Performance Evaluation between HarperDB, Mongo DB and PostgreSQL

[1]Diana Figueiredo, [1]Gonçalo Saraiva, [1]João Rebelo, [1]Ricardo Rodrigues, [4]Filipe Cardoso, [2]Cristina Wanzeller, [2]Pedro Martins, [3]Maryam Abbasi

[1] Polytechnic of Viseu, Portugal
[2] CISeD - Research Centre in Digital Services, Polytechnic of Viseu, Portugal
[3] CISUC - Centre for Informatics and Systems of the University of Coimbra, Portugal
[4] Polytechnic of Coimbra, Coimbra, Portugal

[1]estgv17365@alunos.estgv.ipv.pt, [1]pv23838@alunos.estgv.ipv.pt, [1]estgv17737@alunos.estgv.ipv.pt, [1]estgv16802@alunos.estgv.ipv.pt, [2]cwanzeller@estgv.ipv.pt, [4]filipe@isec.pt, [2]pedromom@estgv.ipv.pt, [3]maryam@dei.uc.pt

Abstract. Several modern-day problems, like information overload and big data, need to deal with large amounts of data. As such, to meet the application requirements, for instance, performance and consistency, more and more systems are adapting to the specificities. The existing Relational Database Management System (RDBMS)'s the processing of massive data has become an issue because these databases do not deal with a massive amount of data. NoSQL is a database management system that makes processing massive and/or unstructured data easier because it uses key-value to store the data, collections or document stores instead of tables. Many companies today tend to start a project using NoSQL. However, HarperDB aims to produce a relational and non-relational DBMS, allowing developers to choose between different solutions. This paper aims to show the most relevant differences between HarperDB, MongoDB and PostgreSQL and compare their performances. Preliminary results show that PostgreSQL performs better with structured data, but HarperDB can integrate NoSQL and SQL, which can be a significant advantage to HarperDB compared to the other solutions.

Keywords: RDBMS, NoSQL, PostgreSQL, MongoDB, HarperDB, Performance

1 Introduction

The data volumes generated by modern-day systems have met staggering growth during the last few years. Managing and analyzing these data is becoming increasingly important, enabling novel applications that may transform science and society. Thanks to advances in storage technology (being larger and cheaper), more data of customers is being stored in the cloud. The problem is that the data collected is no longer relational, and companies have been adopting a new

structure called NoSQL to meet this new requirement. The Relational database management systems (RDBMS) have issues structuring unstructured data and performance/cost problems in processing massive data [1]. NoSQL performs read/write faster because it has a memory mapping function, making NoSQL suitable for processing big data. In addition, unlike RDBMS, which mainly processes structured data, NoSQL can handle unstructured data more easily [1]. This type of database has been around since the 1960s. However, in the last decade, we have seen market traction and the trend toward databases, such as HarperDB, that are capable of storing relational and non-relational data together [2]. HarperDB has been created to address the complexity and expense of current database systems. This database can be built and scaled without prior knowledge or understanding and without sacrificing performance. It has been marketed as a "true native high transnational NoSQL and SQL" single database solution [2]. With this in mind, in this paper, we will analyze the performance differences between PostgreSQL, an RDBMS, MongoDB, a NoSQL database, and HarperDB, which can work with relational and unstructured data. To perform these tests, we will use the TPC-H benchmark that was initially designed to compare database systems end-to-end. Researchers also use it to benchmark implementation details and algorithms. It consists of a suite of business-oriented ad-hoc queries and concurrent data modifications [3].

This document is organized into five sections. Section 2 presents the state-of-the-art articles comparing the performance between HarperDB, MongoDB and PostgreSQL. The section 3 explains, analyzes and compares the different architectures. Section 4 describes the setup used in the tests. Section 5 shows the obtained results in the tests. Finally, in Section 6 conclusions are drawn, followed by the introduction of future work guidelines.

2 Related work(s)

The geographical data that current systems produce has grown astoundingly during the past few years. Managing and interpreting these data is becoming more and more crucial since it enables novel applications that could revolutionize science and society. The effort to deal with this data flood has shown to be greatly helped by distributed database systems. This article [4] compares the performance in terms of response time between a scalable document-based NoSQL data store MongoDB and an open-source object-relational database system (ORDBMS) - PostgreSQL with the PostGIS extension. PostGIS is a spatial extender that adds support for geographic objects [4]. Spatio-temporal queries that simulate actual case scenarios run against a dataset are used to gauge performance. The evaluation of the systems was investigated under various conditions, including a five-node cluster configuration vs a one-node implementation with and without the use of indexes. Each database system was installed on an Amazon Web Services (AWS) EC2 instance and utilized an Amazon S3 bucket to store and retrieve the data. The authors of this paper came to the following conclusions: PostgreSQL outperforms MongoDB in all scenarios; the average

response time is reduced by half, almost always, with the use of indexes; however, PostgreSQL again reduces the response time by a significant amount; and PostgreSQL reduces the dataset size occupied in the system DB by four times because it stores data more effectively [4]. This paper differs in that, in addition to comparing the performance of the PostgreSQL database with the MongoDB performance. We also compare it with HarperDB.

Big data technology has recently attracted interest from many industries and has received close attention from governments worldwide. Big data is viewed as the "oil" of contemporary civilization, and scientific research has shifted its attention to the value of the information it holds [5]. Before the big data era, data was organized using a two-dimensional table schema in the common format, which is relatively simple to manage and use. The complexity of the data kinds in the big data environment, however, makes it hard to predict how the data will be stored. For this reason, the shift from model-driven research to data-driven research was necessary [5]. When comparing the consumption times for writing the data into each database and the compression ratio, the authors of this paper [5] used remote sensing data as the target data to assess the performance differences between unstructured data in PostgreSQL and MongoDB database technologies. According to the research, MongoDB is six times faster than Postgresql at writing unstructured data, and Postgresql performs significantly better at compression than MongoDB. [5]. In our paper, we will try to compare the performance in both aspects and compare the performance with different sets of write and read between these databases and HarperDB.

3 Architecture

3.1 HarperDB

HarperDB was created because of the complexity and expense of database systems. The main goal of this software is to be built and scaled without prior knowledge or understanding by developers of any skill level, all of this without sacrificing performance [6]. This software can run in every operating system, allowing users to collect, process and distribute data in their company, from hosted servers to the cloud. With HarperDB, it is possible to use conventional SQL operations such as joins, order by, and group by. HarperDB can support document storage, data modelling schema, and queries based on API execution.

3.2 MongoDB

MongoDB was founded in 2007. It is an open-source database built on a horizontal scale-out architecture that uses a flexible schema for data storage. It is an open-source document-based NoSQL datastore. MongoDB does not use tables, rows or columns, unlike the SQL database. In MongoDB, the database is a BSON document, a binary data representation. Applications can access it through JavaScript Object Notation (JSON) files. This is applied in most modern programming languages. This database allows programmers and developers

to store structured or unstructured data since it uses JSON-type files to store documents. Even though MongoDB is a non-relational database, it contains features and functionalities of relational databases, such as sorting and secondary indexing. MongoDB also contains relational database operators such as create, insert, read and update [4].

3.3 PostgreSQL

PostgreSQL is an open-source relational database that uses SQL language combined with features that allow you to scale even the most complicated database schemes. It was created in 1986 as the University of California at Berkeley project. PostgresSQL allows compatibility in several operating systems. It follows the ACID concept (Atomicity, consistency, isolation, durability). These properties guarantee the transactions in the database, and even if errors occur during the transition, PostgreSQL ensures that data is not lost [7].

PostgreSQL has gained an excellent reputation for its architecture, reliability, and data integrity. It can be used on most operating systems, has trendy add-ons, and is used by many people and companies worldwide. PostgreSQL is a good option because it is easy to use, it is open-source, has a big community that can support it when needed, has a user-defined data type and many advantages that make this tool so popular. Nevertheless, it also has some disadvantages. For example, many open source apps support MySQL but may not support PostgreSQL.

3.4 HarperDB vs MongoDB

MongoDB is a document store that stores data in JSON-like documents that can vary in structure, offering a dynamic, flexible schema[8]. MongoDB was also designed for high availability and scalability, with built-in replication and auto-sharding, which is excellent for storing unstructured data. It is a database that optimizes the data value chain for any size company without sacrificing features, functionality and stability. HarperDB is designed to run from the edge to the cloud, and it is a distributed database with a REST API and dynamic schema that supports NoSQL and SQL, including joins. (For example, it is possible to ingest data via NoSQL JSON and then immediately query it via SQL), contains an enterprise-level ACID SQL capability for storing documents. It includes a native REST API that supports SQL over JSON files. MongoDB is more optimized for large-scale writes, but not reads. HarperDB's write algorithm allows large-scale reads and writes, which results in better performance generally.

3.5 MongoDB vs PostgreSQL

MongoDB and PostgreSQL are different types of databases, and both serve different purposes [9]. PostgreSQL is one of the most popular RDBMS (relational database management systems) and is entirely open-source. It can support complex procedures, designs and integrations. It is a SQL database, and it follows

the standard SQL queries. It is mainly used to store data that follows a particular structure. MongoDB is a NoSQL database model often used for unstructured data and application development. It is a document-based database. MongoDB, a No-SQL database, is not relational and can have a dynamic schema. MongoDB gives us the flexibility to change the data schema at any time. MongoDB can handle operational, transnational, and analytical workloads easily. MongoDB must be applied in a use case where it is needed to save unstructured data or if it is needed to handle massive data, and in the future, that same data will keep growing. It is also good if the application is cloud-based.

PostgreSQL is good if the data is structured and follows a specific pattern to perform a significant amount of joins in the application. This database should be used if the application does not have much data. The main difference between those two is that MongoDB is a No-SQL database, and PostgreSQL is a SQL database. In MongoDB, the data is saved as a collection, but in PostgreSQL, the data is held in tables.

3.6 PostgreSQL vs HarperDB

HarperDB scales horizontally, which allows for speed. It has bidirectional table-level data replication and uses a simple pub-sub model; data is replicated by publishing data to different "chat rooms" to which different nodes subscribe and can be distributed horizontally [10]. Otherwise, PostgreSQL scales vertically (as it gets bigger, more space or memory is needed). Therefore, it requires downtime to upgrade. HarperDB and PostgreSQL have enterprise-grade ACID SQL transactions, meaning data validity is quite reliable. The problem with PostgreSQL is that there is not a PostgreSQL cloud-like like HarperDB Cloud and MongoDB Atlas, but some cloud providers offer PostgreSQL-as-a-service. HarperDB is more flexible than PostgreSQL, which is a good option for complex or strict data. On the other hand, HarperDB is simpler to install, configure and administer. It allows developers to use SQL and NoSQL knowledge for the same data model.

4 Experimental setup

In order to do the performance tests of the three different databases, we installed MongoDB and PostgreSQL on a computer to have the exact specifications allowing us to compare their performances, and we installed HarperDB on Docker on the same computer. We did the tests, each one at a time, to ensure that the condition of the tests was the same. This computer has Windows software installed and has the following specifications: Intel i5 processor and 8 gigabytes (GB) of Random Access Memory (RAM). We inserted a dataset of 10 gigabytes in each database, generated with the benchmark TPC-H, and then proceeded with the performance tests. The free version of HarperDB has the limitation of 500 megabytes (MB) of RAM, limiting some tests done on HarperDB compared with the performance tests done in the other databases.

As we mentioned earlier, the dataset of 10 GB was generated with TPCH. TPCH is a benchmark that simulates a Decision Support System database environment. The components of the TPCH database are defined to consist of eight separate and individual tables (the base tables). However, we focused our tests on the lineitem table because it is the one with more data.

For this work, we focus on the insert test, which consists of inserting data into the database and selecting the test where we will execute some queries.

To load the data to PostgreSQL and HarperDB, we used the file generated by the benchmark. However, to load the data to MongoDB, we needed to convert the data to JSON format. To do this, we used a script in NodeJS (Code available in `https://github.com/Sworks99/Conver_to_json`). However, we could not load all the datasets into HarperDB due to his limited RAM (0.5 GB of RAM) on the free version. These will also affect the queries we executed on HarperDB to perform the tests.

After inserting the 10 GB dataset into each database, we selected queries 1, 6 and 5 from the TPC-H benchmark to test the performance of each one of them. As we mentioned earlier, we could not insert all the datasets on HarperDB because of the limitations of the free version. MongoDB has a different syntax, so we had to convert the queries used to the MongoDB syntax. For HarperDB, we had to adapt one of the queries and execute it on each database because of the TPC-H queries. We were getting a timeout when executing on HarperDB. Next, we present the custom query created for each database (PostgreSQL, MongoDB and HarperDB, respectively):

Custom Query on PostgreSQL

```
SELECT *
FROM lineitem
WHERE lineitem.l_orderkey = 599456
    AND lineitem.l_partkey = 1321613
    AND lineitem.l_suppkey = 21614
    AND lineitem.l_linenumber = 3
    AND lineitem.l_quantity = 1;
```

Custom Query on MongoDB

```
DB.lineitem.find({
    "l_orderkey" : "599456",
    "l_partkey":"21614",
    "l_suppkey":"3",
    "l_linenumber":"1321613",
    "l_quantity":"1"});
```

Custom Query on HarperDB

```
SELECT *
FROM AEABD.lineitem
WHERE lineitem.l_orderkey = 599456
```

```
AND lineitem.l_partkey = 21614
AND lineitem.l_suppkey = 3
AND lineitem.l_linenumber = 1321613
AND lineitem.l_quantity = 1
```

So basically, for the select queries, we will execute the three TPC-H queries mentioned before on MongoDB and PostgreSQL and another one on all three databases. For inserting data, we performed two tests: one with around 750 MB inserted into PostgreSQL and MongoDB and another with a file of 10 MB inserted on HarperDB. We did these two tests because HarperDB has the size of the file insert limited to 10 MB. The tests with HarperDB are done for more straightforward queries and smaller datasets.

5 Results and analysis

With all the queries prepared, we proceeded to execute the performance tests. Executing the three TPC-H queries on MongoDB and PostgreSQL, we obtained the following results:

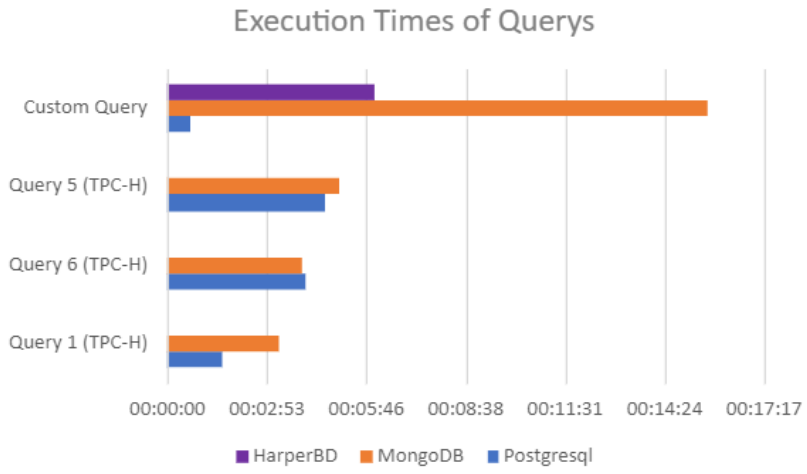

Fig. 1: Execution Times of Querys

As shown in Figure 1 generally, MongoDB takes more time to execute the TPC-H queries, especially on the execution of query 1. On the others, the performances of both databases are similar. With each database's custom query execution results, we can conclude that MongoDB has worse performance than PostgreSQL and then HarperDB and that PostgreSQL has a better performance executing this query.

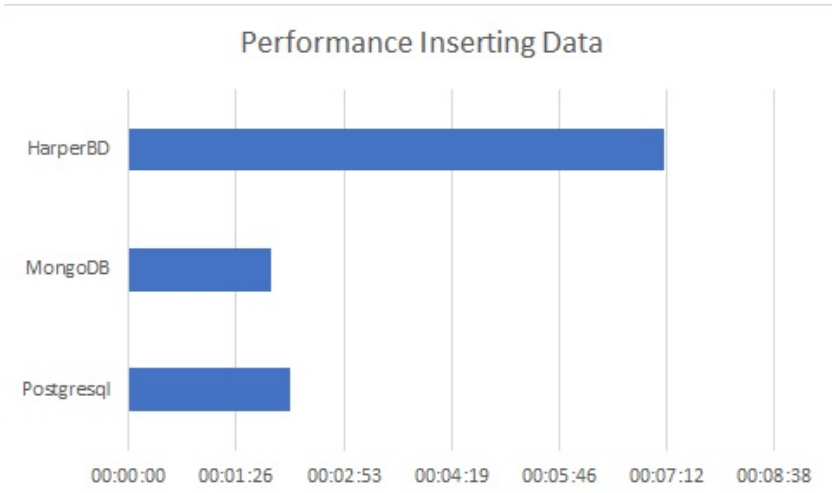

Fig. 2: Results Inserting Data

However, we have to consider that in HarperDB, we do not have all the dataset loaded, so the difference between MongoDB and HarperDB in this query is not that different. With this graphic, we can see that performance improves each time we execute the queries.
With the select queries done, we advanced to test data insertion in each database. For this, we generated the dataset again with the TPC-H benchmark and used the lineitem table with around 750 MB. To insert the data into PostgreSQL, we used the command "COPY" and converted the dataset into a CSV file.
For MongoDB, we again converted the dataset to JSON and used the command "mongoimport" to load the data into the database. With HarperDB, we separated the lineitem CSV file into 10 MB files because it is the limit of upload defined by the database, originating 77 files in total. On HarperDB, we only inserted one of the files because it takes much time to insert each file, as shown in Figure 2.

As we can see in the graphic, MongoDB performs better, followed by PostgreSQL with times of around 2 minutes, with a minimum difference between these two. However, HarperDB uploading only one file takes more than three times more than the other databases. This is a considerable setback for HarperDB if we want to use essential data.

With these results, we can conclude that the database with better performance for structured data is PostgreSQL. MongoDB performance-wise is not that far from PostgreSQL. However, to work with the data and execute complex queries is more complicated and a lot more complex. HarperDB, we consider that is valuable if it is needed to work with NoSQL and SQL, allowing us to work with both in a single database solution.

6 Conclusions and future work

In modern society, nearly every person runs across databases wittingly or unwittingly. For this interaction, people always want a good performance. Database designers are confronted with the task of getting reasonable speed and efficiency. Taking this into account, in this paper, we compared and analyzed the performance between PostgreSQL, MongoDB and HarperDB using the TPC-H benchmark. With the results obtained, we consider that the best database performance of these three is PostgreSQL with MongoDB, with a similar performance in most queries. HarperDB is valuable if it is needed to work with NoSQL and SQL in a single database solution.

6.1 Future Work

Although we presented some results and conclusions on the performance of these databases, we think that analyzing the memory consumption during the query execution could add more insights and obtain better results and conclusions.

Acknowledgements

"This work is funded by National Funds through the FCT - Foundation for Science and Technology, IP, within the scope of the project Ref UIDB/05583/2020. Furthermore, we would like to thank the Research Centre in Digital Services (CISeD), the Polytechnic of Viseu, for their support."

References

1. Sourav Mukherjee. The battle between nosql databases and rdbms. *Available at SSRN 3393986*, 2019.
2. Robert Marsh, Sana Belguith, and Tooska Dargahi. Iot database forensics: an investigation on harperdb security. In *Proceedings of the 3rd International Conference on Future Networks and Distributed Systems*, pages 1–7, 2019.
3. Markus Dreseler, Martin Boissier, Tilmann Rabl, and Matthias Uflacker. Quantifying tpc-h choke points and their optimizations. *Proceedings of the VLDB Endowment*, 13(8):1206–1220, 2020.
4. Antonios Makris, Konstantinos Tserpes, Giannis Spiliopoulos, and Dimosthenis Anagnostopoulos. Performance evaluation of mongodb and postgresql for spatio-temporal data. In *EDBT/ICDT Workshops*, 2019.
5. Yinyi Cheng, Kefa Zhou, and Jinlin Wang. Performance analysis of postgresql and mongodb databases for unstructured data. In *2019 International Conference on Mathematics, Big Data Analysis and Simulation and Modelling (MBDASM 2019)*, pages 60–62. Atlantis Press, 2019.
6. HarperDB. Harperdb. `https://harperdb.io/`. Accessed: 14/05/2022.
7. Felipe Quillici and Leonardo Schiavon. Uma comparação entre sistemas mongodb e postgresql: Comparação entre sistemas mongodb e postgresql. In *Congresso de Tecnologia Fatec Mococa*, volume 3, 2021.

8. Margo McCabe. Which database is right for you? harperdb vs. mongodb vs. postgresql. https://hackernoon.com/which-database-is-right-for-you-harperdb-vs-mongodb-vs-postgresql-8m6a35uf. Accessed: 10/06/2022.
9. Naivskill. Mongodb vs postgresql: Complete comparison in 2022. https://naiveskill.com/mongodb-vs-postgresql/. Accessed: 17/06/2022.
10. stackshare. https://stackshare.io/stackups/harperdb-vs-postgresql. Accessed: 18/06/2022.