBMS not suitable for IoT applications. On the other hand, NoSQL databases emerging in market have claimed to perform better than SQL database. The NoSQL databases are non-relational, schema free, no joins, easy replication support, horizontally scalable, etc. Does NoSQL perform better than SQL in all application scenarios? An effort to answer the same has been made in this paper. This paper compares SQL and NoSQL databases for a small scale IoT application of water sprinkler system and investigates whether NoSQL performs better than SQL in different scenarios.**

Keywords*— Database Management System (DBMS); Internet of Things (IoT); NoSQL; SQL;

I. INTRODUCTION

A lot of research work is going on in Internet of Things(IoT) domain. IoT refers to any object/thing around us which is connected to internet. This object is also able to communicate through internet [1]. Sensors are attached to these objects/things, like temperature sensor connected to oven, fridge or just to measure a room temperature. IoT is network of such things which may communicate with each other or with user. IoT is having application in various domains like smart city, industrial, medical services, etc. As, the application area is wide, the type of data generated by these things is heterogeneous. In near future, data generated by these IoT applications will be big data which can be further analyzed to make the optimum use of resources. Also, the use of IoT applications will increase the generation of real-time data, making it challenging to store and manage the data. Several new technologies and database management systems (DBMS) are emerging to handle the data growth and to improve the system performance. SQL databases have been the most popular database systems so far, question arises whether the traditional relational databases will perform at par with ever increasing user requirements. A new class of database is emerging and is referred to as NoSQL database [2]. NoSQL databases store data in a different manner as compared to the traditional methods of relational database systems. They are meant for data of schema-less structure. NoSQL databases can be easily distributed and hence they provide high scalability and availability. These are the properties which are actually required to manage big data or Internet of Things data.

II. RELATED WORK

To realize the Internet of Things technology, a lot research is going on in different fields related to IoT. IoT data is going to be heterogeneous as there are various applications of IoT in varied domains. The data generated by these IoT applications is further supposed to be processed and analyze so as to optimize the application further. So, tasks like Data collection, storage, data processing, generating reports are managed by the database management system. To handle the huge amount of data and make the data retrieval faster, authors [3] have proposed the design, implementation and evaluation of three data retrieval approaches for RFID Tags. Alternate, Sequential & parallel sensor sampling scheduling schemes proposed were compared based on parameters such as time delay, power consumption, communication range & read rate. Limited use of data and number of sensors is the limitation of the study. Authors [4] are focusing on application oriented work. Emergency medical services related IoT application is used to explore data accessing methods. For various IoT applications, question of whether SQL or NoSQL databases prove to be better, still remains unanswered. Such comparative study are done [5] [6] with data stored in cloud. Analysis of SQL and NoSQL database for specific IoT application is of our interest.

III. DATABASE MANAGEMENT SYSTEM

A. Database Overview

Database can be defined as an structured collection of data. The system which handles the data, transactions, problems or any other aspect of the database is the Database Management System (DBMS). Relational DBMS which use structured query language (SQL) was traditional database system. The latest trend in market is the non-relational database known as

NoSQL [7]. Both these databases are equal potential competitors of each other. This study is to find out the suitable database for IoT application. Before approaching to the comparative study of databases for IoT application, basic characteristics of database are explored. These characteristics of database are given by the CAP theorem which explains Consistency, Availability and Partitioning.

Consistency: means that once an update operation is finished, everyone can read that latest version of the data from the database and such system is consistent system. While, a system in which, updated data cannot be seen by all users at once is known as eventually consistent. Availability: is achieved if the system always provides continuous operation. Availability is achieved by deploying the database as a cluster of nodes, using replication or partitioning data across multiple nodes. In such case, if one node crashes, the other nodes can still continue to work. Partition Tolerance: A database system which can operate even if one of the nodes fail or is inaccessible. This is done by redirecting all queries to the failed node to some other active node of that system.

The traditional database systems like SQL, focused on consistency and supports following ACID properties.

Atomicity: Partially accomplished transactions are discarded. Transaction can only be successful or unsuccessful. Consistency: In the event of transaction failure, system reverts the transaction and goes back to the previous stable state. And hence, system always remains stable. Isolation: Transactions are completed without any interference and are processed independently. Durability: All the committed transactions are saved in logs and will not be lost. This helps to recover the system in case of abnormal terminations.

On the other hand, NoSQL database systems focus more on Availability and Partitioning and give eventual consistency. These systems follow BASE properties.

Basic Availability: NoSQL database focuses on availability of data as per CAP theorem requirements. Soft State: State of the database system is dynamic and may change over time due to eventual consistency. All the replicas of database do not have to be consistent all the time. Eventual Consistency: After any Write or update or delete operation, system may not immediately reflect the modifications done. But, eventually it will become consistent showing the modified data in all replicas.

The system following BASE properties is not strictly consistent. But, all the updates or modifications will be available eventually. In such systems, clients may encounter an inconsistency in data when updation or replication process is in progress. But, the data will reach the expected consistent state after the completion of replication. In IoT application domain, Number of users, speed requirements and user demands are ever increasing and hence partition tolerance of database becomes the important characteristic to be considered. NoSQL database gives priority to availability over consistency. On the other side, SQL follows the reverse order. The aim is to find out which of these two databases will perform better for a small scale IoT application of water sprinkler system.

B. SQL Vs NoSQL from IoT perspective

SQL Database: SQL Database follows relational data model to store the data. In this model, data is stored in rows and columns in a tabular form. Related tables can be interlinked together. Various relational databases available are MySQL, Oracle, SQLServer, etc.

NoSQL Databases: NoSQL follows non-relational data model. Non relational model supports schema-free storage of data in various forms such as document, graph. With features like horizontal scalability, schema less storage, support for unstructured data, NoSQL becomes competent for storing IoT data. Popularity for NoSQL databases spur because of the features it provides including high scalability, easy access, and distributed architecture. MongoDB, Redis, CouchDB, Hbase are some of the popular NoSQL databases available.

From IoT point of view, choosing the right database from these two database types is must. Differences between the two database types are discussed below with IoT perspective.

- *Scalability:* SQL database supports vertical scalability while NoSQL supports horizontal scalability. Vertical scalability refers to the ability to increase the performance of single node with adding resources such as memory or processors to the same node. In horizontal scalability, number of nodes (servers) is increased so as to share the system load. For new IoT applications which are in developing stage, using a database that has future expansion scope will be a practical choice. This will allow expansion of resources as and when required by the IoT application and no need for high initial investment.

- *Data Retrieval:* Faster data retrieval feature will be required when user has to fetch data from database for further processing. In SQL, various tables are linked together. To lookup data from different tables, user has to use JOIN statements which creates Views. This is a time consuming process. On the other hand, in NoSQL, data is stored in form of objects which will contain all the related data. This eliminates process of combining and then displaying the data, hence saving response time.

- *System Maturity:* SQL is an experienced technology and hence most of the issues have taken care of. Security features like authentication, data confidentiality & integrity are incorporated in SQL. On other hand, such security features are yet to be addressed in NoSQL. NoSQL might generate more issues or security breaches because of lack of maturity of the system. In some IoT applications, a secured communication channel is required to transfer sensitive data. For such applications, it is better to use a secured data storage system along with secured communication channel.

IV. SYSTEM ARCHITECTURE

For the research work, garden water sprinkler application is designed. Fig. 1 below shows the block diagram of the system. Temperature, humidity, soil moisture sensors are used in the hardware along with the water level detection. Sprinkler motor is turned ON depending on the inputs from soil

moisture and water level in the tank. If the soil moisture level drops below the threshold value and if water level is TRUE, then sprinkler motor turns ON. The data from these sensors is collected and is sent to database server and user email ID. The aim is to further utilize this data for analysis using data processing tools. This IoT application provides continuous flow of data. Data is stored in both MySQL and MongoDB database systems for carrying out the comparison study.

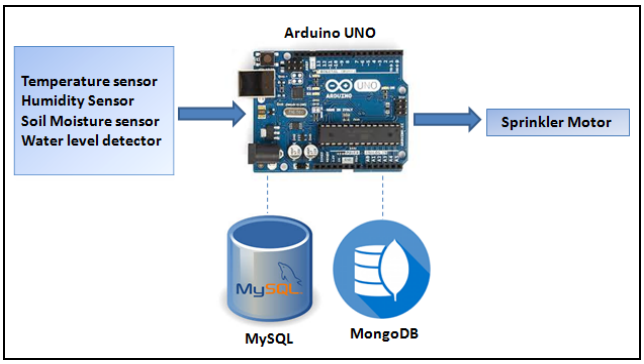

Fig.1. Water sprinkler block diagram

V. Experimental Results

A. Experimental Setup

For experimentation, hardware of water sprinkler system is designed using Arduino Uno board. Readings of soil moisture sensor and other sensors are given to analog input pins of Arduino board. This continuous reading is stored in two database systems namely: MySQL and MongoDB. MySQL version 5.5 is used for the experimentation. Along with it, MySQL graphical user interface of MySQL query browser1.2.17 is used. New schema is created to design table. Database structure created in MySQL for storing the IoT sensor data is shown below in Table I:

Table I: MySQL database structure

Fields	Data-Type
Sensor_id	Int(40)
Humidity	Varchar(10)
Soil	Varchar(10)
Temp	Varchar(10)
Created	Timestamp
User_id	Varchar(10)

For mongoDB, Mongo Booster 1.6.2 client is used to create collection and documents. Connection with local MongoDB server is made. Collection which is similar to table in mysql is created. Collection name: sensor_data is created. Documents (records) can be added in this collection using following syntax:

Db.collection.insert

({sensor_id: <sensor ID number>,

Humidity: <humidity sensor reading>,

Soil_moisture: <moisture sensor reading>,

Temperature: <temperature sensor reading>,

CreateDate: new Date(),

Userid: <User ID> })

Jmeter version 3.0 is used to measure the performances of the two database systems. Apache Jmeter is a java application. It is used for load testing functional behavior of database. Performance of the database is measured using JDBC configuration. Jmeter, MySQL, MongoDB are all open source software used in this study.

B. Results

In this comparative study, response time of two databases is measured for different queries. Detailed scenarios are discussed below:

1) SELECT query with varying number of threads:

The graph of response time Vs number of threads for SELECT query is shown below in fig. 2. Number of threads is varied from 1 to 10. Read process is carried out by using Select query which fetches 1000 records from 10000 records in the database. Response time for MySQL and MongoDB is somewhat similar till the number of threads is 4. From the graph, it is observed that, at thread 10, the rise in response time of MongoDB is increasing as compared to MySQL. In MongoDB, too many threads can overwhelm the system. This adds to the latency and eventually starving the CPU.

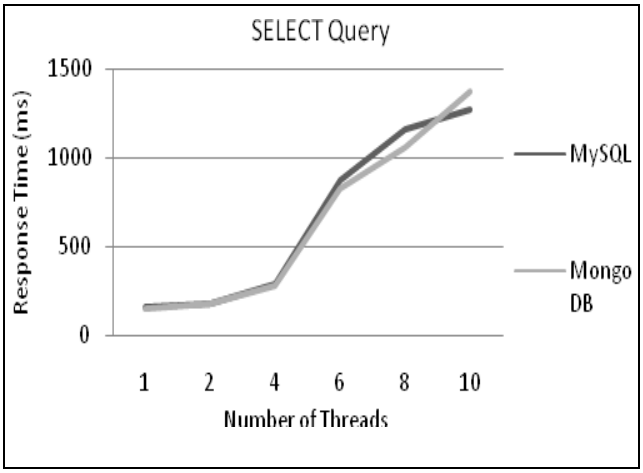

Fig.2. Select query Vs Number of threads

2) INSERT query with varying number of threads:

The graph of response time Vs number of threads for write latency is shown below in fig. 3. For Insert Query operation, the rate of increase of response time for Mysql is more as compared to MongoDB. For insert operation, clearly MongoDB is faster.

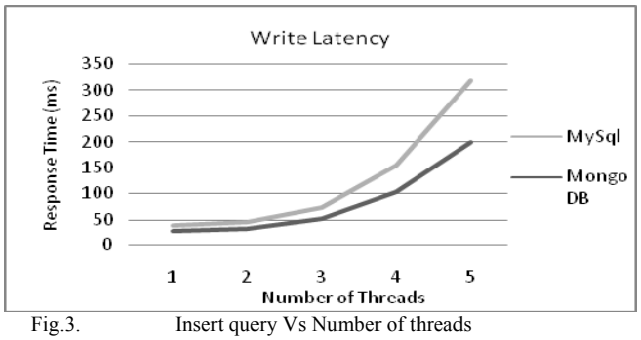

Fig.3. Insert query Vs Number of threads

3) SELECT query with varying number of records:

For SELECT query, graph of response time Vs varying number of records is plotted and is given below in fig. 4. Select query operation for varying number of records shows bit increasing trend line for MongoDB while for MySQL, it is slightly decreasing trend line. For higher number of records MySQL is also stable as compared to MongoDB as we can see drastic changes in response time of MongoDB for 9000 to 10000 records. From graph 1 and 3, it can be said that, for select query, MongoDB response time is more.

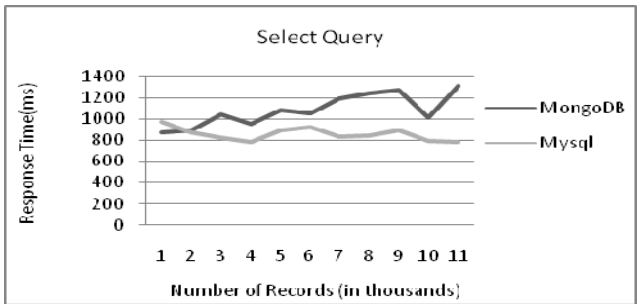

Fig.4. Select query Vs number of records

4) INSERT query with varying number of records:

In the graph shown below in fig. 5, steady line of Mysql response time can be observed. On the other hand, rapid variations in MongoDB are observed. During INSERT operation in MongoDB, variations are observed as sometimes write queue is full and then insert queries are stacked up and response or latency increases. Also, when simultaneous Insert operation is going on, db is locked for that period of time. In this case also, insert can be delayed.

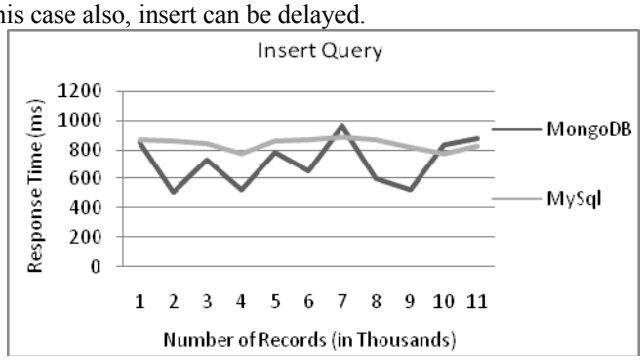

Fig.5. Insert query Vs number of records

VI. CONCLUSIONS

Performance of MySQL and MongoDB databases is compared for the data generated by an Internet of Things application. MySQL which is a traditional SQL database and MongoDB which is a NoSQL database are used to store the IoT data. The comparison study is based on the time taken to execute Select and Insert queries against varying number of records and threads. As, the number of records / size of the table increases, load on the system increases, further increasing the latency or response time. Each database has its own pros and cons. From the study, it was observed that in some scenario, MongoDB required less response time compared to MySQL. But, MySQL responses were stable as compared to MongoDB. Therefore, choosing a better database for IoT depends on which query is mostly used and the requirements of application. Future research run for this work include investigation performed with several other database management systems available with advanced load testing tools. Research can be further extended by using cloud, distributed databases which will make the work more realistic.

References

[1] s. m. riazul islam, daehan kwak, md. humaun kabir, mahmud hossain and kyung-sup kwak1, “The Internet of Things for Health Care: A Comprehensive Survey,” IEEE Access, June 2015.

[2] Satyadhyan Chickerur, Anoop Goudar, Ankita Kinnerkar, “Comparison of Relational Database with Document-Oriented Database (MongoDB) for Big Data Applications,” 2015 8th International Conference on Advanced Software Engineering & Its Applications(ASEA), Jeju, pp.41-47, 2015.

[3] Y. Su, A. Wickramasinghe and D. C. Ranasinghe, "Investigating sensor data retrieval schemes for multi-sensor passive RFID tags," 2015 IEEE International Conference on RFID (RFID), SanDiego, CA, pp.158-165, 2015.

[4] Xu, L. D. Xu, H. Cai, C. Xie, J. Hu and F. Bu, "Ubiquitous Data Accessing Method in IoT-Based Information System for Emergency Medical Services,” in IEEE Transactions on Industrial Informatics, vol.10, no.2, pp.1578-1586, May 2014.

[5] Hammes, Dayne; Medero, Hiram; and Mitchell, Harrison, "Comparison of NoSQL and SQL Databases in the Cloud" SAIS 2014 Proceedings, 2014.

[6] Y. Li and S. Manoharan, "A performance comparison of SQL and NoSQL databases," Communications, Computers and Signal Processing (PACRIM), 2013 IEEE Pacific Rim Conference on, Victoria,BC, pp.15-19, 2013.

[7] Ion LUNGU, Manole VELICANU, Iuliana BOTHA, “Database Systems – Present and Future,” Informatica Economică vol. 13, no.1/2009