Preliminary Study on Effects of Object-Relational Mapping on the Efficiency of Monolithic and Distributed Relational Database Systems

Starić, Elvis

SVEUČILIŠTE JURJA DOBRILE U PULI
FAKULTET INFORMATIKE U PULI

ZAVRŠNI RAD

PRELIMINARY STUDY ON EFFECTS OF OBJECT-RELATIONAL MAPPING ON THE EFFICIENCY OF MONOLITHIC AND DISTRIBUTED RELATIONAL DATABASE SYSTEMS

Elvis Starić

Pula, Svibanj 2024.

SVEUČILIŠTE JURJA DOBRILE U PULI
FAKULTET INFORMATIKE U PULI

ZAVRŠNI RAD

PRELIMINARY STUDY ON EFFECTS OF OBJECT-RELATIONAL MAPPING ON THE EFFICIENCY OF MONOLITHIC AND DISTRIBUTED RELATIONAL DATABASE SYSTEMS

Elvis Starić

JMBAG: 0246074863
Kolegij: Programsko inženjerstvo
Znanstveno područje: Društvene znanosti
Znanstveno polje: Informacijske i komunikacijske znanosti
Znanstvena grana: Informacijski sustavi i informatologija

Mentor: doc. dr. sc. Nikola Tanković

Pula, Svibanj 2024.

Sažetak

Kako postoji sve veća potreba za pohranjivanjem i procesiranjem velike količine podataka za ispravan rad mnogih aplikacija, tradicionalne monolitne relacijske baze podataka, zbog svoje arhitekture, postaju nedostatne za takve aplikacije. Kao potencijalno rješenje predstavljaju se distribuirane baze podataka. Distribuirane baze podataka pohranjuju podatke na više instanci, što omogućuje veću skalabilnost, fleksibilnost, dostupnost baze podataka te potencijalno bolje performanse u obradi podataka.

Nadalje, objektno-relacijsko preslikavanje (eng. ORM – object-relational mapping) je tehnika programiranja koja olakšava interakciju između baze podataka i programa pisanog u objektno orijentiranom jeziku. Ona omogućuje pisanje upita na bazu podataka u objektno-orijentiranoj paradigmi umjesto korištenja SQL programskog jezika. Iako objektno-relacijsko preslikavanje znatno olakšava pisanje programskog koda, također je poznato da negativno utječe na performanse baza podataka.

Ovaj rad ima za cilj utvrditi utjecaj objektno-relacijskog preslikavanja na monolitne te distribuirane relacijske baze podataka te usporediti ima li veći utjecaj na monolitne ili distribuirane baze podataka. Koristiti ćemo TPC
(Transaction Processing Performance Council) – C benchmark. MySQL će se koristiti kao monolitna baza podataka, TiDB kao distribuirana te SQLAlchemy kao ORM.

Preliminary Study on Effects of Object-Relational Mapping on the Efficiency of Monolithic and Distributed Relational Database Systems

Elvis Starić *, Nikola Tanković *
* Faculty of Informatics, Pula, Croatia
elvis.staric@unipu.hr, nikola.tankovic@unipu.hr

Abstract

As there is a growing need to store and process large amounts of data for the correct operation of many applications, traditional monolithic relational databases, due to their architecture, are becoming insufficient for such applications. Distributed databases are presented as a potential solution. Distributed databases store data on multiple instances, which enables greater scalability, flexibility, database availability, and potentially better performance in data processing.

Furthermore, object-relational mapping (ORM) is a programming technique that facilitates the interaction between a database and a program written in an object-oriented language. It allows writing database queries in an object-oriented paradigm instead of using the SQL programming language. Although ORMs make it much easier to write programming code, they are also known to affect database performance negatively.

This paper aims to determine the impact of ORM on the performance of monolithic and distributed relational databases and to compare whether it has a greater effect on monolithic or distributed databases. We will use the TPC (Transaction Processing Performance Council)-C benchmark. MySQL will be used for the monolithic database, TiDB for the distributed database, and SQLAlchemy for ORM.

Keywords - Relational database, Monolithic database, Distributed database, TPC-C, ORM

I. Introduction

ORM (object-relational mapping) is a powerful and popular technique that facilitates interaction between databases and programs written in an object-oriented paradigm. ORMs were first developed in the mid-1990s to increase developer productivity by allowing developers to work with database data using the same object-oriented structures and mechanisms they would use for any internal data. The main intention is that developers do not need to rely on special techniques or learn a new querying language like SQL to be productive with data [1].

OMR makes development easier for the developer, but it has been known to harm the performance of the database itself. This is because ORMs must translate between the object-based representation of data in code and the relational database structure, which are very different. The main differences are that objects are instantiated at runtime, whereas data in the relational model is constant; objects have no fixed schema, while databases do. This data representation varies depending on the implementation, and database calls from the application are translated through this internal data model before execution [2]. This translation process can be time-consuming, especially if the ORM must go through multiple object layers to fetch the requested data. As a result, in some cases, even simple queries result in unnecessarily complex and suboptimal SQL queries.

Another reason is that ORMs often load entire objects and their related data, which can lead to unnecessary dana retrieval and decreased performance. [3].

This preliminary study aims to determine ORM's impact on the performance of monolithic and distributed relational databases and compare whether it has a greater effect on monolithic or distributed databases.

II. Motivation

With a growing need to store and process large amounts of data for the correct operation of many applications, traditional monolithic relational databases are becoming insufficient. Distributed databases, which store data on multiple instances, enabling better performance in dana processing [4], are presented as a potential solution.

Distributed databases are becoming increasingly popular to manage and process large volumes of dana efficiently [5]. They are being used increasingly in many different applications, such as financial institutions, telecommunications, IoT, and many others. Almost any system that requires high availability, scalability, and reliability from its database systems can benefit from using this type of database when developing new solutions.

Additionally, ORMs are a widely used tool whose main aim is to increase developer productivity and accelerate development time; however, as has been stated, they do come with some drawbacks. Nevertheless, ORMs themselves are not aware of the architecture of the database, and they do not, therefore, consider the intrinsic details (such as partitioning) of the distributed database system when generating the SQL query.

Since distributed databases are becoming more important for modern application development, it is necessary to determine whether ORMs could still be used or should be avoided when it comes to distributed databases.

III. Related work

When it comes to ORMs, it is well known that using them in development, even though they are a great tool for developers in the sense that they increase productivity, comes with a trade-off in database performance. This has again been proven in the paper – "Object Relational Mapping Framework Performance Impact" [6]. This paper researched the impact of different ORMs (each for a different programming language) on relational database performance. The results led to the conclusion that the impact of an ORM on relational query performance is a significant decrease in execution time when using an ORM compared to not using an ORM.

The paper [7] from 2017 also discussed the impact of ORMs on database performance. The results have shown some of the consequences that ORMs may have on query execution performance, but they have also found some potential solutions, such as parametrization, multi-schema models, and using unstructured databases as an alternative to relational ones.

In another study [8] on this topic from 2020 one of the goals was to investigate the impact of the ORM on the relational database. The results showed that ORM has negative performance implications as query complexity increases, in terms of measurable performance impacts to the calling application and system resources.

The contribution of this preliminary study to the topic is to investigate whether the already proven impacts of ORMs on relational databases are any different when it comes to usage in monolithic and distributed databases.

IV. Experiment setup

Experiments were conducted on a single machine with an AMD Ryzen 7 5800H processor and 16 GB of RAM running a Windows 11 operating system with Windows Subsystem for a Linux environment.

The benchmark used in this experiment is the TPC (Transaction Processing Performance Council)-C benchmark. MySQL is used for the monolithic database, TiDB Cloud version for the distributed database, and SQLAlchemy for ORM.

TPC-C benchmark simulates a warehouse network. It includes a combination of five transactions (New-Order, Payment, Order-Status, Delivery, Stock-Status). The test aims to evaluate the transaction processing speed when multiple simultaneous requests are sent to the database.

The first step was to create and populate both MySQL and TiDB databases with a dataset defined by the TPC-C specification [9]. The database consists of nine relations: Warehouse, District, Orders, OrderLine, NewOrders, Stock, Item, Customer, and History (the number of warehouses in our dataset is 20).

After the databases had been set up, the next step was to run the tests. To test the databases' performance without ORM, we used Percona-Lab’s tpcc-mysql benchmark [10].

As for the test with ORM, since there were no tools readily available, we created our own based on the tool used to test the performance of different ORMs [11], adapted to our databases and tests we wanted to run [12].

The tests we ran were the performance of the databases with all five transactions (standard TPC-C test), the performance of the database with each of the five transactions individually, and latency. All tests were run for ten minutes.

V. Results

The first test was run with all five transactions. The results are shown in figures 1 and 2.

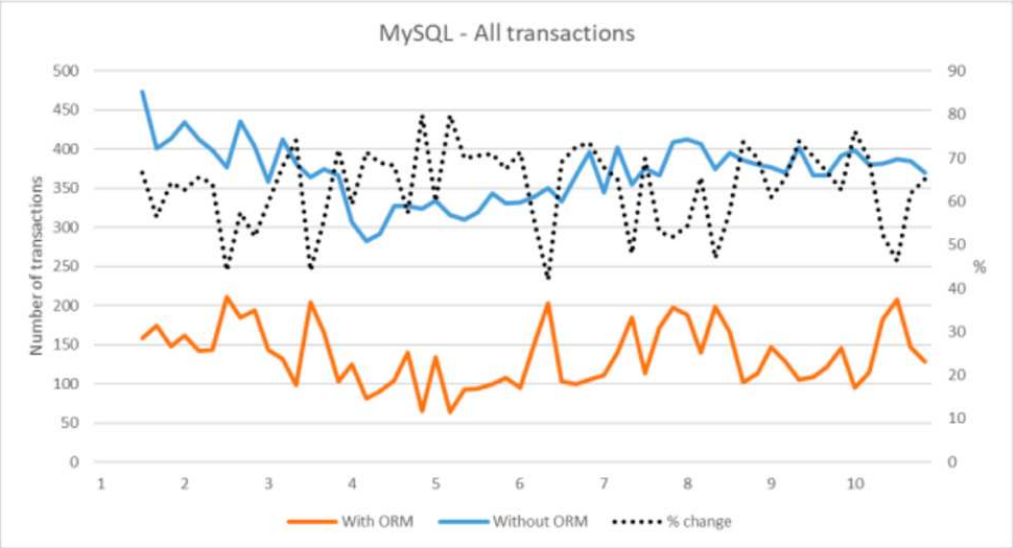

Figure 1 - MySQL – Transactions per minute with all transactions

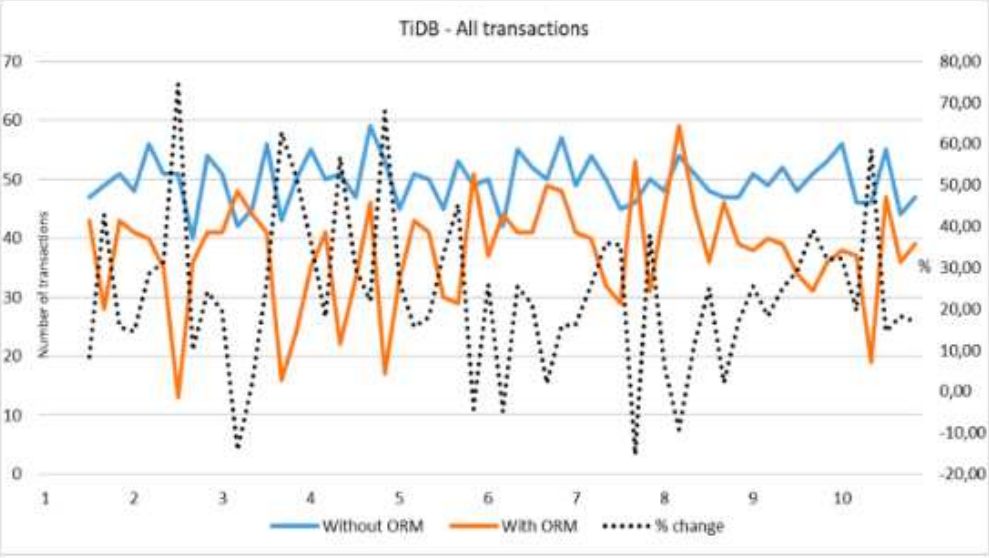

Figure 2 - TiDB – Transactions per minute with all transactions

The test results proved what had already been stated, which is, that using the ORM impacts the performance of the database negatively. However, the impact on the performance seems very different between MySQL and TiDB. With MySQL, the average % difference in the number of transactions executed is 63,33%, whereas with TiDB, it is only 24,03%. Therefore, we can conclude that the usage of ORM has a lesser impact on distributed database performance than monolithic database performance. To further try to understand why this is, we decided to run the test for each of the transactions separately to try to establish whether any of the transactions had a bigger impact on the performance, that is, if transaction complexity is slowing down the performance, or are the transactions slower regardless of the complexity.

Figure 3 - MySQL – Transactions per minute with each transaction running separately

Figure 4 - TiDB - Transactions per minute with each transaction running separately

TABLE I. - % DIFFERENCE IN NUMBER OF TRANSACTIONS

	MySQL		TiDB	
	Average % difference	***p-value***	***Average % difference***	***p-value***
All transactions	63,33% *	p < 0.001	24,03% *	p < 0.001
NewOrder	52,94% *	p < 0.001	30,88% *	p < 0.001
Payment	52,77% *	p < 0.001	26,53% *	p < 0.001
OrderStatus	55,79% *	p < 0.001	55,79% *	p < 0.001
Delivery	57,29% *	p < 0.001	24,56% *	p < 0.001
StockLevel	56,30% *	p < 0.001	17,25% *	p < 0.02

*statistically significant

Table I shows the average % difference in the number of transactions without and using ORM when all transactions were running and with each transaction running separately with the associated p-value. We can see from these results that transaction complexity is not the reason for the difference in the performance of the database when ORM is used since the % difference is lower in almost all cases (it is the same for the OrderStatus transaction) for the TiDB database.

The last test we ran was the latency test to see the maximum response time for each ten-second interval.

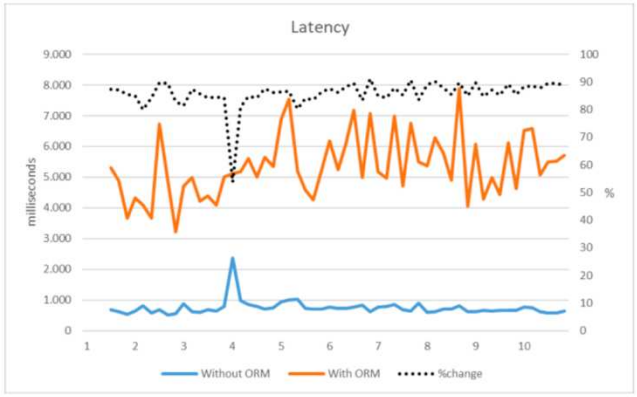

Figure 5 - MySQL latency

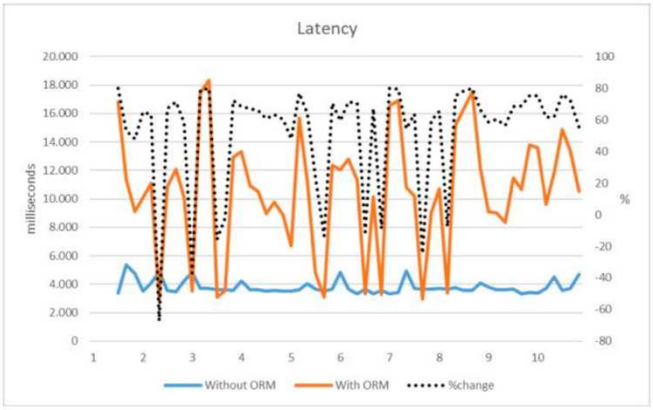

Figure 6 - TiDB latency

TABLE II. - LATENCY % DIFFERENCE

	MySQL		TiDB	
	Average % difference	***p-value***	***Average % difference***	***p-value***
Latency	85,75%	p < 0.01	52,26%	p < 0.01

Latency results once again proved that the execution of transactions is faster while using a distributed TiDB database.

VI. Conclusion

Distributed databases are becoming more prevalent in today's modern application development. Their ability to store and process large amounts of data is one of the main reasons for the popularity increase. Additionally, ORMs are a widely used tool to facilitate application development for developers. However, they are known to impact the performance of the database negatively. Taking both factors into consideration, this paper aimed to establish whether the use of ORMs had a bigger impact on monolithic databases or distributed databases to test if the usage of ORM is still feasible in distributed databases.

To test the performance of the databases, we used the TPC-C benchmark and tested the execution of all transactions, each transaction individually, and latency. The first test showed that there is a difference when using the ORM with monolithic and distributed databases; that is, the execution time was faster in a distributed database. To try to understand better why this is, we tested the execution of each transaction separately, to test if the transaction complexity affects the execution times differently, and latency, to compare maximum response times for each ten-second interval. The results of the tests further supported the results of the first test.

To summarize, the results of our preliminary study supported the already known findings, which is that ORMs do have some negative impacts on the performance of the databases since they sometimes produce complex and suboptimal SQL queries. However, the results also showed that they have a lesser impact on the performance of the distributed databases. Taking that into account, along with the fact that technology today is rapidly evolving and that extensive research is being put into this field, which could lead to potential new solutions to mitigate the negative impacts of ORMs further, ORMs can still be considered for use when creating systems with distributed databases.

VII. Limitations

This paper has some limitations that should be noted. The primary limitation is that this is a preliminary study, for which only one ORM and one monolithic and distributed database were used to experiment. Furthermore, all the tests were run on a single machine, and it is possible that the results would be somewhat different on a different machine. Because of this collected data is generalized and should be interpreted cautiously. Despite these limitations, this preliminary study provides useful insight into the discussed topic and could be used as a reference for further research.

References

[1] J.M.Barnes, 2007 *Object-Relational Mapping as a Persistence Mechanism for Object Oriented Applications*
[2] M.Subramanian, V.Krishnamurthy, 1999, *Performance Challenges in Object-Relational DBMSs*
[3] C.Ireland, D.Bowers, M.Newton, K.Waugh, 2009 *Understanding Object-Relational Mapping: A Framework Based Approach*
[4] P. Valduriez, 2011 *Principles of Distributed Data Management in 2020?*
[5] M.T.Ozsu, 1996 *Distributed and Parallel Database Systems*
[6] V.Sivakumar, T.Balachander, Logu and R.Jannali, 2021 *Object Relational Mapping Framework Performance Impact*
[7] D.Colley, C.Stanier, Md Asaduzzaman, 2017, *The Impact of Object-Relational Mapping Frameworks on Relational Query Performance*
[8] D.Colley, C.Stanier, Md Asaduzzaman, 2020, *Investigating the Effects of Object-Relational Impedance Mismatch on the Efficiency of Object-Relational Mapping Frameworks*
[9] TPC-C Specification, https://www.tpc.org/TPC_Documents_Current_Versions/pdf/tpc-c_v5.11.0.pdf
[10] Percona's TPC-C Benchmarking tool for MySQL, https://github.com/Percona-Lab/tpcc-mysql
[11] ORM performance benchmark, https://github.com/DominovTut/Python_ORM_Benchmark
[12] Our test, https://github.com/elvisstaric/impact_of_orm