



MEASURING PERFORMANCE OF DATABASE SYSTEMS FOR APPLICATIONS OF IOT

K. Samson Paul¹, H. Ateeq Ahmed², A. Emmanuel Raju³

¹Assistant Professor of CSE, Dr.KVSRIT, Kurnool, Email: kakarlasamson1984@gmail.com.

²Assistant Professor of CSE, Dr.KVSRIT, Kurnool, Email: ateeqh25@gmail.com.

³Assistant Professor of CSE, Dr.KVSRIT, Kurnool, Email: ae.raju08@gmail.com.

ABSTRACT

Choosing the right database platform(s) for IoT solutions is daunting. First, IoT solutions can be distributed across geographical regions. As opposed to a centralized cloud-based solution, more solutions are adopting a combination of fog computing at the edge and cloud computing. As such, your database platforms must offer you the flexibility to process the data at the edge and synchronize between the edge servers and the cloud. Second, depending on your IoT use cases, the capabilities you want in your database could range from real-time data streaming, data filtering and aggregation, near-zero latency read operations, instant analytics, high availability, geo distribution, schema flexibility and so on. This article walks you through the four steps in choosing the right database platforms for your IoT solutions. The amount of data stored in IoT databases increases as the IoT applications extend throughout smart city appliances, industry and agriculture. Contemporary database systems must process huge amounts of sensory and actuator data in real-time or interactively. Facing this first wave of IoT revolution, database vendors struggle day-by-day in order to gain more market share, develop new capabilities and attempt to overcome the disadvantages of previous releases, while providing features for the IoT.

KEYWORDS

Database systems performance evaluation, document databases, relational database systems, IoT, IoT Data

1. INTRODUCTION

Internet of Things (IoT) refers to services that are able to sense, communicate and share data. Thus, databases performance capabilities are crucial and significant for the storage and management of IoT data. The variety of today's databases management systems has put users in a big dilemma on which one is the most suitable for each offered IoT service.

Database systems started gaining ground in the 60's. Different types have been developed, each one using its own data representation schema. Initially set as navigational databases based on linked-lists, transformed later on to relational databases with joins, triggers, functions, stored procedures and object-oriented capabilities. In

the late 2000s NoSQL emerged and became a popular trend. The most commonly used database implementations today are based on the relational model which uses SQL as its query language.

Normalization should get rid of whatever is not needed but not at the cost of integrity. De-normalization is the inverse process of normalization, where the normalized schema is converted into a schema which has redundant information.

That way it decreases the number of tables and complicated table joins because a bigger number of joins can delay the process. De-normalized schema can greatly improve performance under extreme read-loads but the updates and inserts become perplexing as the data is duplicated and hence have to be updated/inserted in more than

one places

The primary tasks of IoT services are to acquire, filter, analyze and mine IoT data objects, so as to identify patterns and take appropriate actions accordingly via notifications or triggers. Thus, databases performance capabilities are crucial and significant for the storage and retrieval of IoT data. The variety of today's databases management systems has raised a big dilemma on which one is the most suitable for IoT services. While agents that apply data-mining and deep learning algorithms on IoT data require big memory chunks and CPU processing capabilities for selection queries, since they use database stored procedures and aggregation functions.

2. RELATED WORK ON IOT DATA

Benchmarks of the main business and open-supply databases on Binary Large Objects were tested via way of means of Starcu-Mara and Baumann's. [13]. Experimental situations encompass the open-supply databases of PostgreSQL and MySQL. PostgreSQL model used

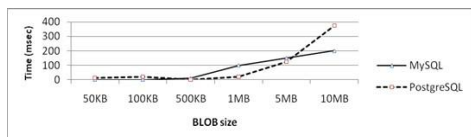


Figure 1. Big data insert queries performance of MySQL and PostgreSQL

8.2.three and MySQL model became 5.0.45. This survey has proven that PostgreSQL had a whole lot higher pick queries overall performance than MySQL on BLOB sizes bellow 5MB. Figure 1. Big facts insert queries overall performance of MySQL and PostgreSQL PostgreSQL as compared to MySQL became now no longer a whole lot greater green for the duration of insert queries for BLOB sizes above 100KB. It became out that MySQL outperformed PostgreSQL in pick queries of BLOB sizes above 5MB. For huge BLOB sizes, MySQL and PostgreSQL confirmed comparable Master-slave scalability performances. The MySQL and PostgreSQL study (pick) and write (insert) overall performance outcomes are proven in Figures 1 and a couple of correspondingly [13]. Considering the look at that has been performed via way of means of the [14], authors used a huge wide variety of information(>one

hundred,000) of most 1KB in report length. They made the studies on MySQL and PostgreSQL databases and from the accumulated outcomes they concluded that MySQL is quicker than PostgreSQL. However, PostgreSQL is quicker in case of concurrency and competition growth for small servicing requests rates (as much as 100req/sec). An e-store internet software evaluation the use of MySQL and MongoDB databases hence has been done via way of means of [2] and has proven that the overall performance of MongoDB became higher while as compared to that of MySQL [2]. Figures three, 4, display the execution time distinction among one hundred numbers of lower back information and 25.000 numbers of lower back information for the duration of a unmarried question for MySQL and MongoDB. The overall performance assessment has been additionally proven throughput (queries/sec) proportional to the information saved or lower back. Figure three. Select-locate queries consistent with 2nd over wide variety of lower back information The outcomes gift that on the burst insert queries test, the MySQL outperforms MongoDB in queries much less than 1MB. MySQL and MongoDB carry out similarly, in queries above 1MB each of them have nearly the equal insert reaction time. For pick queries experimentation, as report sizes growth (greater than 700Kbytes of information sizes facts consistent with transaction) then MongoDB and MySQL gift comparable execution time. That phenomenon occurs for low length transactions (much less than 100Kbyte information sizes. For information of suggest length 100KByte-700Kbyte), MongoDB plays a whole lot higher than MySQL. In conclusion, the pick test suggests that the MySQL database overall performance is worse than MongoDB. 14986004159885Research that has been done via way of means of Fiannaca [4], throughput among the MongoDB and PostgreSQL the contemporary Robot Operation System (ROS gadget) [12]. PostgreSQL done importantly worse than MongoDB and outcomes are proven at Table 2. This isn't particularly regarding on the grounds that MongoDB is made for dealing with JSON facts even as PostgreSQL is designed to control relational facts most effective with extensions for JSON report facts. The ameliorations from relational facts to JSON report facts are time eating while mentioned overall performance.

3. EXPERIMENTS AND RESULTS ON IOT DATA



Performance measurements were performed via way of means of the authors of this paper among relational databases (MySQL 5.6.three and PostgreSQL 9.6) and NoSQL (MongoDB 2.6.10) database. For the cause of this paper, the server used is a P4 at three.2GHz unmarried center PC with 2GB of RAM and a RAID 1 disk array of 120GB. The authors this configuration, due to the fact it's far the minimal month-to-month fee SaaS configuration provided via way of means of the Microsoft Azure cloud, for small companies (\$50/month for a digital device walking on Ubuntu Linux, with 1 center, 2GB RAM, 128GB garage and redundancy and one hundred,000 garage transactions consistent with month). The experimental database server done regionally the use of Python scripts due to the fact authors desired to reduce community delays and jitter. The quantity of concurrent database connections is ready to 2,000 for MySQL, PostgreSQL and for MongoDB. For MongoDB the wide variety of OS open record descriptors is ready to 150,000. During the experimentation, most effective the examined provider (MySQL, PostgreSQL or MongoDB) is the lively provider walking. All database offerings use the equal quantity of reminiscence for a 2,000 max_connections configuration value. MySQL database configuration makes use of InnoDB garage engine, with a pool buffer length of 1,3GB (65% of the to be had reminiscence) to lessen I/O transactions, the use of 512KB of overall study and kind buffer sizes and 128MB of key buffer length. PostgreSQL makes use of 1,3GB of shared_buffers. MongoDB has no reminiscence length limit configuration parameter and makes use of the entire reminiscence in phrases of different offerings. The OS gadget and offerings burn up to 500-700MB of resident reminiscence; for the duration of experimentation, the record reminiscence mappings of MongoDB did now no longer exceed at all of the 1.3GB of reminiscence RAM. Authors used a medium content-length IoT facts obtained from a meteorological station that carries 1-yr measurements for MySQL and PostgreSQL (as much as 570,000 information). The fields that the database has are coming from sensory measurements of time, temperature, humidity, pressure, dew point, rainfall and wind pace and wind direction. All facts are saved as variable char fields and every report length varies from 48-128Bytes of facts. The authentic database became a MySQL database, which the authors migrated to PostgreSQL the use of the pgloader tool [7].

4. CONCLUSIONS

Databases and NoSQL databases. Relational databases dangers relay at the unease design, normalization paperwork and brands for IoT offerings, their barriers on most garage information, and their corruption are procumbent to huge facts that basically calls for the usage of unique type. In this case, effectively restore software program isn't constantly a solution. NoSQL databases are new and emerge as famous particularly designed for IoT, as they offer horizontal schema-much less collections, exceedingly beneficial for IoT facts originated from specific reassets of various structure, sensory hardware and transmission protocols. The relational databases examined via this paper are MySQL and PostgreSQL, in addition to the MongoDB non-relational database. At first a quick literature evaluation has been done specializing in database IoT skills and BLOB facts garage assessment. Then Experimental situations came about the use of IoT sensory facts in 3 specific experimental cases: 1) IoT facts insertion time, 2) IoT agent pick queries execution time and three) IoT agent database aggregation feature execution time. According to the authors' experiments and outcomes, for small wide variety of decided on information PostgreSQL outperforms MySQL and MongoDB. MongoDB plays higher in recognize to MySQL and PostgreSQL for huge wide variety of decided on information. MySQL outperforms higher than PostgreSQL for huge wide variety of decided on information (>20000) however nevertheless can't carry out higher than MongoDB. For insert queries and small quantity of IoT information, MongoDB outperforms MySQL and PostgreSQL, while for huge wide variety of information PostgreSQL gives the least execution time in contrast to MySQL and MongoDB. Aggregation features execution experiments has proven that PostgreSQL is the maximum appropriate database gadget for acting aggregation features on a small wide variety of IoT facts information. On the alternative edge, for an aggregation feature carried out on a huge wide variety of IoT information, MySQL gives the satisfactory overall performance outcomes in phrases of execution time. MongoDB isn't a very good choice for aggregation features execution on IoT facts.

- [1] PostgreSQL (1996)., "PostgreSQL: The World's Most Advanced Open Source



- Relational Database”, Internet:
<https://www.PostgreSQL.org> [Apr. 2010]
- [2] Aboutorabi S., Rezapour M., Moradi, M., and Ghadiri, N. (2015), “Performance evaluation of SQL and MongoDB databases for big e-commerce data ”, In proc. of CSICSSE conf., DOI: 10.1109/CSICSSE.2015.7369245
- [3] Damodaran D. B., Salim S. and Vargese M. V. (2016), “Performance evaluation of MySQL and MongoDB databases ”, International Journal of Cybernetics & Informatics IJCI, Vol. 5, No. 2, ISSN:2320-8430
- [4] Db-engines. (2018), “The DB-Engines Ranking ranks database management systems according to their popularity”, Internet: <https://db-engines.com/en/ranking> [Oct. 2018]
- [5] Fiannaca A. J. and Huang J. (2015), “Benchmarking of Relational and NoSQL Databases to Determine Constraints
- [6] Maksimov D. (2015)., “Performance Comparison of MongoDB and PostgreSQL with JSON types”, Master Thesis, Tallin University of Technology, faculty of Information Technology, <https://digi.lit.ttu.ee> [May 2017]
- [7] MariaDB foundation. (2015), “free MySQL database”, Internet: <https://mariadb.org> [Jun.2016]
- [8] Fontaine D., (2017). “Pgloader tool.” Internet: <https://pgloader.io> [Nov. 2017]
- [9] MongoDB (2012). ”MongoDB document databaseand documentation”,Internet: <https://docs.mongodb.com> [Mar. 2015]