Radboud University Nijmegen

Faculty of Science

Database energy benchmarks: an evaluation

Thesis BSc Computing Science

Author:
Anne den Hartog

Supervisor:
Bernard van Gastel

Second reader:
Djoerd Hiemstra

12-01-2024

Contents

1 Abstract

This thesis examined the use of benchmarks in evaluating the energy efficiency of two databases: PostgreSQL and MySQL. Using these results to look at how small to medium businesses can use the metrics generated by benchmarks to make energy-conscious database choices. Experiments were conducted by running the TPC-C benchmark on the two databases with variations in scale and duration. The results show that PostgreSQL can do about 6x as many transactions for the same amount of energy as MySQL under the TPC-C benchmark. However, benchmarks are limited in which ways they can inform buyer decisions since they do not reflect the real world well. This is because performance benchmarks (like the TPC-C) aim to maximize the performance of the database, while energy consumption needs to be minimised. This means that a benchmark is limited in informing business decisions since the benchmark tests maximum (peak) capacity, but a typical business is not always operating at full load.

2 Introduction

2.1 Context

From websites to employee management, IT systems are indispensable to modern businesses. System choice is often judged on reliability, performance and cost. The global rise of sustainable businesses and rising energy costs have added another important factor: energy consumption. At the core of an IT system, the database takes care of managing and saving the data. Data is necessary for the daily operations of all companies. Therefore a database is crucial for the smooth operation of a system and consequently a business. It handles a lot of logistics without the intervention of the application designer. Thus choosing an energy-efficient database can impact the energy consumption of the entire system. But how can you measure how energy-efficient a system is?

Benchmarks are a tool that could offer a solution. Benchmarks are seemingly objective ways to gain insight into the performance of a database. Thus they are used extensively to inform database choice for businesses in terms of performance [16]. Within the realm of research into energy consumption, benchmarks also are the primary tool used to generate workloads to evaluate proposed optimisations [6, 5]. This is done by comparing the energy use of a database under the benchmark with or without the optimisation.

To make sustainable decisions during system design, developers need to know the energy efficiency of the software they're considering. To achieve that, accurately modelling the workload a system will be under is crucial, possibly using benchmarks. Since that would create some standardized information on the energy consumption of software. Possibly moving to an energy labelling system somewhere in the future. This would allow developers to make informed choices on the environmental impact of the systems they design. In addition, a labelling system like this could inform developers and users that software choice has an impact on energy consumption [17]. Furthering the ability of people to work on sustainable goals in all aspects of their business.

2.2 Goal

The goal of this thesis is to investigate the extent to which performance benchmarks can be used to inform sustainable business decisions. Specifically addressing the question: *How can a benchmark provide metrics to help small to medium businesses choose a database based on energy consumption?* By exploring this question, this thesis aims to contribute insights into the relationship between energy efficiency, database performance and how to benchmark that. Ultimately, allowing informed decision-making for sustainable IT system design.

2.3 Scope

Comparing the two most popular and widely used open-source databases: MySQL and PostgreSQL. This thesis focuses on assessing the use of energy benchmarks for small to medium enterprises (SMEs). These make up the vast majority (99%) of the existing businesses in OECD countries [12]. These businesses should be able to make sustainable choices in their IT systems. Changing the habits of the biggest users allows for the biggest impact.

2.4 Contributions

This thesis will produce:

1. A reusable test set-up that measures the energy consumption of databases under a workload generated by a benchmark.

2. An evaluation of benchmarking as a tool for energy measurements. Giving an exploration of the results in the context of a small to medium business.

2.5 Approach

First, this thesis will give preliminaries about the two open-source databases and benchmarks in general. This will explain some theoretical differences such that the results can be considered in context. It further provides insight into the history of benchmarks and database evaluation (Chapter 3). The next chapter is a global overview of the related work that has been done in the field of database energy measurements and how benchmarks are commonly used there (Chapter 4). From this follows the experimental set-up which will be used to perform the repeatable energy tests using a benchmark. The set-up needs to be repeatable since the updates in programming languages and systems mean that results for one version don't necessarily translate to newer versions (Chapter 5).

The experiment will give information on what the energy consumption of databases is like under the benchmark. The results from the experiment will provide the springboard from which the evaluation can continue. Comparing the two databases, which are both wildly used by small to medium businesses, provides the context needed to evaluate benchmarks as a tool. The results will be shown using both visualisations and by highlighting which parts of the results are interesting in the context of this thesis (Chapter 6).

The evaluation will then further explore the results and their usefulness in informing energy decisions. There will be considerations on what the results mean in the context of a small to medium business and how the results can or cannot be used (Chapter 7). Lastly, this thesis will finish with the concluding thoughts and recommendations for future work paying special attention to new questions raised by the results (Chapter 8).

3 Preliminaries

3.1 Databases

There are two open-source databases up for consideration in this thesis: MySQL and PostgreSQL. These two databases were the most popular in the 2023 Stack Overflow developer survey, with PostgreSQL getting first place with 45.55% of the votes and MySQL in second place with 41.09% [13]. The popularity of these two databases is the reason why they were chosen for this thesis. Since the scope encompasses the most popular use case; an OLTP (online transaction processing) system for small to medium businesses.

The two databases share more characteristics. Both became popular open-source projects in the late 90's and have an enthusiastic community of developers that have been extending the libraries for over 30 years [4, 11]. As opposed to an option like SQLite, PostgreSQL and MySQL have a client/server model which means they can be reached over a network, also both ensure ACID properties [4, 11]. ACID (Atomicity, Consistency, Isolation, and Durability) properties are safety guarantees that describe fault tolerance in database systems. The exact interpretation of these properties varies per implementation, thus while the general idea behind these concepts holds value, the specific meaning of *'Isolation'* for example is not universal [9].

3.1.1 MySQL

MySQL was first released to the public in 1996 to solve some specific problems with data management. It has since grown to be a massive open source project, that is used by many companies of different sizes [11]. MySQL resists efforts to give a formal definition of its architecture since it is mainly a collection of functions that the creator needed for their specific use case. Due to the insight with which these functions were written, the system could still evolve into a complete database server. Any categorisation that exists today was thought of after the creation of the system. Developers familiar with MySQL tend not to think about MySQL as a whole, but more like a collection of useful functions, directories etc [14].

3.1.2 PostgreSQL

PostgreSQL (more commonly known as postgres) is also an open-source system that was developed in 1989 and has been in active development ever since [4]. The development of PostgreSQL started at the University of Berkley with multiple papers describing the internals under the name POSTGRES. Over the years the system evolved through research at Berkley most of which is documented in published papers. In 1995 the system was adjusted to adhere to the SQL standards and PostgreSQL was born as we know it today: an extensive open-source database [4].

3.2 Benchmarking

Performance has been one of the main factors in influencing businesses' buying decisions for decades [16]. This necessitates some form of standardised testing such that different databases can be compared. Benchmarking is one of the main tools used for deciding which database is better [1]. Two main authorities focus on creating benchmarks and publishing official results, the TPC (Transaction Processing Performance Council) and SPEC (Standard Performance Evaluation Corporation). Especially the benchmarks published by the TPC are seen as industry standards for comparing databases and are often used in research [1, 2]. But in the past few years, partly due to how involved the

implementation is, many new databases have adapted the benchmarks such that they can make marketing claims [16]. Circumventing the original purpose of the benchmark, which was to publish controlled standard metrics to a central library. Which would give buyers a fair platform to compare different databases. Adjusting the benchmark is not always unjust, since it has been shown that the TPC benchmarks are less suited to NoSQL databases. This means that relational databases have an unfair advantage when tested by these benchmarks [8].

4 Related Work

4.1 Energy-efficiency in databases

Tsirogiannis et al. analysed how energy was consumed by a database server. They mapped out how much energy all specific hardware parts of their test server consumed. They then used a variety of testing workloads and queries to determine that in the vast majority of database tasks "the most energy-efficient configuration is the highest performing one" [23].

Schall & Harder took a different approach and designed a new database that considered energy-efficiency as its main metric of optimisation. They also showed which parts of a database can be changed to increase energy-efficiency in general [19].

Guo et al. designed additions to existing databases such that their query optimisation and processing take energy use into account. This means that the expected energy consumption is modelled and used to inform query decisions. This technique is to extend databases that are already in use to also include energy-specific optimisations [5]. There are many avenues in research for improving the energy efficiency of databases, of which some of the main areas are highlighted here [6].

However, interest in green databases has waned over the last couple of years. In a literature review about "Energy Efficient Database-systems" published in 2022 only about 25% of the research considered was done after 2016 [6]. This means that a large part of the literature was published in the first half of the time frame the review considers. Despite this apparent lack of interest 'green databases' were indicated in a literature review on green IT as an area that required more research and attention anyway [15].

4.2 Energy benchmarking

Rivoire et al. made one of the first ventures into measuring the energy consumption of a DBMS using a benchmark. They proposed a balanced benchmark that could inform the best hardware and software choices for data centres and smaller users. Running this benchmark showed that focusing on energy-efficiency gives rise to a unique combination of characteristics in hardware and software that cannot be achieved otherwise [17].

Between 2008 and 2012 the TCP in collaboration with SPEC developed the TPC-Energy specification. The TPC-Energy is called a common benchmark since it can be added to all existing TPC benchmarks [22, 16]. When this specification was published it was not without criticism, namely that idle use was not well represented. They further proposed additions to the specification such that it would better represent real-world use [18]. Since then, different TPC benchmarks have been used extensively in research to simulate workloads, mainly to evaluate proposed energy use improvements [6].

Jin et al. started working on a benchmarking platform for the energy consumption of databases, indicating the need to have an updatable source of information on the energy use of different databases [7]. The platform they describe in their paper has since disappeared.

Liu et al. considered the energy use of different databases for edge computing, they compared fundamentally different databases (SQLite, LevelDB, MongoDB). They used a workload generated by another benchmark namely the YCSB (Yahoo! Cloud Serving Benchmark) since they were investigating a cloud set-up [10].

5 Experiment Design

5.1 Design and Implementation

This section describes the repeatable test set-up that is listed as one of the contributions of this thesis. It includes considerations for benchmark, database and code suite choices.

5.1.1 Benchmarks

Benchmarks are used in the industry to test the performance of a system against a standardized workload that tries to represent real use. Thus running a workload using specifications set by the benchmark designers makes sure that the workload is repeatable and representative. One of the creators of such benchmarks is the TPC (Transaction Processing Performance Council), which has been creating benchmarks that have become industry standards since 1988[20].

In this thesis the experiments will be run using the TPC-C benchmark, which was designed by the TPC. The TPC-C simulates a workload used by OLTP systems (online transaction processing); like online web shops. The scale of the OLTP system can be adjusted in the benchmark to represent the size of the business[21].

To run this benchmark a testbed called Benchbase will be used. Benchbase (formerly known as OLTPbench) was built to save others' time by taking care of the implementation of a variety of benchmarks. It takes care of generating the workload according to the benchmark specifications. Benchbase was first published by Difallah et al. in 2013 and has since grown as an open-source project [3]. It includes many different benchmarks and support for multiple relational databases.

5.1.2 Databases

The TPC-C benchmark was originally designed for relational databases, which is not surprising since it was first published before 2000. With the rise of NoSQL databases, the specification now indicates that they can be adjusted to run on other types of databases (e.g. document, graph) [21]. But to do this fairly requires quite a bit of effort [8]. In addition, benchbase only implements benchmarks for relational databases [3], introducing another code suite to run the benchmark on other types of databases would introduce another variable that disallows comparisons. For these reasons, only relational databases will be compared. The two databases are PostgreSQL and MySQL.

The complexity of understanding the general structure and architecture of these databases is a field of work that developers specialize in. In the scope of a small to medium business, it's not likely that such a developer is present. Thus in this experiment, the databases will be used 'out-of-the-box' since realistically users will not have the know-how to optimize based on their specific workload.

5.2 Experiments

The measurements are running on an ODroid H3 with 32 GiB of RAM and an M2 SSD which is connected on both sides to an ampère meter (the INA-260). The energy is measured by calculating the difference between Joule coming in and Joule coming out. A gitlab runner then starts up a docker image and runs the database as a GitLab service. Then a container running benchbase runs the benchmark on the database service. From the moment the benchbase container starts, energy metrics are being recorded every second.

- For the basic experiment, the TPC-C benchmark will be running for 60 seconds on both databases using 1 warehouse. This will form the baseline for comparison between the two databases.
- Then the TPC-C is run for different intervals 10, 30, 60, 120, and 240 seconds. This is to investigate the effect of caching and how duration affects the performance of a database under a benchmark.
- The TPC-C benchmark is run using 1, 5, and 10 warehouse(s) (1-100 is considered a small to medium business by the TPC) for 60 seconds on both of the databases. This is to test the influence of scale on energy consumption. A warehouse is just a name for indicating that the database is x times as big. Thus 5 warehouses are 5 databases of size 1 pasted together. As shown in Figure 1.
- After the benchmarks are run there will also be energy measurements for 60 seconds while the system is idle.

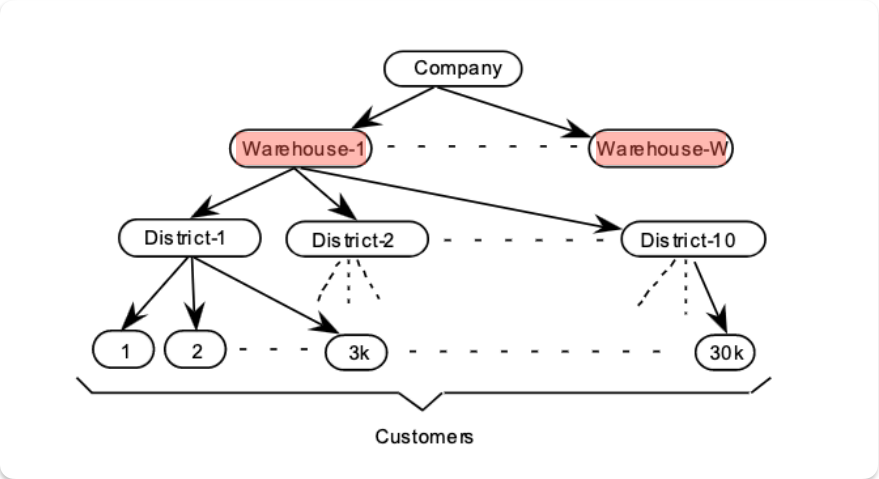

Figure 1: Structure database TPC-C with multiple warehouses [21]

5.3 Validity

Both the container for the database and the container running the benchbase image are running on the same device, while the measurement is done on the whole device. This means that not just the database is being measured, also the container running the benchmark. So the results can be compared to each other (measurements done in the same set-up), but should not be considered independently.

Tuning of the databases can lead to drastic increases in performance (and decreases in energy use) [8], so to keep things fair this research will not tune any of the considered relational databases. Thus this research will only consider which 'out-of-the-box' database is more energy efficient under the benchmark.

The scope of the study was chosen such that the test set-up represents the situation in actual use. But, finding out what a 'typical' server set-up is for a small to medium business is a research topic of its own, so there will most likely be some differences.

6 Experiment Results

The experiments were done to give a starting point for the evaluation of benchmarks as a metric. The foundational experiment mostly focuses on the comparison between PostgreSQL and MySQL. The next two experiments give more information on the properties of the benchmark since the databases remain the same but the benchmark changes. First some background information on the unit used to consider the results. Then the results of the three experiments will be discussed in the following order: the foundational experiment, variations in duration, and variations in scale.

6.1 The unit

The TPC-Energy specification provides the unit with which the results of the experiments can be interpreted. According to the TPC-Energy: *"The primary metric reported as defined by TPC-Energy is in the form of 'Watts per performance'"[22].* These 'performance units' are unique for each TPC benchmark. For example, the TPC-C returns the metric tpmC (transactions per minute) [21]. Watts are equal to Joule per second, so to reach the metric "Watts per performance" the Joule and number of transactions are measured for the same amount of time. By equalizing the time elements on both sides of the unit, the result is the unit Joule per transaction (J/tC). Giving rise to the equation:

$$\text{Watts per perfomance} = \frac{Joule}{Transactions} = \frac{J}{tC}$$

This unit allows the results to be generalized across the different experiments. It makes sure that the results can be compared to each other.

6.2 The foundational experiment

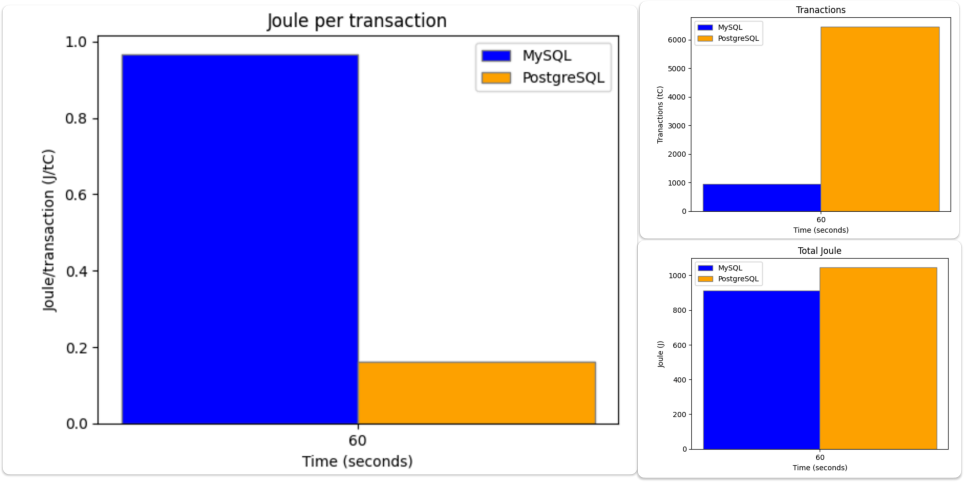

Figure 2: Benchmark is run for 60 seconds, figures show Joule per transaction, transactions and Joule

In this experiment, the benchmark was run for 60 seconds, during which the energy was measured. As shown in figure 2 PostgreSQL outperformed MySQL by a large

margin. Both databases used roughly the same amount of energy, with MySQL using a small bit less. However, within that energy limit, PostgreSQL had more than 6 times the amount of transactions in the same amount of time. When calculating the amount of Joule used per transaction that means that PostgreSQL only uses 0.14 J/tC and MySQL about 0.90 J/tC.

These results speak for themselves, even though the energy consumption is almost equal between the two databases PostgreSQL can do so many more transactions that it completely outperforms MySQL. Consequently, for any company that has a workload of more than 2000 transactions per minute, PostgreSQL is the most energy-efficient option. When the database is not running and the system is idle, both databases use the same amount of energy.

6.3 Adjustments in duration

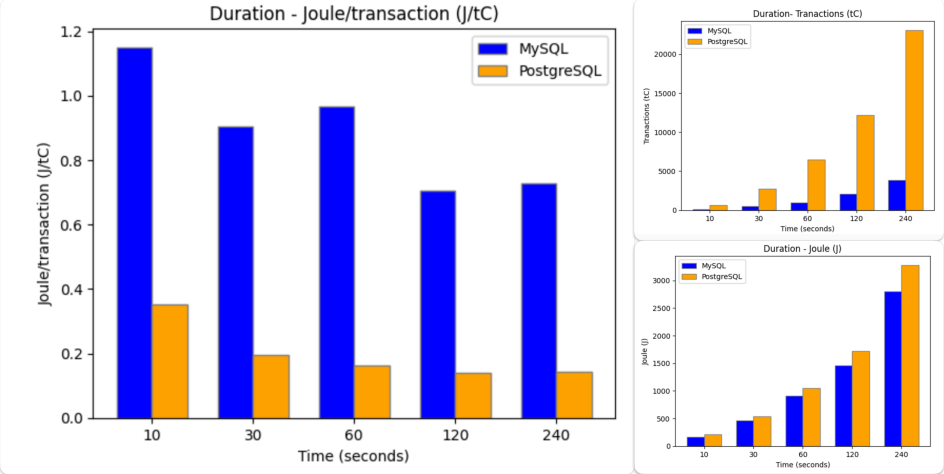

Figure 3: Benchmark is run different durations, figures show Joule per transaction, Transactions and Joule

The second experiment aimed to map out how the duration of the benchmark affects the performance. This was to investigate how the databases perform when the benchmark was run for different durations. It gave insight into how optimisations like caching influence the performance. Figure 3 reports the joule per transaction when the benchmark was run for 10 to 240 seconds.

Interesting here is that both MySQL and PostgreSQL used fewer Joules per transaction when the duration of the benchmark was longer. The databases became more energy-efficient, the more transactions they could perform. Since the TPC-C workload runs similar transactions many times, the effect of caching or other optimisations can have quite a big impact. For example, PostgreSQL was almost 3 times as efficient when the benchmark was run for 60 seconds (0.14 J/tC) compared to 10 seconds (0.39 J/tC).

It's good to note however that the improvements for both database's performance stagnate at some point. Since at some point, all versions of the queries have been performed once, the optimisations have no further effects. For PostgreSQL this happens at 0.14 J/tC when running for 60 seconds, running for longer doesn't provide any further improvements. The same goes for MySQL, which stagnates at 0.68 J/tC at 120 seconds.

Furthermore, PostgreSQL most likely benefits a lot from its own efficiency under the benchmark. Since it's able to perform many more transactions than MySQL in a short time it can quickly reap the benefits of these quick cached transactions. While MySQL takes a longer time to achieve this extra cache efficiency since it doesn't have as many transactions. This further exacerbates the difference in performance between PostgreSQL and MySQL.

6.4 Adjustments in scale

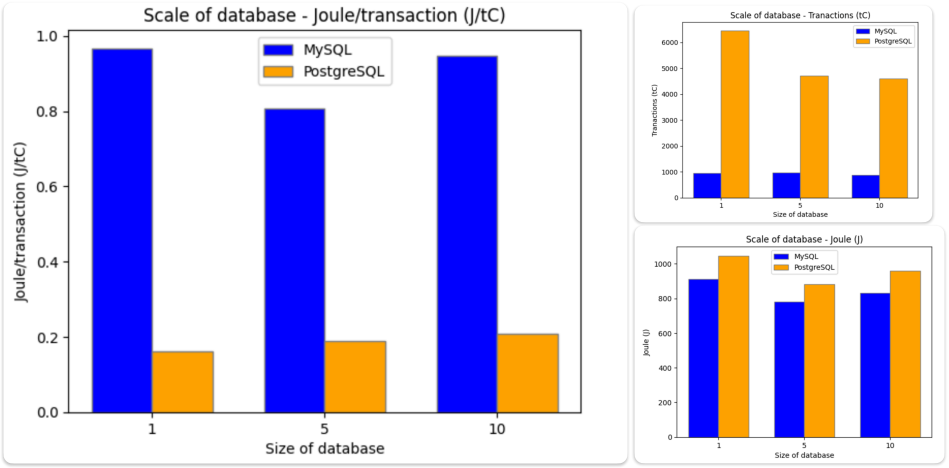

Figure 4: Benchmark is run for 60 seconds for different database sizes, figures show Joule per transaction, transactions and Joule

To measure the effect of scale the tests were run on 1, 5, and 10 warehouses. Which means that the database gets made 5 and 10 times bigger than the one in the first experiment respectively. In general, the scale did not show differences in the same way the previous experiments did. The difference between PostgreSQL and MySQL is similar as in the previous tests and scale does not negatively affect the performance of the databases in a big way. While idle energy use is higher for the bigger databases, the differences were not that big. For PostgreSQL increasing the scale did reduce the amount of transactions by about 30%, but the performance remains miles above MySQL.

7 Evaluation of benchmarks for standardizing energy measurements

In existing literature, the use of benchmarks as an evaluation tool is almost an assumption. *"To test and evaluate the energy efficiency of a DBMS server, a complete set of benchmark is needed."*[5], the benchmark used in this paper is also designed by the TPC, which reflects the general trend in other research as well [6]. While there has been some criticism on how the TPC benchmarks are designed [19, 18], this was mainly focussed on how it dealt with idle use but not the entire benchmark. Over the years these criticisms did not reduce the use of TPC benchmarks in research.

Research into energy consumption is inherently practical, the goals are driven by the need to reduce energy use. In this sense, most research relates directly to the real world through necessity, which is why the benchmarks used to measure these results should reflect the actual use as well as possible. This is not always the case, for example, the result that the most energy-efficient DMBS is the highest performing one [23] can also be attributed to the use of performance benchmarks to run the tests according to Schall and Härder[19].

Transitioning to applying this critical thinking to the results from the experiment, leads to some questions. The obvious conclusion from these results is that someone should choose PostgreSQL if they wish to conserve energy. PostgreSQL performs way better considering the metrics provided by the TPC-Energy. This result disproves the idea that for a database to process more transactions, it must also use more energy. So then; why is MySQL still one of the most popular databases, when it performs so much worse than PostgreSQL?

This prompted a reevaluation of the methodology's reliability. It's crucial to evaluate how the results came to be before drawing conclusions about them. To start figuring that out, first some information on what a performance benchmark is supposed to do.

7.1 What do benchmarks aim to do?

Benchmarks are used to standardize loads across databases that test where the boundaries of performance lie. When all competitors are pushed to the edge of their abilities you can make relatively fair comparisons between databases. This highlights the strengths and weaknesses of DBMSs and allows you to move beyond what all databases are at least 'okay' at; storing data. It helps users explore performance bottlenecks, compare different DBMSs and collect 'standard' metrics to make comparisons with real use[3]. In summary, a database performance benchmark gives boundaries to the *extent* of a DBMS's abilities.

When trying to test the energy consumption of a database you have a different goal in mind, you aim to minimise the amount of energy your system is using. The benchmark should test whether the database makes energy-efficient choices for the typical workload it handles. In addition, it needs to give an indication when the exact workload is not clear yet, but there is an estimation. But generally, a small to medium business will not stretch the database to its limits, only in very exceptional circumstances will it reach the limits of performance. And if that's the case, wouldn't it be time to scale up? Thus using a performance benchmark to generate the workload to test the energy consumption is not as straightforward as it seems.

Think of buying a car; fuel efficiency is an important factor that informs buyers' decisions which needs to be measured. If the car company were to test the fuel efficiency on a race track driving 200km/h it would report a very bad fuel efficiency. Whilst if the car only drove 60km/h the test would report an amazing fuel efficiency for the same car. Both tests present a distorted view of reality, which means that fuel measurements

are not relevant to the situation they will be used in. When reporting on fuel efficiency it's important to accurately represent real use, otherwise the results do not present any valuable information to the buyer.

This metaphor illustrates the same issue that running energy measurements using performance benchmarks has. When running a benchmark you try to maximize performance, whilst you try to minimize energy consumption. When making energy measurements the workload must represent real-world use. Otherwise, it's difficult to use your results to inform buyer decisions

7.2 In the context of the experiment

To get back to the results at hand, this context allows us to ask the question: "Is PostgreSQL better because it has more transactions for the same amount of energy?". From the arguments laid out above it follows that this type of experiment cannot answer that question. To answer questions in regards to energy consumption you must represent the real-world conditions in your workload generation. Energy consumption is entirely dependent on the situation in which the systems are used.

The difficulty lies with the performance benchmark, using this benchmark you cannot answer the question: "What if PostgreSQL was only able to perform 900 operations in 60 seconds?". Because *inherently* a performance benchmark explores the extent of a database's abilities. This does not allow for running tests that aim to mimic real-world use. Consequently, the results require too much critical thinking to be used as a base metric for database choice.

The weakness of using performance-oriented benchmarks for energy measurements is further demonstrated by the results from the duration experiment. Some of PostgreSQL's good results can be attributed to quick caching. This accounts for good results during the benchmark, but it would not have this profound an impact during regular use. When you consider long-term use this advantage will be less present since both databases will be able to run using optimal caching after some time of use. This means that in the long term, the effects of efficient caching will most likely even out across the two databases. This does not mean that they will perform at the same level, just that *using a benchmark* gives a distorted view of what will happen.

7.3 What CAN a performance benchmark tell you?

All tools trying to simulate the real world are imperfect. The issue with using performance benchmarks for energy measurements is that they present the image of an 'objective' tool of evaluation. But if the results given by a test cannot be evaluated at face value, it's not a good test. At this point, the results require too much critical thinking to be interpreted well.

However, the results can still be applied to the benchmark and the same goes for all research that has been done using this method. Under the benchmark PostgreSQL completely outperformed MySQL. So even though applying this to a real-world scenario is limited, the results still stand. For example, the TPC-C can be seen as a tool to compare optimisations regarding the energy consumption of software. It just cannot tell you what effect the optimisations would have if they were implemented practically.

Thus performance benchmarks are unsuited for doing energy measurements to inform the decisions of those that build an OLTP system for a small to medium business.

8 Conclusion

Throughout this thesis, I have investigated in which ways benchmarks can and cannot be used to model the energy consumption of the two most popular databases; MySQL and PostgreSQL. First elaborating on which information can be gleaned from the results of the experiments. And then focussing on how these metrics can and cannot inform small to medium businesses in making energy-conscious database choices.

There has been research in developing benchmarks for measuring energy efficiency [17, 18]. However, using the benchmarks developed by the TPC (TPC-C and TPC-A) to generate a workload remains the main choice in almost all research [6]. The experimental results indicate that there is a big difference between the energy consumption of PostgreSQL and MySQL when running the TPC-C workload. In which PostgreSQL comes out as the clear winner in terms of energy efficiency. However, if one wishes to use these results to inform business decisions there are some caveats.

The use of performance benchmarks to generate workloads for energy measurements has some inherent limitations. Mostly explained by the fact that a benchmark like the TPC-C maximises the performance, while businesses are trying to minimize energy use. This means that a benchmark is limited in informing business decisions since the benchmark tests maximum (peak) capacity, but businesses tend to have a stable load that is far from the maximum. Therefore, while PostgreSQL may show superior energy efficiency under the benchmark, its real-world performance may vary based on the workload generated by a small to medium business. Businesses seeking to make energy-conscious database choices should consider the specific workloads and usage patterns relevant to their day-to-day operations. In doing that, they recognise that benchmark results provide a valuable but limited perspective on the overall energy performance of databases.

Thus a performance benchmark can be a part of the solution for designing an energy labeling system, but it can not be the only metric. An energy consumption label in particular should consider the workload of the most common user for it to be relevant and accurate. Such a label can be an effective tool in aiding sustainable decision-making only when taking a comprehensive approach to designing it.

Allowing businesses to make more sustainable choices will go a long way in reaching sustainable development goals. This thesis investigated one method for preemptively evaluating databases, such that sustainability can be one of the deciding factors next to cost and efficiency. It evaluated a standard practice within this research field and discussed its practical limitations. Adding nuance into how we should evaluate the energy-efficiency of databases in a very practical sense.

8.1 Future Work

This thesis raises some interesting questions opening up avenues for future work. Mainly: "If benchmarks are limited, what would be the proper way to measure the energy consumption of software and hardware products?". In the following section, some of these avenues are highlighted.

To build an accurate energy benchmark, one area that first needs to be investigated is what a 'typical' workload looks like for the most common business. This question is important as it addresses the variability in workloads among different websites. Understanding what constitutes a typical workload can help create standardized benchmarks for energy consumption.

Once this is mapped out it's possible to continue developing an easy-to-use benchmark that is focussed on measuring energy consumption. Paying attention to why an energy-specific benchmark like the one developed by Rivoire et al. is not used, but the

TPC benchmarks are. Since it's no use developing a new benchmark if it's not going to be used.

Another thing to investigate is what other options there are outside of benchmarks to map out the energy consumption of database systems. Exploring alternative options beyond benchmarks can provide a more comprehensive understanding of energy usage in different scenarios.

The energy-efficiency of the databases under the benchmark can also be explored further. For example, by running a similar test as was done in this thesis by using multiple benchmarks or adding more databases like NoSQL databases or SQLite. This would move beyond the scope of a small common business and give further insight into how these databases, benchmarks and energy consumption relate.

References

[1] Yijian Cheng, Pengjie Ding, Tongtong Wang, Wei Lu, and Xiaoyong Du. Which category is better: Benchmarking relational and graph database management systems. *Data Science and Engineering*, 4(4):309–322, 2019.

[2] Murilo R. de Lima, Marcos S. Sunyé, Eduardo C. de Almeida, and Alexandre I. Direne. Distributed benchmarking of relational database systems. *Advances in Data and Web Management*, page 544–549, 2009.

[3] Djellel Eddine Difallah, Andrew Pavlo, Carlo Curino, and Philippe Cudré-Mauroux. Oltp-bench: An extensible testbed for benchmarking relational databases. *PVLDB*, 7(4):277–288, 2013.

[4] PostgreSQL 16.1 Documentation. Preface. `https://www.postgresql.org/docs/16/preface.html`, 2023. Accessed on: 17-12-2023.

[5] Binglei Guo, Jiong Yu, Bin Liao, Dexian Yang, and Liang Lu. A green framework for dbms based on energy-aware query optimization and energy-efficient query processing. *Journal of Network and Computer Applications*, 84:118–130, 2017.

[6] Binglei Guo, Jiong Yu, Dexian Yang, Hongyong Leng, and Bin Liao. Energy-efficient database systems: A systematic survey. *ACM Computing Surveys*, 55(6):1–53, 2022.

[7] Yong Jin, Baoping Xing, and Peiquan Jin. Towards a benchmark platform for measuring the energy consumption of database systems. *Interdisciplinary Research Theory and Technology*, Nov 2013.

[8] Asya Kamsky. Adapting tpc-c benchmark to measure performance of multi-document transactions in mongodb. *Proceedings of the VLDB Endowment*, 12(12):2254–2262, 2019.

[9] Martin Kleppmann. *Chapter 7: Transactions*, page 221–232. O'Reilly, 2017.

[10] Jian Liu, Kefei Wang, and Feng Chen. Understanding energy efficiency of databases on single board computers for edge computing. *2021 29th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)*, 2021.

[11] MySQL 8.0 Reference Manual. 1.2 overview of the mysql database management system. `https://dev.mysql.com/doc/refman/8.0/en/what-is.html`, 2023. Accessed on: 17-12-2023.

[12] OECD. Enterprises by business size (indicator). doi: 10.1787/31d5eeaf-en, 2023. Accessed on: 06-12-2023.

[13] Stack Overflow. 2023 developer survey. `https://survey.stackoverflow.co/2023/#most-popular-technologies-database`, June 2023. Accessed on: 17-12-2023.

[14] Alexander Sasha Pachev. *MySQL History and Architecture*, page 1–19. O'Reilly, 2007.

[15] Showmick Guha Paul, Arpa Saha, Mohammad Shamsul Arefin, Touhid Bhuiyan, Al Amin Biswas, Ahmed Wasif Reza, Naif M. Alotaibi, Salem A. Alyami, and Mohammad Ali Moni. A comprehensive review of green computing: Past, present, and future research. *IEEE Access*, 11:87445–87494, Aug 2023.

[16] Meikel Poess. Tpc, where art thou? *Datenbank-Spektrum*, 22(3):241–248, 2022.

[17] Suzanne Rivoire, Mehul A. Shah, Parthasarathy Ranganathan, and Christos Kozyrakis. Joulesort. *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, 2007.

[18] Daniel Schall, Volker Hoefner, and Manuel Kern. Towards an enhanced benchmark advocating energy-efficient systems. *Topics in Performance Evaluation, Measurement and Characterization*, page 31–45, 2012.

[19] Daniel Schall and Theo Härder. Wattdb - a journey towards energy efficiency. *Datenbank-Spektrum*, 14(3):183–198, 2014.

[20] TPC. History of tpc. `https://www.tpc.org/information/about/history5.asp`, 1998. Accessed on: 23-10-2023.

[21] TPC. TPC-C Standard Specification, 2010; v5.11.

[22] TPC. TPC-Energy Specification, 2012; v1.5.0.

[23] Dimitris Tsirogiannis, Stavros Harizopoulos, and Mehul A. Shah. Analyzing the energy efficiency of a database server. *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, 2010.