INTERNET IMPACT TO DBMS PERFORMANCE

Ilja Stanišević, Ph.D. [1]**; Miloš Živković, M.Sc.** [2]**; Ćebić Branko, MA**[3]**, Beljić Dejan, MA**[4]
[1] Akademija strukovnih studija Zapadna Srbija, Valjevo, Serbia, ilja.stanisevic@vipos.edu.rs
[2] Akademija strukovnih studija Zapadna Srbija, Valjevo, Serbia, milos.zivkovic@vipos.edu.rs
[3] Akademija strukovnih studija Zapadna Srbija, Valjevo, Serbia, branko.cebic@vipos.edu.rs
[4] Akademija strukovnih studija Zapadna Srbija, Valjevo, Serbia, dejan.beljic@vipos.edu.rs

Abstract: *In order to select database managmeent tool (i.e., DBMS) for information system developed for Valjevo Business School, testing of two available options (i.e. MS SQL and MySQL) was performed. These DBMS were examined in three different environments: on a stand alone workstation, within a local area network and on internet. Instead of using available benchmark software, simple software to test features relevant to the system was developed. The aim was not to provide general evaluation of the tested DBMS but to evaluate specific features and to determine option more relevant for the specific situation and environment. This paper describes applied methodology and displays realized results for the internet environment, since other situations were described and published earlier. In the first part of the paper applied strategy and methodology are presented. Then, the technical conductions under which the tests were conducted are specified. All results of the measurement conducted on internet using TCP/IP protocols are presented in detail. Finally, comparation of all three examined cases is displayed and discussed.*

Keywords: *Database management systems (DBMS), Microsoft SQL Server, MySQL, query response speed, performance evaluation, benchmark, local area network, internet*

1. INTRODUCTION

In the beginning of designing the integral information system for Valjevo Business School the question which DBMS (i.e., database management system) should be applied had arose. Different members of the development team had different suggestions. These suggestions depended on their experience and their leaning, so it can be told that they were subjective. Therefore, it was necessary to develop objective and neutral method and tool for evaluation which should give adequate and argumentative answer to the question. During the development of the method and the adequate software measurement tool, another question arose. It was necessary to determine and separate influence of the network protocols from performance of the tested DBMSs. This influence, as well as influence of the applied software framework, can lead to misleading and incorrect results.

It is important to emphasize that the main goal of this research was not to provide general, all-purpose evaluation of the tested DBMSs in various environment, but to provide estimation and arguments for a unique and specific case which was presented during development of the school's information system.

In this paper we present development and application of the method and tool as well as the achieved results, with special reference to network impact on the results. Three different situations are described: testing on the local stand-alone computer, testing within local area network and, finally, testing using the internet environment.

2. APPLIED STRATEGY

In order to be able to objectively determine the performance of a database management system, appropriate measurements must be made with a suitable benchmark software tool. These measurements should provide objective evaluation of the tested database management systems. According to Gray [1], high-quality database testing benchmark system should provide the following characteristics:

- relevance – it has to be adequate for the largest number of potential users;
- portability – it can be applied on many different (desirably all) existing database management systems;
- simplicity – it has to be simple, easy to use and not to consume too many resources;
- scalability – it has to be adequate for many different (desirably all) computer systems and architectures, large as well as small.

These characteristics can be seen as contradictory. For instance, portability feature is frequently contradictory regarding the simplicity feature. Accordingly, it is necessary to determine reasonable compromise among these characteristics [2].

There are many factors that can influence results of the measurements. Some of these factors include hardware components. Thus, core numbers, working frequency, CPU speed, the amount of RAM memory, hard disk drive speed, bus speed, memory access speed, quality of the mainboard, etc. can have a significant influence on the measured results. System software can also have an impact on the results (for instance the way operating system manages the memory, threading of locking). Other significant factors could be data schema, database configuration (dedicated cash memory for query processing, number of established connections to the database, network protocols dedicated for database access, index implementation, etc.), the amount of previously recorded data, type of database access application, etc. [3]

One possible approach to resolve the problem is to apply some of the existing benchmarking tools. There are many measuring software tools which can test and evaluate different database features. Some of the benchmarks are made for general purpose. On the other hand, some of the benchmarks are specialized for testing certain aspects of the tested database management system. Some are dedicated for transaction processing tests, the others are for relational databases or for object-oriented databases, there are benchmarks for testing XML based databases, for decision support systems (i.e., DDS) evaluation, for cloud-based databases, for evaluation of non-SQL databases etc. [2] [4] Due to the variety of benchmarking tools, it was necessary to establish a separate independent organization to deal with database benchmark standards. This organization is the non-profit corporation founded by Transaction Processing Performance Council – TPS and it issues standards for database benchmarks and verifies accuracy of benchmarking tools [4].

There are many database benchmarks available on the market today. Significant part of them is open-source or free software and their usage does not require additional funding. For instance, it is possible to use Quest [5], STS Soft [6], HammerDB [7] or any other. However, there is a significant issue related to usage of these benchmark tools as most of them are designed and developed for general purposes. Consequently, these tools test, measure and evaluate as many features as possible. Many of these features are irrelevant for the objectives of our project.

For instance, the school's system will have only few input spots so that evaluation marks dedicated for dealing with many connections to the database is in this case of no importance. Furthermore, the school's system will be available only within local area network and will not be available through internet, so that internet performance is completely irrelevant.

On the other hand, there are benchmark tools designed to be applied for certain specific purposes [8] as well as benchmarks which evaluate only small number of database management system features [9]. These tools are also not appropriate to determine the most suitable DBMS for the purpose of our project due to the limited scope of their application.

Having all this in mind, the development team has decided to adopt a different strategy. Instead of using the existing benchmarking tool(s), the team decided to develop a simple program to test and evaluate only features relevant to the development of the school's system [10]. Furthermore, the program was to be developed in the same environment and using the same technology as the school's system itself. The tool should be perfectly adjusted to meet and test evaluation criteria requested in this case. Of course, this tool would not satisfy Gray's criteria of portability and scalability [1] but these features are not relevant for our purpose. Anyway, the tool would to a large degree meet criteria of relevance (as designed and developed for a particular case it would be most relevant for this case) and simplicity (only features relevant for the particular case would be tested, therefore the tool should be simple and easy to use).

3. APPLIED METODOLOGY

Since the school's database is a relational one hence the testing tool should be designed to test relational databases. The main adopted evaluation criterium is query response speed [11]. Therefore, the testing tool is intended for measuring response speed to various SQL queries. This includes data retrieve queries (i.e., SELECT queries) as well as action queries (i.e., INSERT, UPDATE and DELETE queries). There were separately taken into consideration queries without any logical condition and queries containing simple or composite logical conditions (i.e., containing WHERE clause). Furthermore, there were separate measures for queries under single data table (i.e., relation) and under several tables (i.e., containing JOIN clause). The measurements were conducted under different amount of data already written in the database and with various amount of data to be written or changed. Queries were generated in a way that logical conditions were met by approximately 20% of records.

A selection of which database management systems should be tested was made according to technical characteristics of the school's equipment as well as to development team member's experience. The school had no valid license for some commercial DBMS like ORACLE, DB, IBM DB2, Informix etc. On the other hand, the development team members had no experience working on some other open source or freeware systems like PostgreSQL, Ingres, Informix etc. It was

estimated that introduction of these DMBSs would require increase of resource consumption (either in additional funds or in the extension of working hours), therefore these systems were not taken into consideration. For final testing two most common systems were selected: MS SQL Server (version 2016 Standard, 64-bit) and MySQL (version 8.0.16 – MySQL Community Server – GPL). Preliminary MS Access (version Professional Plus 2016) was also taken into consideration, but it achieved extremely poor results during preliminary measurements performed on a stand-alone workstation, so it was abandoned for the final testing in the network environment [12].

It was necessary to determine the amount of data in the relations. Empirically based, it was decided that all measurements would be performed up to maximum 10,000 records in the relations. Measurements under higher burden of records would be irrelevant for our research. Therefore, measurements were conducted under various number of records in the tables from 100 records gradually up to 10,000 records in the tables, as explained in [13]. For each combination total of seven measurements were performed. The highest and the lowest results were discarded. The arithmetic mean of the remaining results was accepted as the final result for each particular case.

4. TECHNICAL ASSUMPTIONS

To be as adequate as possible, the measurements were performed in real working environment at the school where the school's information system would be running. As mentioned before, three particular cases were examined.

The first one was testing DBMS and testing software running on the same single workstation. Thus, we minimize influence of network protocols, since no network traffic had occurred. The computer was run under Windows 10 Professional 64-bit edition, operating system. The processor was 64-bit AMD Ryzen 7 27000X on 3.7 GHz, with 8 cores. The motherboard was Gigabyte X470 AORUS 5 Wi-Fi. The computer had 16 GB RAM memory, Kingston SUV 500/480 SSD disk used as a primary partition and a Western -Digital hard disk drive capacity 4 TB with 3 partitions. The working partition (i.e., the one containing the software and data) had capacity of 976 GB.

The second case was performing tests within local area network (LAN). For connection to the testing databases only Windows Authentication was applied. In this way we avoid influence of TCP/IP protocols to the results. The testing was performed on the actual school LAN. Dedicated database server had Intel Pentium CPU G2130 processor, which run on 3.2 GHz and had 8GB of RAM memory. It run under Windows Server 2008 Enterprise without Hyper-V SP2 64-bit version operating system. Hard disk drive was divided into two partitions containing 232 GB each. Database management system run on the primary partition. Regarding software used for the development of the testing tool, it was developed in the programming language C#.NET, the very same as the one that was used for the school's information system development. Consequently, influence of the programming language on the measurements results could be ignored. Development framework was .NET Framework, version 3.5 which was adopted as a standard for every workstation within the school's local area network. Reason for this was combability among different Windows versions installed on working stations of the school's information system. Of course, the same framework version was used for the information system development. For the sake of compatibility, the final executable version of the testing tool, as well as of the information system, was adjusted for 32-bit processor. Integrated development environment used was MS Visual Studio Community 2017, version 15.9.14.

The final examined case involved data transfer through internet using TCP/IP protocols. The server hosts an SQL database, a web framework application that gives the client the ability to make an inquiry to the SQL database and return a response in the appropriate form. The server is connected to the Internet with a 20 / 20Mbs optical cable. There is a bridge device between the server and the Internet so there are no significant time delays in the internal infrastructure on the server side.

The computer configuration of the server is Windows server with OS Windows Server 2019 Standard, Microsoft SQL server 2019 64bit, MY SQL (MariaDB) 10.4.20, Tomcat Apache server 9.2, located on the internet. The internet speed is 20 / 20Mbs on optical cable with port speed 1G. The computer has CPU with 4 core, Intel Xeon E5-2620 2GHz, 64MB RAM ECC, LAN 1G, 2x SSD SAMSUNG 500GB M.2 NVMe MZ-V8V500BW 980 Series SSD 3500/3000 MB/s read/write speed and configured with RAID -1. The web portal/testing application was developed using eclipse, spring boot, java, java script, thymeleaf, hibernate, with basic security protocols.

The user's web browser connects to a web portal/server with the HTTP protocol, and uses standard GET/POST methods to exchange data. It sends the data to the client using the POST method in HTML format, the result is displayed in the form and in the grid. Although it is possible to compress data using the Gzip method, it was not used, so that we could compare the results with previous tests. We also did not use HTTPS security protocols and other communication elements provided by the TCP / IP protocol. The form displays the times when the user sends the request via the client application, the time when the request arrives on the server, the time required for the server to process the request, the time when the

response from the server to the client and the time when the data is displayed on the client side. The times are measured in nanoseconds, so that the time shown by the client computer or the server computer is not taken, in order to avoid the error in the time difference on the client server computers. The user who tests should select the table he is processing, the activity he wants to perform on that table and the number of records he is processing. After that the test should be run. As the application could not be made to work with both databases at the same time, and to have the same conditions for both types of SQL database, a separate application was created for each SQL database.

On the client side, there is an Internet connection of 75/10Mbs, the computer is connected via a router to the Internet. The computer configuration is Intel I5-9400F CPU 2.9GHz, 32GB RAM, SSD M.2 Patriot viper VPN100, motherboard Gigabyte Z390. A web browser is used to access the web portal on the server. Microsoft Edge, Google Chrome and Firefox Mozilla were used here.

The application testing time was chosen so that there were no users on the server side, in order to avoid additional unwanted server overloads and show only the impact of the internet connection on the data transfer from the server to the client. The times displayed refer to the speed from the moment the request is sent by the user to the server to the moment the activity completion information is displayed by the web browser.

Since the web browser works asynchronously, i.e., it displays the data as it arrives, we have ensured that the time we measure is always taken when the last data is displayed on the screen. This means that we also measured the time it takes for the application client to display data to the user.

5. THE RESULTS PRESENTATION

Results for the first two cases are explained and given in detail in [12] for measurement performed on a single workstation, and in [13] for testing within a LAN environment. In this paper we provide results in detail for the third case, connection to DBMS via internet, using TCP/IP protocols.

5.1. Test results of INSERT records into the table

This testing involves sequentially record data into a database table. The process takes place through a single transaction, which ends with the last record of data. The results are given in Table 1. Graphical interpretation is provided in Figure 1.

Table 1: Display of the time of INSERT records in a table in seconds

Number of records	10	100	250	1000	2500	5000	7500	10000
MS SQL	0.342	1.207	2.377	7.313	17.496	37.213	53.248	68.333
MY SQL	0.343	1.351	2.689	8.867	22.654	45.065	65.849	88.064

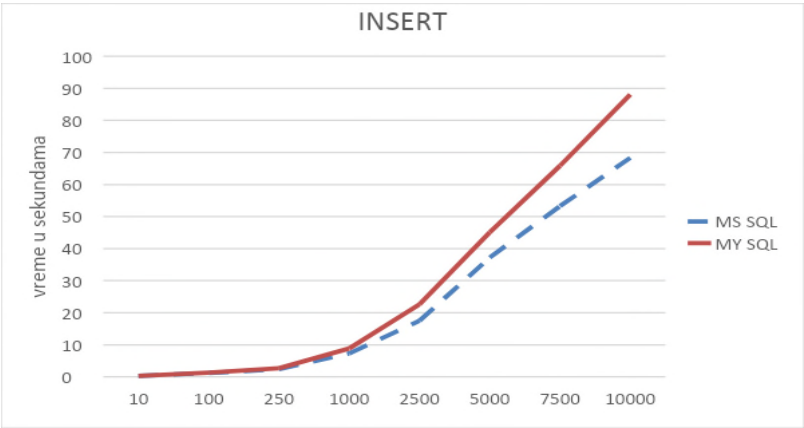

Figure 1: Results of measuring INSERT records in a table (in seconds)

Figure 1 shows that the advantage in the speed of INSERT records is on the MS SQL side over MY SQL, when increasing the number of records to be written to the table.

5.2. Test results of UPDATE records in the table

This measurement involves updating a subset of the data from the number of records entered in the table. The results are given in Table 2. Graphical interpretation is provided in Figure 2.

Table 2: Display of the time of UPDATE records in the table in seconds

Number of records	10	100	250	1000	2500	5000	7500	10000
MS SQL	0.288	0.176	0.184	0.191	0.273	0.25	0.419	0.34
MY SQL	0.229	0.228	0.192	0.228	0.274	0.298	0.353	0.408

Figure 2: Results of measuring UPDATE records in the table (in seconds)

Although the speed of UPDATE is almost linear in My SQL, we see that there are deviations in MS SQL. The amount of data that is sent by the server to the client is small, i.e., it contains information about the elapsed times in the measuring points. These deviations are consequences not only to the time required to process the data on the foreign SQL server, but also to the processed results to reach the client and be displayed to him. As we did not show individual times in these measurements from the sending of the client's request to the display of the response, the influence of the Internet on the display of information to the client can be seen here.

5.3. Test results of DELETE records from the table

Deletion includes the time required to delete part of the data from the number of records in the table. The results are given in Table 3. Graphical interpretation is provided in Figure 3.

Table 3: Display the time of DELETE records from the table in seconds

Number of records	10	100	250	1000	2500	5000	7500	10000
MS SQL	0.245	0.226	0.167	0.159	0.286	0.268	0.308	0.229
MY SQL	0.237	0.177	0.232	0.227	0.199	0.198	0.2	0.277

Figure 3: Results of measuring DELETE records from the table (in seconds)

The picture shows large deviations in processing - deletion. These deviations are not only in MS SQL but also in MY SQL. Here, as with UPDATE, we can see the impact of the Internet and the speed of data transfer to the client. The explanation is that if we know that with the increase in the number of records that are to be processed on the server side the processing time should be longer, and that we have deviations on the client side, we come to the conclusion that the flow on the Internet influenced the obtained results. The measurements were repeated several times and the mean value of the obtained results was shown.

5.4. The results of testing a simple SELECT records from the table

The WEB portal uses appropriate tools, classes to access data from the database. Some of the tools are Entity Manager, Persistence context. Each SELECT data from the table converts the obtained result into objects (class) that are further worked on in programming. If one wants to get a complex structure through SELECT, Entity Manager will return an object that keeps that structure in the working memory and enables faster and easier work with those objects. The portal converts the objects obtained in this way into the appropriate format and sends them to the client for display in a table. The results are given in Table 4. Graphical interpretation is provided in Figure 4.

Table 4: Display of the time of SELECT records from the table in seconds

Number of records	10	100	250	1000	2500	5000	7500	10000
MS SQL	0.356	0.723	1.402	4.141	10.22	19.539	28.93	38.843
MY SQL	0.23	0.622	1.284	3.975	9.632	19.235	29.066	43.13

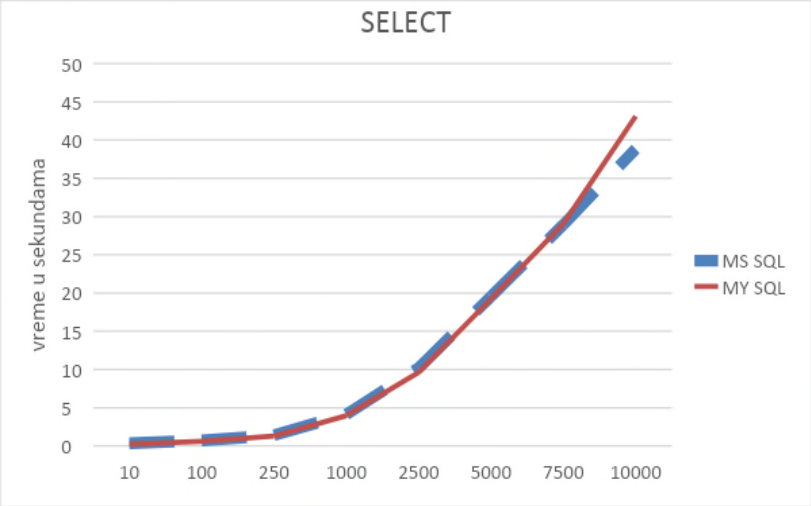

Figure 4: Results of measuring SELECT records from the table (in seconds)

In simple SELECT, we take the maximum number of records created by the insert into the table, and display them to the user. Although in the real system the display of all data is not done at the same time, but instead only a part of the data is displayed, we tested the time required for the delivery of all data by the server. As already mentioned, the final time represents the time that has elapsed from the moment the client sends the request, through the web portal to the SQL server, returning the results to the client side to the moment the final data is displayed to the user on the screen.

With SELECT we see that the speed of processing and obtaining data on the client side is almost the same - up to 7500 records, and that with a larger number of records the time for processing or sending data to the client is slightly reduced for MS SQL, and with MY SQL slightly increases.

5.5. The SELECT test results with JOIN records from multiple tables

A complex SELECT with data from multiple tables is designed to always return the specified number of records. The data sent to the user are objects that contain all the relationships from the database itself. The results are given in Table 5. Graphical interpretation is provided in Figure 5.

We can see that the speed of obtaining data by MS SQL is higher with a larger amount of data, where it is necessary to return to the user all objects and their relations in order to create objects for the client.

Table 5: Display time SELECT including JOIN records from tables and WHERE logical condition in seconds

Number of records	10	100	250	1000	2500	5000	7500	10000
MS SQL	0.356	0.723	1.402	4.141	10.22	19.539	28.93	38.843
MY SQL	0.23	0.622	1.284	3.975	9.632	19.235	29.066	43.13

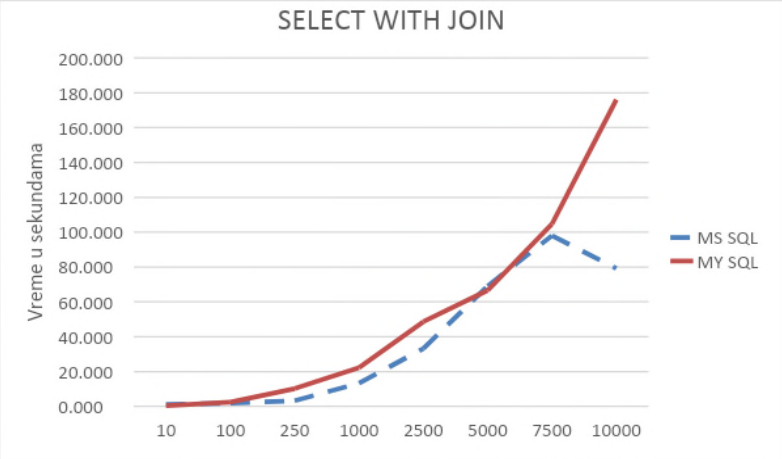

Figure 5: SELECT measurement results with JOIN records from tables (in seconds)

5.6. Comparation of the measurement results for all three environments

Finally, we can compare response time for all examined environments. Average time response depended on a number of records in the database (i.e., arithmetic means of all performed measurements) as well as average delays for all three tested environments (i.e., stand-alone workstation, response within LAN environment, database connection through internet using TCP/IP protocol) for Microsoft SQL Server are given in Table 6.

Table 6: Average results and delay times in seconds for MS SQL Server

	INSERT	UPDATE	DELETE	SELECT	SELECT/JOIN
MS SQL local WS	1.72	0.01	0.37	0.04	0.11
MS SQL LAN	4.06	0.92	0.80	0.57	6.79
MS SQL Internet	23.44	0.27	0.24	13.02	37.40
MS SQL LAN delay	2.34	0.91	0.43	0.53	6.68
MS SQL Internet delay	21.72	0.26	-0.13	12.98	37.29

Same data are displayed for MySQL in Table 7.

Table 7: Average results and delay times in seconds for MySQL

	INSERT	UPDATE	DELETE	SELECT	SELECT/JOIN
My SQL local WS	7.98	0.01	0.64	0.1	0.14
My SQL LAN	9.89	1.19	1.06	0.72	7.35
My SQL Internet	29.36	0.28	0.22	13.39	53.91
My SQL LAN delay	1.91	1.18	0.42	0.62	7.21
My SQL Internet delay	21.38	0.27	-0.42	13.29	53.77

These results are not what we were expecting. While the response speed is the most rapid in local workstation case, then in LAN environment case and, finally, the slowest for internet environment, as expected, in case of performing INSERT, SELECT and SELECT with JOIN clauses, in the same time results for applying UPDATE and DELETE statements are unusual.

In case of UPDATE statement internet response is significantly faster than response speed in the local network environment. Even more surprising results we've got measuring response speed performing DELETE clause. As shown in Table 6 and Table 7, response time is faster in internet environment than in case of performing the tests on stand-alone workstation. These tests were repeated several times, but the results still remain the same. The explanation of these anomalies should be provided through the future research.

6. CONCLUSION

It can be concluded that, in general, MS SQL Server provides slightly better performance than MySQL in the internet environment. This difference increases in case of larger databases, with larger number of records. Anyway, the difference between the two is not significant for practical use.

Execution of INSERT and SELECT clauses is significantly faster on local area network than on internet. These two commands are most frequently used in everyday work (for instance, for work with customers on the public counters). Therefore, this delay should be taken into consideration when designing internet applications. Still, internet is somewhat faster in case of updating and deletion of records, but these operations are seldom performed.

During this research it was shown that it is, in the decision- making process, possible to rely on small but efficient self-made testing tools adapted for real-life situation and adjusted for concrete requirements instead of depending on often inadequate and possible misleading benchmarking tools designed for general purpose [10].

The main contribution of this research, however, is to show and promote that it is possible to resolve a problem not only by using complex general-purpose tools, but by developing a smaller and simpler goal-oriented tools designed for a single specific purpose. Such development is much cheaper, faster and requires less resources. This strategy is important in the cases of software development under low-budget conditions [14].

REFERENCES

[1] GRAY, J. (ed.): *The Benchmark Handbook for Database and Transaction Processing Systems – 2nd edition*, Morgan Kaufman, 1993.

[2] DARMONT, J.: *Object Database Benchmark*, Encyclopedia of Information Science and Technology, I-III, Idea Group Publishing, pp. 2146-2149, 2005.

[3] MySQL: *MySQL Performance Benchmarks – Measuring MySQL Scalability and Throughput*, A MySQL Technical White Paper, March 2005, available at http://www.jonahharris.com/osdb/mysql/mysql-performance-whitepaper.pdf.

[4] DARMONT, J.: *Data Processing Benchmarks*, Encyclopedia of Information Science and Technology, Third Edition, Idea Group Publishing, pp. 146-152, 2014.

[5] Quest: *Benchmark Factory for Databases*, available at: https://www.quest.com/products/benchmark-factory/.

[6] STS Softy: *Database Benchmark*, available at: https://www.quest.com/products/benchmark-factory/.

[7] *HammerDB*, available at: https://www.hammerdb.com/.

[8] CUDRE-MAUROUX, P., KIMURA H., KIAN-Tat L., ROGERS J., MADDEN S., STONEBRAKER M., ZDONIK S.B., BROWN P.G.: *SS-DB: A Standard Science DBMS Benchmark*, XLDB 2010, Stanford University, CA, Oct. 6-7, 2010.

[9] McKNIGHT W., DOLEZAL J., BARKER R.: *Cloud Database Performance Benchmark, Product Profile and Evaluation: Actian Vector and Microsoft SQL Server*, MCG Global Services, February 2018.

[10] STANIŠEVIĆ I., VIĆENTIĆ M., OBRADOVIĆ S.: *MS SQL Server and MySQL Response Speed Comparation*, IJEEC - International Journal of Electrical Engineering and Computing, e-ISSN: 2566-3682, Vol. 4, No.2, pp.111-120, December, 2020

[11] STANIŠEVIĆ I., OBRADOVIĆ S., VIĆENTIĆ M.: *DBMS Response Speed Testing System*, International Scientific Conference UNITECH 2019, pp. II-39:II-43, Gabrovo, Bulgaria, EU, 15-16 November 2019.

[12] STANIŠEVIĆ I., OBRADOVIĆ S., VIĆENTIĆ M.: *Database Managment System Selection*, International Scientific Conference UNITECH 2019, pp. II-33:II-38, Gabrovo, Bulgaria, EU, 15-16 November 2019.

[13] STANIŠEVIĆ I., OBRADOVIĆ S., VIĆENTIĆ M.: *Poređenje brzina odziva MS SQL servera i MySQL-a*, 19th International Symposium INFOTEH, RSS-3-1, pp.216-221, Jahorina, BiH, 18-20 March, 2020.

[14] STANIŠEVIĆ I., OBRADOVIĆ S., *Characteristics of the Optimal Method for Software Design in a Low-budget Environment*, Megatrend Review, Vol 8,No 2, UDC 005.8:004.41, ISSN 1820-4570, pp. 597-524, 2011.