JOURNAL
COMPUTER SCIENCES INSTITUTE

JCSI 24 (2022) 250–257
Received: 14 June 2022
Accepted: 6 July 2022

Performance analysis of relational databases MySQL, PostgreSQL and Oracle using Doctrine libraries

Analiza wydajności relacyjnych baz danych MySQL, PostgreSQL oraz Oracle z zastosowaniem bibliotek Doctrine

Marcin Choina* , M. Skublewska-Paszkowska

Department of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract
In modern applications, databases perform a very important function but the choice of a database system and additional libraries may affect the speed of the operations. The paper presents a time analysis concerning the performing of insert, update, delete and select operations on three database systems, MySQL 8.0, PostgreSQL 14.1 and Oracle 21c, cooperating with an application using Doctrine libraries. The obtained results showed differences between performing operations with and without object-relational mapping. In cooperation with the application, the operations were carried out the fastest using the PostgreSQL system. The Oracle system performed data selection faster without mapping on a large data set.

Keywords: relational databases; Doctrine; ORM; PHP

Streszczenie
We współczesnych aplikacjach, bazy danych pełnią bardzo ważną rolę, jednak wybór systemu bazodanowego i dodatkowych bibliotek może wpływać na szybkość wykonywania operacji. W niniejszej pracy przedstawiono czasową analizę dotyczącą wykonywania operacji bazodanowych insert, update, delete i select dla trzech systemów baz danych MySQL 8.0, PostgreSQL 14.1 i Oracle 21c, współpracujących z aplikacją wykorzystującą biblioteki Doctrine. Badania wykazały różnice między wykonywaniem operacji wraz z mapowaniem obiektowo-relacyjnym, a wykonywaniem samych zapytań. Przy współpracy z aplikacją, operacje najszybciej przeprowadzono korzystając z systemu PostgreSQL. System Oracle szybciej wykonywał operacje pobierania danych bez udziału mapowania na dużym zbiorze danych.

Słowa kluczowe: relacyjne bazy danych; Doctrine; ORM; PHP

*Corresponding author
Email address: marcinchoina1997@gmail.com (M. Choina)

1. Introduction

Database systems are a common method of data storage today. They are used in various types of IT systems, computer programs and applications. Currently, databases can be divided into several types (e.g. noSQL, object-oriented, hierarchical), but the most common choice for application development are relational databases [1]. There are many free and commercial relational database management systems available on the market, differing in capabilities and speed of operations. The performance is also influenced by the technology in which the application using the database is created and the operating system on which it is run.

This paper aims to compare the time performance of MySQL 8.0, PostgreSQL 14.1 and Oracle 21c database systems running with an application using Doctrine libraries, one of the most popular and fast technologies using object-relational mapping, allowing for the connection of an application written in PHP with the database. The performed tests allow to verify the time of operations and assess which database systems are the fastest and the slowest in carrying out operations of inserting, updating, deleting or selecting data, using, among others, the mechanisms of sorting, grouping and joining tables.

1.1. Doctrine

Doctrine is an open source library set for PHP technology under the MIT (the Massachusetts Institute of Technology) license. It is the default communication mechanism with the database for the Symfony framework and can be much more efficient compared to other mechanisms, e.g. Propel library [2]. The Doctrine abstraction layer (DBAL) enables cooperation with many available database systems. Additionally Doctrine is based on object-relational mapping (ORM) technology that allows to present information retrieved from the database using entity class objects. This solution allows to significantly improve the quality of the code and enables faster information management in the database [3]. Doctrine supports relationship mechanisms, which is achieved through associations between class objects. They make it possible to refer to other objects, and the library ensures that the relevant data is retrieved from the database [4].

1.2. MySQL

MySQL is a database system licensed under the GNU GPL license, currently developed by Oracle [5]. It is supported by a large number of technologies as well as

by the most popular programming languages such as PHP, Java, C ++, and it can be used on all popular operating systems [6]. It is also part of the server environment on the Linux - LAMP platform.

1.3. PostgreSQL

PostgreSQL is an advanced database system released under the PostgreSQL license, which is very similar to the MIT license. The system introduces many proprietary extensions to the implementation of the SQL standard. It enables the creation of stored procedures in various programming languages such as Python or Perl. PostgreSQL is a combination of a relational and object-oriented database system, which allows for a more precise adaptation of the database to the needs of the application being created [7].

1.4. Oracle

The Oracle database management system is a commercial solution released by Oracle Corporation under a paid license [8]. There is also a free version, the Express Edition, with limited functionality. Like PostgreSQL, it contains elements of an object-oriented database system. It also provides its own PL / SQL language, which enables the extension of standard functionalities to automate some activities carried out in the database.

2. Related works

Many scientists analyzed the differences between individual database solutions, their capabilities and performance. The authors of the article [9] chose to compare various relational database systems, Oracle Database 19c, SQL Server 2019, PostgreSQL 12 and MySQL 8. For this purpose, on each of them they created a database according to the same scheme, and then tested on it data selecting, grouping, and inserting operations as well as backup and restoring data. These tests showed differences in the performance of databases, where the most efficient in terms of time turned out to be the Oracle and SQL Server systems, and the MySQL system was the worst.

Oracle and SQL Server databases are commercial solutions which were compared by the authors of articles [10], [11]. The first one presents an analysis of the differences and performance of the servers in the given databases. The research presents the advantages of the Oracle database, which has better accessibility, in terms of the possibility of installation on various operating systems and support by a larger number of programming languages, as well as having multi-layer security. On the other hand, MS SQL is supported by simpler language syntax and better query performance for single and joined tables. In the second of the mentioned articles, the performance of these systems was compared with the use of a desktop application. In the study, an external application was used to perform various actions on the database, and the execution time of a given actions was taken from the views of the database system such as V$SQL for the Oracle database and sys.dm_exec_query_stats for MS SQL. The operations were performed on a set of 500-100,000 records for text or numeric data and on a set of 1-50 records for binary data larger than 50MB, and the cache was cleared after each action. The results showed that the MS SQL database performs DML (Data Manipulation Language) operations such as inserting and updating data better. The Oracle database is much better suited for DQL (Data Query Language) operations, which means data selecting.

Authors of many articles compare Oracle, SQL Server and MySQL databases. Examples of this comparison are the articles [12], [13]. In the first one, the time of updating one column of a table, transferring records to another table, and selecting records using sorting, grouping and joining tables in a database system running on a home computer was examined. The MySQL was the best for most operations in this case. In the second article, the authors compared these relational databases with non-relational databases such as Mongo, Redis, GraphQL and Cassandra, where the times of basic operations were compared. The obtained results confirmed that queries to non-relational databases were executed much faster than in the case of relational databases. Considering relational systems, in the Oracle system there has been achieved one of the fastest data selection times.

Currently, various frameworks or libraries are used to support the application development process, as well as connecting to the database and managing the data stored. Since the libraries are designed so that they can cooperate with various database systems, they can have an impact on the efficiency of performed operations. One of the most popular frameworks for creating web applications in PHP language are Symfony and Laravel. For Symfony, the preferred way to communicate with the database is to use Doctrine libraries, while Laravel has its own mechanism Eloquent ORM [14]. In the article [15], the authors examine the performance of relational databases cooperating with an application written using the Laravel framework. The test consisted in comparing the average execution times of operations such as insert, update, delete and select on a database running on mid-range hardware. To the research open source databases such as MySQL 8 and PostgreSQL 12 and the commercial Microsoft SQL Server 2017 were used. The results showed that in the case of a small number of records stored in the database (up to 1000 records), MySQL was the best solution, while with a larger number of records, the PostgreSQL system turned out to be better.

Another article presents the analysis of MySQL, MS SQL and PostgreSQL database systems in the context of web applications using the Spring framework, the Hibernate library and the JDBC interface [16]. In the case of database cooperation with a web application, PostgreSQL turned out to be the most efficient system for complicated operations on joined tables, using both the Hibernate library and the JDBC interface. In the case of a single table, the average times of performing opera-

tions were similar, and when performing operations directly on the database, MS SQL was the most optimal system.

In all above-mentioned articles, authors focused mostly on comparing the average times of performing basic operations on the database. In the article [17] the authors also analyzed the CPU load and the memory usage, and the authors of the article [9] examined the times of making and restoring a database backup. In most studies one model to build the database was used for each tested system. Some researchers used an additional application [11] or technology [15, 16], but few studies were related to the cooperation of databases with Doctrine libraries.

Based on the literature analysis, the following hypotheses were made:

- Insert, update and delete operations should be performed the fastest by the PostgreSQL database in cooperation with the Doctrine library,
- The Oracle database should perform the operations of selecting data the fastest with a large number of records (100,000) in the table.

3. Research method

In this paper, it was decided to compare the average times of performing select, insert, update and delete operations. These times were measured for the query with object-relational mapping on the data and without it. For the needs of the research, an application that uses Doctrine libraries has been created. It contained a module that allows to test the database. On each tested system, a database based on the model shown in Figure 1 was created.

The research was carried out according to a scenario. The purpose of the study was to verify the time of performing subsequent operations specified in Table 1, depending on the used database system and the number of records stored in the database tables. The initial condition for the study was a fixed number of records in each table of the tested database, which was 1,000, 10,000 or 100,000 records, respectively.

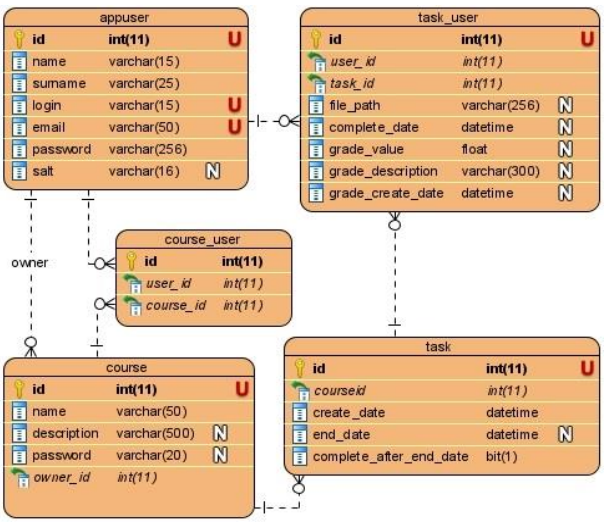

Figure 1: Database model.

The following activities were performed during the study:

- selecting one of the operations (Table 1),
- test of operations on MySQL database,
- test of operations on PostgreSQL database,
- test of operations on Oracle database,
- restoring the initial state of databases.

Before each test, the cache of the database system and application was cleared, and the entire test was repeated 10 times for each operation. Computer with the following parameters was used for the tests:

- Intel® Core™ i5-10300H processor,
- 8GB DDR4 2400 MHz RAM,
- Xioxia BG4 512GB NVMe SSD,
- Windows 10 64-bit operating system.

Table 1: Database operations

Operation	Query
Insert one row into the table	INSERT INTO appuser values (100001, “Adam”, ”Nowak”, ”anowak”, ”anowak@pollub.pl”, ”zaq1@WSX”)
Update one row in a table	UPDATE appuser SET name = “Marek” WHERE id = 567
Delete one row from the table	DELETE FROM task_user WHERE id = 567
Select all rows from one table	SELECT * FROM appuser
Select all rows sorted	SELECT * FROM appuser ORDER BY surname, name
Select rows using pattern search	SELECT * FROM appuser WHERE UPPER(name) LIKE ‘%ADA%’
Select one row from a table based on the primary key	SELECT * FROM appuser WHERE id = 567
Select rows with joining tables using JOIN construction	SELECT * FROM course c INNER JOIN appuser u ON u.id = c.owner_id INNER JOIN task t ON t.course_id = c.id
Select rows with joining tables using WHERE construction	SELECT * FROM course c, appuser u, task t WHERE u.id = c.owner_id AND t.course_id = c.id
Select rows using grouping functions	SELECT c.* FROM course c LEFT JOIN task t ON t.course_id = c.id GROUP BY c.* HAVING count(t.id) = 0
Select rows using a correlated query	SELECT t.* FROM task_user t WHERE grade_value = (SELECT MAX(grade_value) FROM task_user tu1 WHERE t1.user_id = t.user_id)

4. Results

The first tested operations were DML queries. First, insert of single user operation were tested (Figure 2-3). The results do not vary significantly depending on the number of records in the tables for each database system. This type of operations for both the query with mapping and without it were executed the quickest in the PostgreSQL database. With the other systems, insert operations were performed much slower. By performing the query without mapping to the database, using the MySQL database the shorter execution time was achieved than for Oracle database, but for this system the application processed the entity object into a query much longer. The application did not take a long time (18-21ms) to process the object to insert it into the Oracle database, which shows that this system works quicker with the application than MySQL.

The results of the update and data delete operation (Figure 4-7) were similar to the data insert operation. The fastest execution times were achieved using PostgreSQL database.The results of the other two systems differ depending on whether the object-relational mapping was considered. The shorter time was gained performing

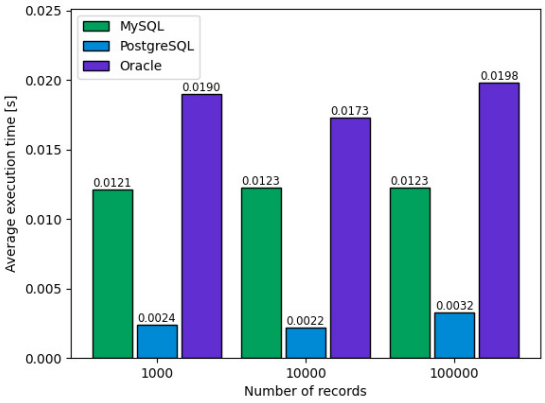

Figure 2: Average time of INSERT operation.

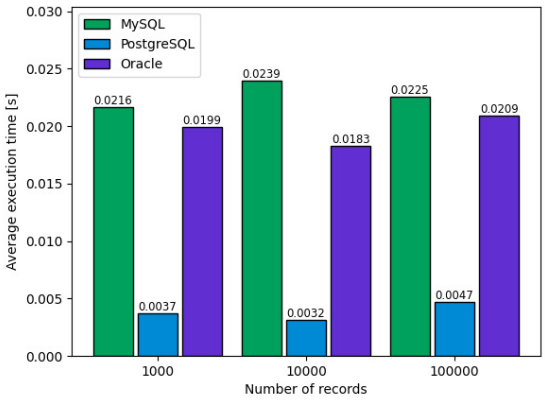

Figure 3: Average time of INSERT operation with ORM.

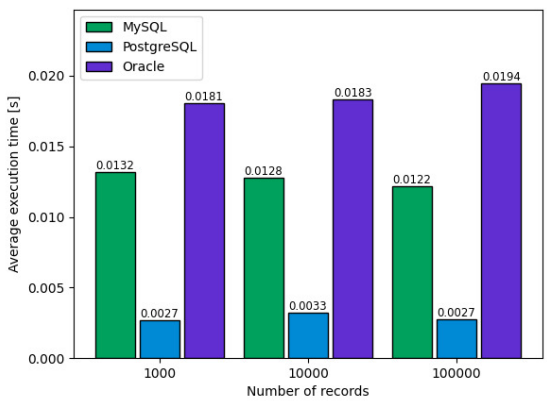

Figure 4: Average time of UPDATE operation.

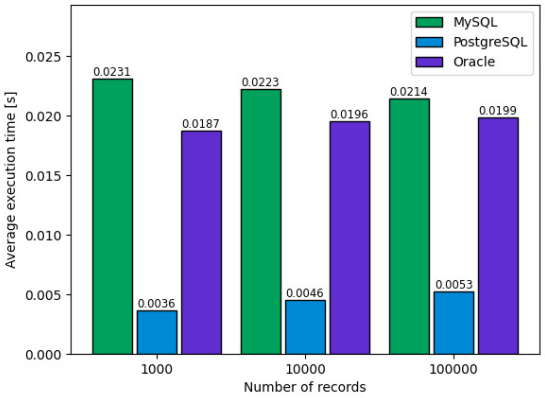

Figure 5: Average time of UPDATE operation with ORM.

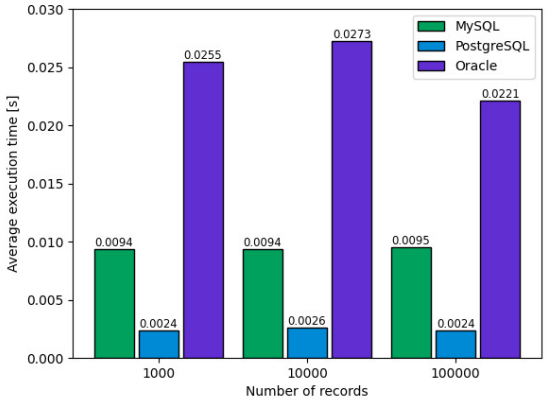

Figure 6: Average time of DELETE operation.

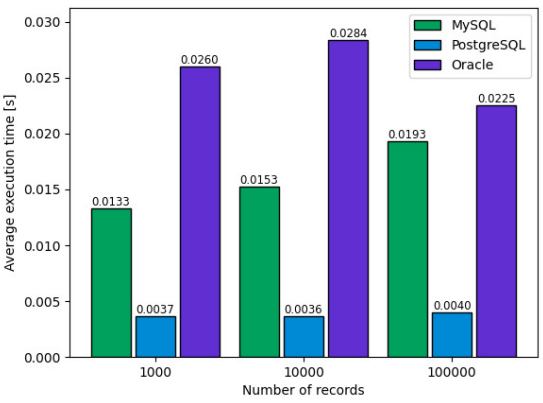

Figure 7: Average time of DELETE operation with ORM.

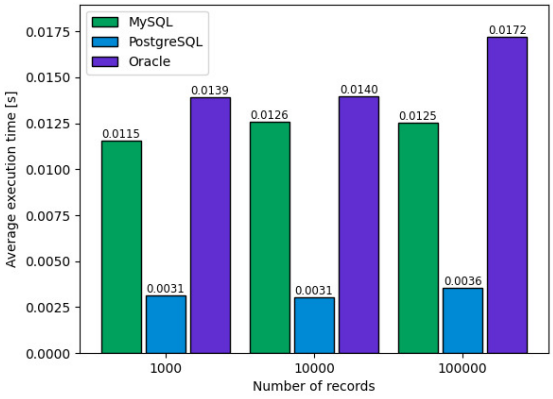

Figure 8: Average time of select one row operation.

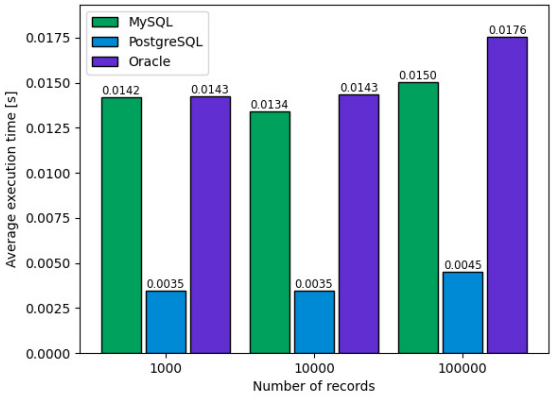

Figure 9: Average time of select one row operation with ORM.

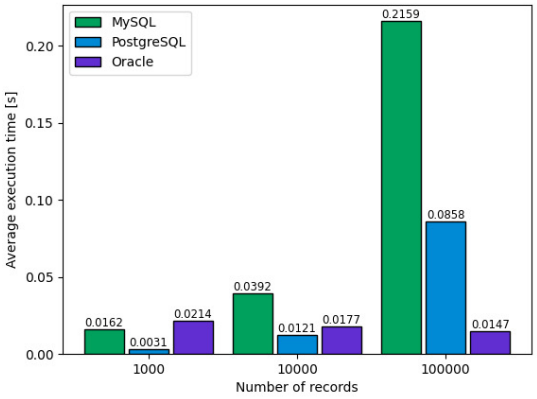

Figure 10: Average time of select all data operation.

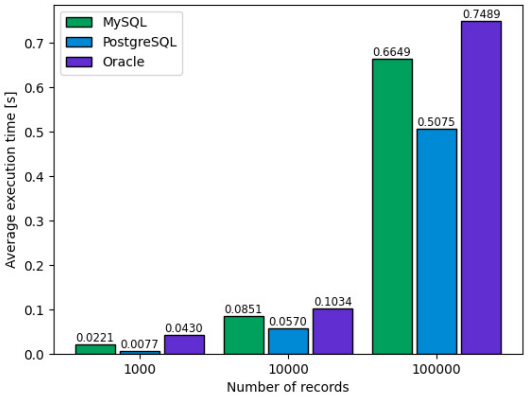

Figure 11: Average time of select all data operation with ORM.

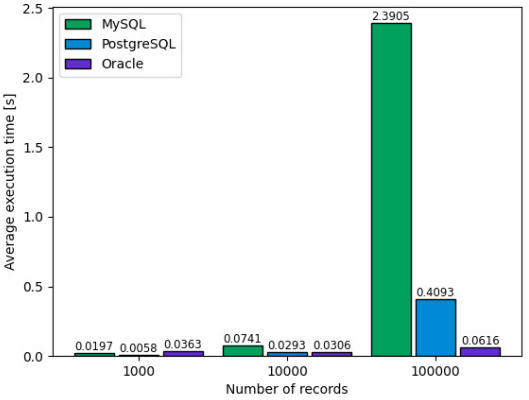

Figure 12: Average time of select sorted data.

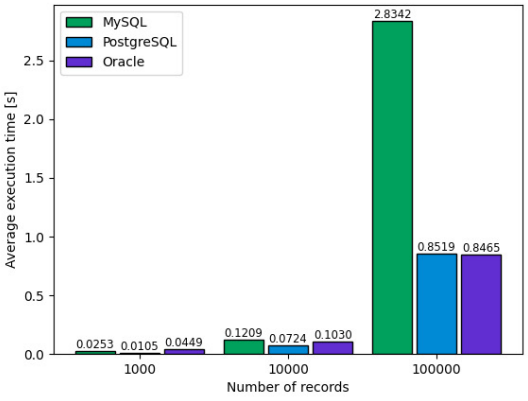

Figure 13: Average time of select sorted data with ORM.

update operation in cooperation with the application using the Oracle database than MySQL. Using the MySQL system theshorter time was obtained performing the update query without cooperating with the application. Aditionally the MySQL system performed the delete operation with and without object-relational mapping faster than the Oracle system.

Next operations belong to the group of DQL instructions that allow to retrieve specific data from the database. One of the most commonly used operations is selecting one row from a table based on its primary key. The results of this operation were similar to the results of DML operation studies (Figures 8-9). In the PostgreSQL database the data from the database has been obtained the fastest, but regarding to other databases, the performing query wihout mapping was faster with MySQL database than whit Oracle. While cooperating with the application, the longest times were obtained for a large number of records (i.e. 100,000) using the Oracle system. With less number of records in the tables, its results were similar to the MySQL system.

When selecting all the rows from the database (Figures 10-11), with PostgreSQL the shortest average time was obtained when the number of records in the

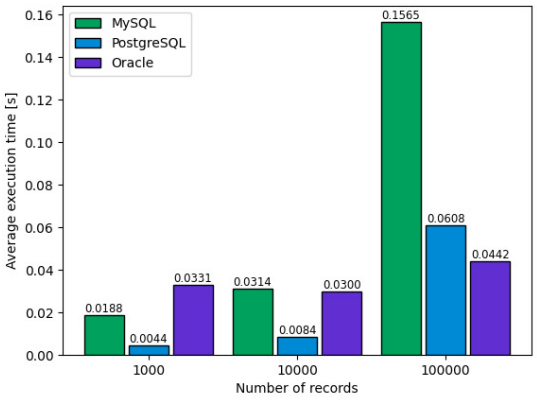

Figure 14: Average time of select operation using pattern search.

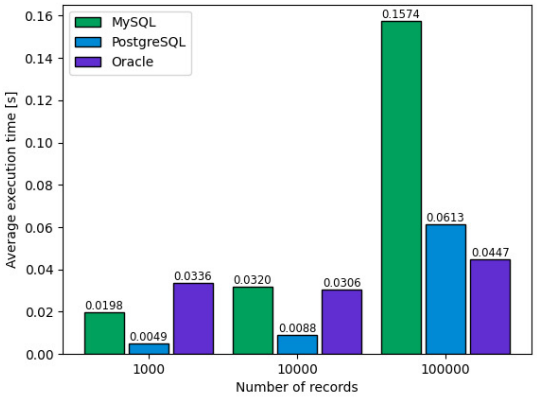

Figure 15: Average time of select operation using pattern search with ORM.

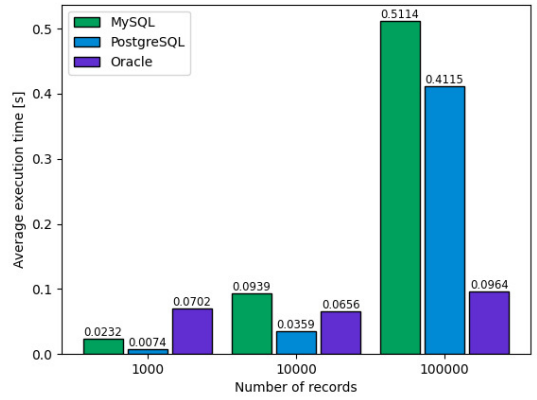

Figure 16: Average time of join operation using JOIN construction.

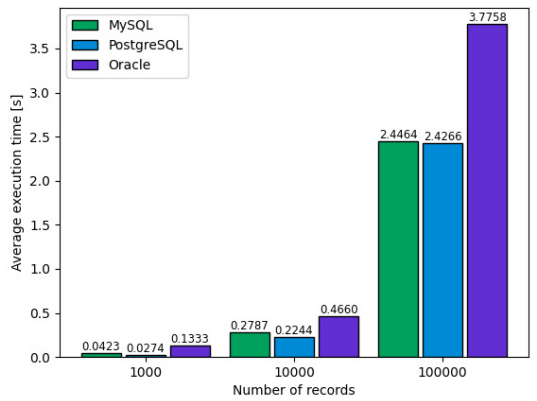

Figure 17: Average time of join operation using JOIN construction with ORM.

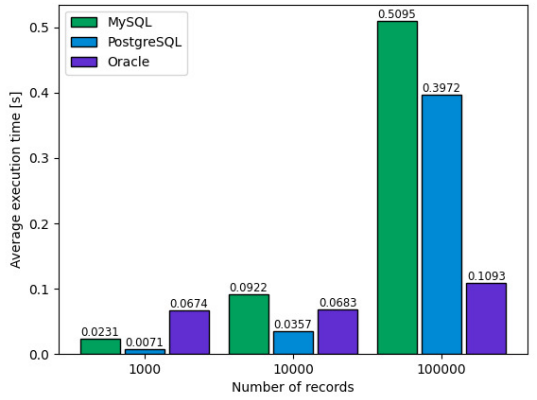

Figure 18: Average time of join operation using WHERE construction.

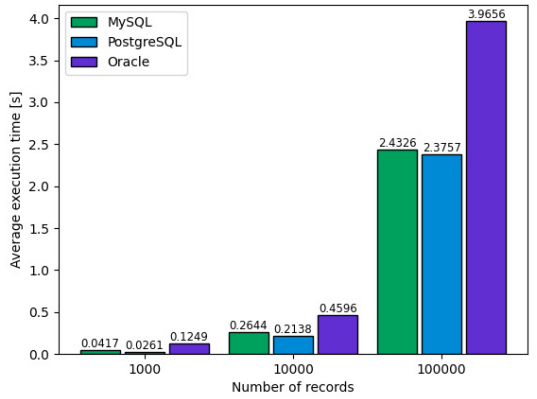

Figure 19: Average time of join operation using WHERE construction with ORM.

tables was low or medium (up to 10,000). With a larger number of records, the operations were performed much faster using the Oracle system, for which the average operation execution time decreased while the number of records increased. However, when working with the application, using the Oracle database the longest times were achieved and the best times were obtained using the PostgreSQL database, regardless of the number of records in the tables.

In the Oracle the best times were obtained when executing select queries using sort or pattern search (Figures 12-15) on a table with a large number of rows (i.e. 100,000). With fewer numbers, PostgreSQL performed queries the quickiest. The MySQL system with a large number of records in the tables, carried out both operations several times longer than the other systems. Regarding mapping with a large number of records (i.e. 10,000), there have been obtained a very similar result for the Oracle and the PostgreSQL system when sorting. When searching for pattern, the time was up to 27% shorter using Oracle system.

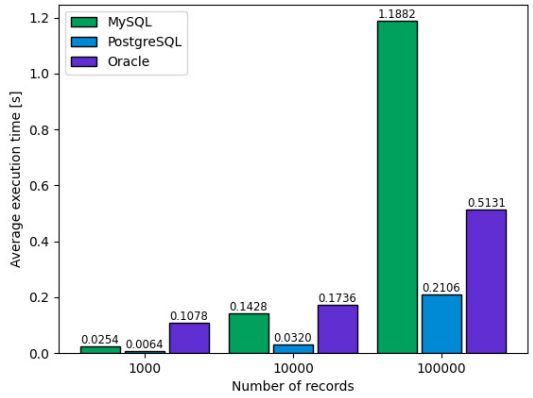

Figure 20: Average time of group operation.

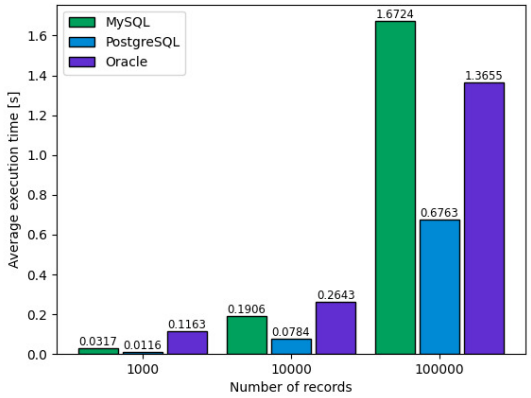

Figure 21: Average time of group operation with ORM.

When operations of joining the tables were performed, the results for both tested methods were similar (Figures 16-19). The Oracle system was very quick at executing a query for a large number of records in the database, while for a smaller number of records, the shortest time was achieved using the PostgreSQL system. When cooperating with the application, in the PostgreSQL database the best time was achieved, regardless of the number of records, but also using the MySQL system a very similar average execution time was acheved.

For the grouping operations, the best times were obtained using PostgreSQL database with and without mapping (Figures 20-21). The Oracle system performed operations for small and medium number of records (up to 10,000) the slowest, while in MySQL system the slowest execution time was obtained for a large number of records stored in tables.

For the select operation using correlated query (Figures 22-23), while executing the query without object-relational mapping on a large number of records the shortest time was obtained using Oracle database. With a smaller number of records in the table, or when the object-relational mapping was performed, the quickest times were gained using PostgreSQL database.

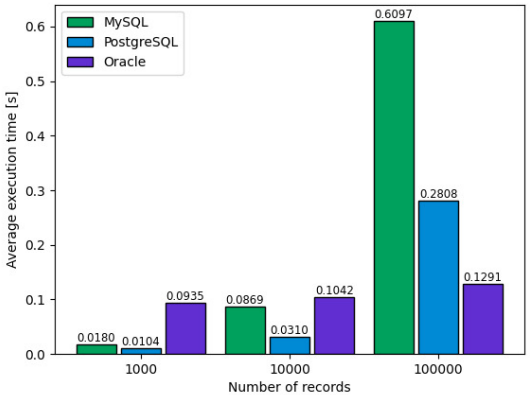

Figure 22: Average time of select operation using correlated query.

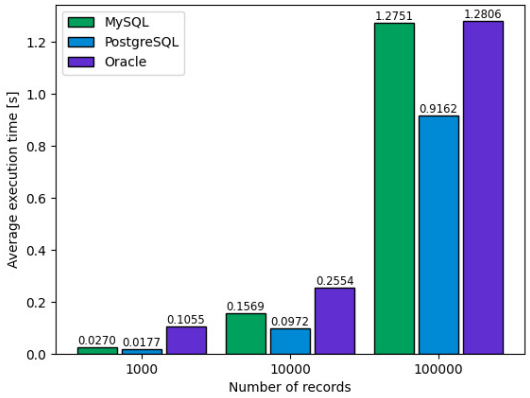

Figure 23: Average time of select operation using correlated query with ORM.

5. Discussion

In the study, in addition to various database systems, the number of records in the tables of a given database, and whether object-relational mapping was used, was taken into account. For different database systems, when executing the query without mapping, for data manipulation operations and for the rest of operations where the number of records in database tables was less than 100,000, PostgreSQL perform operations the fastest. With the Oracle system the shortest times was obtained for almost all data select operation when the number of records in the tables was 100,000, except for retrieving a record using a primary key and using grouping. The obtained result confirmed the results of the research in articles [9] and [13], in which the Oracle system performed select operations the fastest, when the operation was performed directly on the database.

Results vary depending on whether cooperation with application was used, that means executing query with object-relational mapping. Regarding the select operation with a like clause, on a table containing 100,000 records, using Oracle database system the significantly shorter time was obtained comparing to other analyzed database systems. Analising query execution with obiect-relational mapping, operations were performed even several times faster using the

PostgreSQL system. That confirmed the results obtained in the article [16], in which, the PostgreSQL system performed the operations the fastest using additional technology.

The operation execution time increases when object-relational mapping is performed. Regarding the DML operations executed by the MySQL database, the performed queries with mapping lasted even 10ms longer than the queries without it. The difference for time query execution with and without object-relational mapping for other analyzed database systems varied from 1 to 2 ms. For the select operations, the mapping time depended on the number of records stored in the database. The biggest difference between performing operations with and without mapping occurred for the Oracle database for a large number of records in the tables (i.e. 100,000). This database system executed the query without mapping the fastest, but with the mapping the execution times were the longest. When all rows were selected, the system performed operation up to 50 times longer with mapping than without it.

Based on the gathered results, it can be concluded that the first hypothesis, which is that insert, update and delete operations should be performed the fastest by the PostgreSQL database in cooperation with the Doctrine library, was confirmed. PostgreSQL always performed operations of inserting, updating and deleting data the fastest. The second hypothesis, that the Oracle database should perform the operations of selecting data the fastest with a large number of records (i.e. 100,000) in the tables, could be partially confirmed. Oracle database executed most of the queries for selecting data from tables with a number of 100,000 records the fastest. However, in cooperation with the application, the operation time was significantly increased in favor of the PostgreSQL system.

6. Conclusions

In the presented paper, the times of various database systems were tested and compared while working with the Doctrine library, which allowed to verify the hypotheses. The study showed that the PostgreSQL system is the fastest solution when working with an application using Doctrine libraries. The Oracle system works the best when performing select queries without mapping to the database. The research focused on the time of performed operations, which is the most noticeable feature of databases for the end user. Other aspects such as CPU load and memory utilization were not analyzed, which may be an interesting direction of future research.

References

[1] T. Connolly, C. Begg, Database Systems, A practical approach to Design, Implementation, and Management, sixth edition, Pearson, 2015.

[2] K. Sawłuk, M. Miłosz, Comparison of object-relational data mapping technology in Symfony 3 framework, Journal of Computer Sciences Institute 8 (2018) 235-240, https://doi.org/10.35784/jcsi.687.

[3] M. Lorenz, G. Hesse, J. Rudolph, Object-relational Mapping Revised - A Guideline Review and Consolidation, Proceedings of the 11th International Joint Conference on Software Technologies - ICSOFT-EA, (2016) 157-168, https://doi.org/10.5220/0005974201570168.

[4] Doctrine documentation, https://www.doctrine-project.org/index.html, [03.11.2021].

[5] MySQL documentation, https://dev.mysql.com/doc/refman/8.0/en/introduction.html, [26.05.2022].

[6] Supported Platforms: MySQL Database, https://www.mysql.com/support/supportedplatforms/database.html, [24.01.2022].

[7] PostgreSQL website, https://www.postgresql.org/about/, [26.05.2022].

[8] Oracle documentation, https://docs.oracle.com/en/database/oracle/oracle-database/21/cncpt/introduction-to-oracle-database.html, [26.05.2022].

[9] A. Solarz, T. Szymczyk, Oracle 19c, SQL Server 2019, Postgresql 12 and MySQL 8 database systems comparison, Journal of Computer Sciences Institute 17 (2020) 373-378, https://doi.org/10.35784/jcsi.2281.

[10] M. Ilić, L. Kopanja, D. Zlatković, M. Trajković, D. Ćurguz, Microsoft SQL Server and Oracle: Comparative performance analysis, The 7th International conference Knowledge management and informatics (2021) 33-40.

[11] G. Dziewit, J. Korczyński, M. Skublewska-Pawszkowska, Performance analysis of relational databases Oracle and MS SQL based on desktop application, Journal of Computer Sciences Institute 8 (2018) 263-269, https://doi.org/10.35784/jcsi.693.

[12] K. Islam, K. Ahsan, S. Bari, M. Saeed, S. Ali, Huge and Real-Time Database Systems: A Comparative Study and Review for SQL Server 2016, Oracle 12c & MySQL 5.7 for Personal Computer, Journal of Basic & Applied Sciences 13 (2017) 481-490, https://doi.org/10.6000/1927-5129.2017.13.79.

[13] R. Čerešňák, M. Kvet, Comparison of query performance in relational a non-relation databases, Transportation Research Procedia 40 (2019) 170-177, https://doi.org/10.1016/j.trpro.2019.07.027.

[14] Eloquent documentation, https://laravel.com/docs/5.0/eloquent/, [26.05.2022].

[15] R. Wodyk, M. Skublewska-Paszkowska, Performance comparison of relational databases SQL Server, MySQL and PostgreSQL using a web application and the Laravel framework, Journal of Computer Sciences Institute 17 (2020) 358-364, https://doi.org/10.35784/jcsi.2279.

[16] K. Lachewicz, Performance analysis of selected database systems: MySQL, MS SQL, PostgerSQL in the context of web applications, Journal of Computer Sciences Institute 14 (2020) 94-100, https://doi.org/10.35784/jcsi.1583.

[17] Y. Bassil, A Comparative Study on the Performance of the Top DBMS Systems, Journal of Computer Science & Research 1 (1) (2012) 20-31, https://doi.org/10.48550/arXiv.1205.2889.