International Journal of Innovations in Engineering and Technology (IJIET)
http://dx.doi.org/10.21172/ijiet.102.04

Analysis Of The Disrtributed Databases In Different Environment

Vladimir Saso[1], Srdjan Jovkovic[2],Borivoje Milosevic[3], Nikola Davidovic[4]

[1,2]*University of Alpha Belgrade, Serbia*
[3]*University UNION - Nikola Tesla, Belgrade, Serbia,*
[4]*University of East Sarajevo,Faculty of Electrical Engineering,Republic of Srpska, Bosnia and Hercegovina,*

Abstract-This paper will explore and compare Open Source MySQL and commercial Microsoft MS SQL solutions. A large number of authors use programming that dictates the trend of Open Source solutions.Theyallow programmers around the world to find possible solutions, simply by updating the codecontinually in orderto get to the ideal solution and the "ideal" code for the set problem. On the other hand, commercial solutions are offered by large companies such as, for example, Oracle, Microsoft, etc. Naturally, companies around the world use commercial security solutions that are hidden behind the corporationsmentionedabove. They decide to use commercial tools because they have technical support from these manufacturers in terms of solving some of the problems that can happen in the real environment of an information system.An application that imitates the distributed database management systemhas been programmed. In the first case, we used the method of organizing a database from the Open Source family, specifically the My SQL database, which we set up on the Apache server and accessed by using PHP. In the second case, we used the method of organizing a database from a closed-source family, specifically MS SQL, which was installed on Microsoft's IIS server and accessed via ASP.

Key words: Distributed database, Query, Execution time, Open source, Commercial solution, UQRT.

I. INTRODUCTION

Data stored in the database usually originates from different, mutually remote resources. The distributed database allows data to be collected and controlled at the place of their creation. Locations thus have full autonomy over their data, which improves the management and control of them. It is not a rare case that one who generates data actually uses them. Setting up the node of a distributed database at the location where the data is generated and used reduces the remote network communication significantly, thereby raising the performance of the entire system. Distributed systems from the aspect of capacity are much more powerful than centralized ones. The expansion of the system's capacity is reduced to adding a new node to a distributed system, which is a much simpler and cheaper procedure than replacing the centralized system with a larger one. Because of this feature, they are often used when a constant system growth is predicted. In the event of a malfunction on a distributed system, performance is reduced, but there is no complete shutdown, as in the case of centralized systems. Also, redirection of distributed systems (servicing, replacement of parts ...) can be performed while the system is active. Data located in a disabled location can be obtained from another location where they are replicated, and which is operational in the given location. All this affects the reliability of the whole system. Therefore, the use of distributed databases is prepped by those systems that require high reliability. The need for simultaneous use of several different databases, which are mostly of the same type, is a common case in real systems. Heterogeneous distributed databases enable the implementation of several different database types into one logical entity (a single distributed database). Repeatability significantly increases the availability of data and reduces the chances of their loss. However, the method of updating such data is much more complex, due to the necessity of updating all the copies of the data. Also, there is part of the system on which one of the copies of the object being updated can be dropped, which prevents simultaneous transmission of the update to all physical copies of the object. One way to solve this problem is to apply the downloaded update and the primary copy method. Working with a distributed database requires the use of distributed queries that can be executed on multiple locations in the database. It does not matter how and in what order a query will be executed. In order for the response time of the system to be acceptable, it is necessary to have a good algorithm for selecting the best query optimization strategy. Where to place a catalog (centralize it, make copies of copies in all locations, place each part on each location ...) is another problem of distributed databases.

II. BASIC SETTINGS

The laboratory for carrying out the described analysis consists of:

2.1 Hardware Used In The Work:

Network Internet connection 100 mbps
Computer 1 with features: Mobile AMD Sempron ™, Processor 3600 +2.01 GHz, 2 GB RM
Computer 2 with features: Intel® Celeron® CPU 2.13 GHz, 1GB RAM
Computer 3 with features: Intel® Atom ™ CPU D425 1.8 GHz, 1GB RAM
Computer 4 with features: Intel® Core ™ CPU 6200, 2.13 GHz, 2GB RAM- Ethernet Switch 1/8 TP LINK TL - SF 1008D25.2

2.2 Software Used In The Work:

Installation of the Apache server version 2.2.11
Installation of the Microsoft IIS server that is part of the network operating system Microsoft Windows Server 2008
Installation of the Work bench program 5.2
Microsoft SQl Server Management Studio 2008 R2
Microsoft Visual Web Developer 2008 express edition
Work parameters that will be references for analysis and measurements to be examined are: efficiency of realization of queries over the database, the degree of required reliability of the operation of the SUBP and the availability of database usage, the degree of required protection of the database against unauthorized access and destruction or damage. To give a more credible and detailed illustration of the comparison of these two solutions, i.e. Open source and commercial solution on the other side, I will present the practical part of the paper in two parts:

Part I: The complete database is on one server, Scenario 1
Part II: The same database is distributed on three servers, Scenario 2.

III. OPEN SOURCE SOLUTION – SCENARIO 1

The database is on a single server, the Server is a MSI (Mobile AMD Semprom ™ processor 3600 + 2.01 GHz, 2GB RAM) and it represents the Web server on which the complete database is located. Apache server version 2.2.11 is installed on the server. The database is designed using MySQL language. To display the data, one HTML file was used in which the code is inserted and one PHP file is inserted, which is used for pulling and displaying data from the database. The database is accessed via the Internet through a remote client.
What we are testing and what we are interested in is the time of execution of the query, i.e. database response. Hypothetically, the database response for a different input of the code must be different because the number of alphanumeric data for each predefined table is not the same, so the response time of the database needs to be somewhat different. And the experimental method confirms this, Table 1, Figure 1.

Table 1 Execution time of the query T1….T5

Solution	T1 (execution time sec)	T2(execution time sec)	T3 (execution time sec)	T4(execution time sec)	T5 (execution time sec)
Open source	0,030074	0,019009	0,021409	0,027407	0,015472

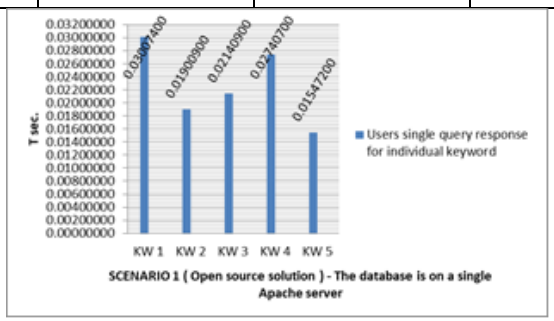

Fig. 1 The database is on the single server

3.1 Measuring performance of Apache server via "ab" server tools

The assumption is that we have 100 consecutive requests (requests that follow one at a time in a very short time interval), which are referred to the string KW5. The results of the test will be read from Figure 2. Among other things, it can be seen that the test time is 0.219 seconds, which means that it is possible to execute 457.14 requests per second, as illustrated in Figure 2. We see that we needed 2,188 ms for one request. The download speed is 556.7 kbytes / s.

We will now test if there were 100 competing demands and 1000 consecutive requests. Under competitive requirements, there are 100 simultaneous accesses to the database.The results of this test are shown in Figure 2. As we can see from the picture, the time required for testing is 2,234 seconds. The number of requests per second dropped to 447.55 / s. This results in an increase in the time required for one request at 2,234 ms, while the download speed has decreased to 545,02kbytes / s.

The conclusion is very easy to make. There were logical consequences of the server performance drop compared to the first case. If in this way the server loads enough to a certain limit, serious degradation of performance and server stabilitywill occur.

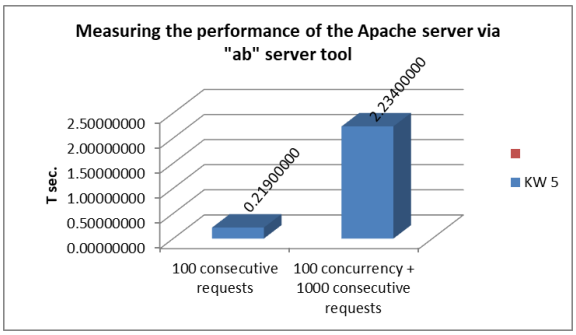

Fig. 2 Performance of the Apache server via ‘’ab’’ tool

IV. OPEN SOUCE SOLUTION - SCENARIO 2

One server on which the complete database is located can be recovered by dividing the database into three servers that are interconnected. Base response will be slower, but server reliability is obviously noticeable. We will prove this conjecture, conceive it, that is,support it by practical realization and measurements.

On the basis of the results obtained, when calculating the execution time of the query, we will calculate the mean time of the query, table 2, figure 3.

Conclusion and comparison with the measurements obtained in Scenario 1: In Scenario 1, the response time was 0.0226742 sec. and now it has been increased to 0.101162 sec. This is a significant increase considering percentage, bearing in mind that it is a database that does not have a conditionally stated excessive number of data, which leads to the conclusion that the larger the database (it assumes that there are a larger number of data, a larger number of tables) the longer the time for the execution of the query. There was a logical consequence. However, at the expense of more time executing the query this measure takes a lot of server relieving.

Table 2 Execution time of the query T1….T5

Solution	T1 (execution time sec)	T2(execution time sec)	T3 (execution time sec)	T4(execution time sec)	T5 (execution time sec)
Open source	0.057011	0.256593	0.070374	0.0552241	0.101162

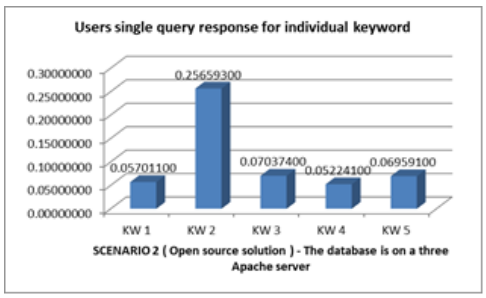

Fig. 3 UQRT for database on three servers via ‘’ab’’ tool

4.1 Measuring performance of Apache server via "ab" server tools

The assumption is that we have 100 consecutive requests. The following figure 26.5.1 shows the results of the test, so we will interpret them and give an appropriate comment.

We will now test if there were 100 competing demands and 1000 concurrent. Under competitive requirements, there are 100 accesses on the same page, and 1000 requests from clients requesting other information from the server. As we can see from the picture, the time required for testing is 2,438 sec (0.297 sec in the first case). The number of requests per second is 410.26, and in the first case it was 336.84.

This results in an increase in the time required for a single request of 2.438 ms (from 2.269 ms in the first case), while the download speed is much reduced to 162.26 kbytes / sec (in the first case it was 545.02 kbytes/s).Addition of comment: During the different time periods, there were minor variations in tolerance, due to the general load of the processor, and RAM on the servers, figure 4.

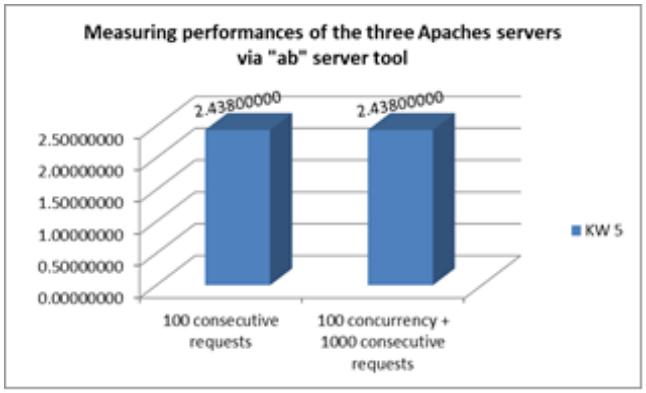

Fig. 4 Performance of tree servers via ‘’ab’’ tool

V. COMMERCIAL SOLUTION – SCENARIO 1

The database is on a single server, the server is a MSI (Mobile AMD Semprom ™ processor 3600 + 2.01 GHz, 2GB RAM) and it represents the Web server on which the complete database is located. The server has a network operating system Microsoft Windows server 2008, on which the IIS server is installed. The database was designed using the Microsoft MS SQL Language version 2008 and used the ASP.Net program code. The SQL Management Studio application was used to configure and populate the database.Programming was done in ASP. Net was made through Microsoft Visual Developer 2008 Express Edition.Within the IIS,web sites are realized within the basic web site via virtual directories, Figure 5. Now we have access to the centralized database from the client (because it is currently on a single server). We previously inserted the record into the host file on the Web client. Through the trace function, we will monitor certain parameters when calling the function. Access to a particular page will be done using pre-defined numbers that are numeric string (1,3,5,7,9), Table 3.

Table 3 Comparative query execution times for Commercial solutions

Solution	T1 (execution time sec)	T2(execution time sec)	T3 (execution time sec)	T4(execution time sec)	T5 (execution time sec)
Microsoft	0,013187	0,013822	0,017719	0,013532	0,011398

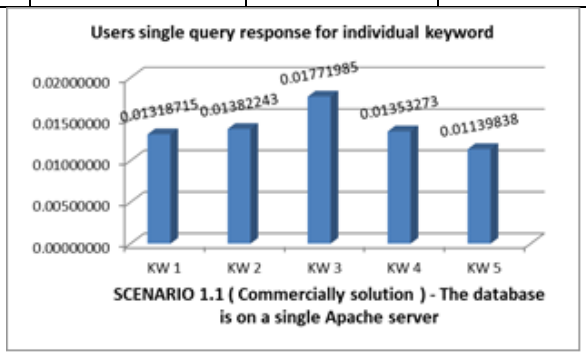

Fig. 5 Database is on s single server

With Open Source solutions Tave. = 0.0226742 sec, and in the commercial solution Tave. = 0,0134688 sec. This means that in a commercial solution, the query execution speed is shorter, and in this test the commercial solution has won.
Comparative query execution times for Open Source solutions and Commercial solutions are given in Table 4. The following table 4 shows the tabulated results of the measurement in Scenario 1 and Scenario 2 when it comes to Microsoft's platform.

Table 4 Comparative query execution times for Open Source solutions and Commercial solutions

Solution	T1 (execution time sec)	T2 (execution time sec)	T3 (execution time sec)	T4 (execution time sec)	T5 (execution time sec)
Open source	0,030074	0,019009	0,021409	0,027407	0,015472
Microsoft	0,013187	0,013822	0,017719	0,013532	0,011398

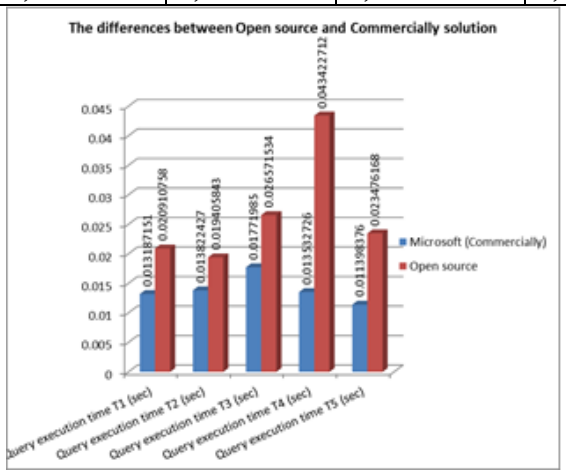

Fig. 6 Open source & Commercial solutions

VI. COMMERCIAL SOLUTION – SCENARIO 2

Through the Database Explorer in Visual Web Developer, we will display a response analysis of the distributed database.
The following figure 7 shows results when the database is distributed on three computers. When the same database is distributed on three servers we can obtain the following as in, Table 5:

Table 5 Tabulated results of the measurement in Scenario 2

Scenario	T1 (execution time sec)	T2 (execution time sec)	T3 (execution time sec)	T4 (execution time sec)	T5 (execution time sec)
Scenario 2	0,020910	0,019405	0,026571	0,043422	0,023476

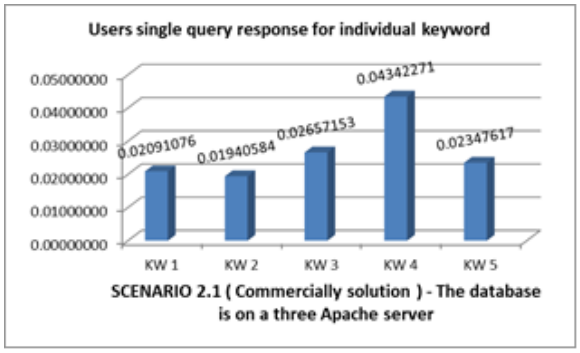

Fig. 7 The database is on a three servers

VII. CONCLUSIONS

Table 6 and figure 8 give comparative results of Scenario 1 and Scenario 2 respectively. By looking at the query execution times for certain pages in the Table 7 and Figure 8, it is seen that in Scenario 2, the time required to respond to the database is logical, but the server reliability 1 is high because the database is now distributed to three servers. Of course, in cases where faster and faster servers are used, at higher bandwidth of the Internet, this time decreases, but in any case, the results presented reflect fully the difference between these two scenarios.

Table 6 Tabulated results of the measurement in Scenario 1 and Scenario 2

Scenario	T1 (execution time sec)	T2 (execution time sec)	T3 (execution time sec)	T4 (execution time sec)	T5 (execution time sec)
Scenario 1	0,013187	0,013822	0,017719	0,013532	0,011398
Scenario 2	0,020910	0,019405	0,026571	0,043422	0,023476

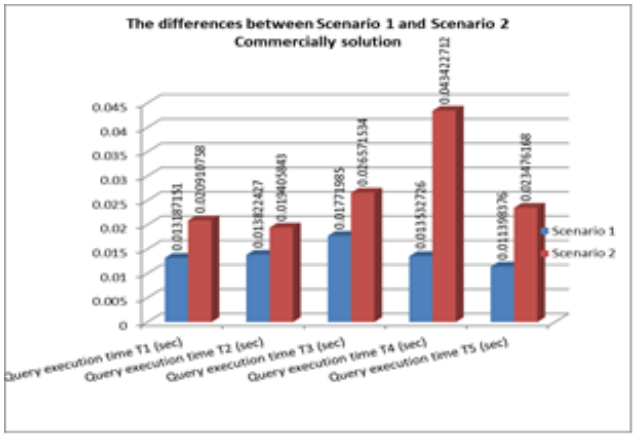

Fig. 8 Differences between Scenario 1 and Scenario 2

Solution of the Apache server using MySQL and PHP code from the point of view of security, stability, integrity of the entire distribution information system with the distribution of a part of the main database on several servers (in this case two) has excellent performance. The solution that can be made has the ability to upgrade, since it is an open source code, and I believe that Open Source solution, gives developers the ability to continually improve. When it comes to commercial solutions, as given in this paper on the Microsoft platform (IIS server, MS SQL, ASP), we got better performance using this solution. But it should be questioned whether commercial solutions have a "back door". The second is the fact that a commercial solution needs to be paid well, and the third is the fact that the commercial solution is conditional on additional services.

VIII. REFERENCES

[1] Abiteboul, S, Hull, R, Vianu, V. (2004): Fundation of Databases, Addison Wesley, Boston, MA
[2] Anderson, J.K. (2010): VISA Information System Security, Chicago Magazin, mar-apr, pg.35-38, University of Chicago Press, Chicago, IL
[3] Bernstein, A. Phillip, Newcomer, E. (2009): Principles of Transaction Processing, 2nd Edition, Morgan Kaufmann (Elsevier), ISBN 978-1-55860-623-4
[4] Jeffrey, A. Hoffer, M.B. Prescott, F.R, McFadden, (2005): Modern Database Management, Pearson, Prentice Hall
[5] Johnson ,L.James, (1997): Database: Models, Languages, Design, Oxford University Press, London, UK
[6] Kifer,M, Bernstein, A, Lewis, P.M. (2004): Database, Systems, Pearson, Addison Wesley
[7] Mullins, S. Craig, (2003): Administracijabazapodataka, Kompjuterbiblioteka, Čačak
[8] Thalheim, B. (2003): Fundamentals of ER Modeling, Springer Verlag, Berlin
[9] Weikum,G, Vossen, G. (2001): Transactional information systems: theory, algorithms, and the practice of concurrency control and recovery, Morgan Kaufmann, ISBN 1558605088
[10] Welling, L, Thomson, L. (2009): PHP and MySQL: Web Applications, Micro book, Belgrade, 2010.
[11] Michael Stonebraker, Paul M. Aoki, Robert Devine, WitoldLitwin and Michael Olson, Mariposa: A New Architecture for Distributed Data, Computer Science Div., Dept. of EECS, University of California, Berkeley, California 94720
[12] Justin DeBrabant Andrew Pavlo Stephen Tu, Anti-Caching: A New Approach to Database Management System Architecture, The 39th International Conference on Very Large Data Bases, August 26th - 31st 2013, Riva del Garda, Trento, Italy. Proceedings of the VLDB Endowment, Vol. 6, No. 14

[13] HoreaGrebla, Grigor Moldovan, Sergiu Adrian Darabant, AlinaCâmpan, DATA ALLOCATION IN DISTRIBUTED DATABASE SYSTEMS PERFORMED BY MOBILE INTELLIGENT AGENTS, Proceedings of the International Conference on Theory and Applications of Mathematics and Informatics - ICTAMI 2004, Thessaloniki, Greece
[14] Oszu,M.,Valduriez,P. Principles of Distributed Database Systems, 2nd Ed., Prentice Hall, (1998).
[15] Oracle® Database Administrator's Guide, 10g Release 2 (10.2) B14231-02
[16] Umesh Dubey, How to install Apache, PHP and MYSQL on Windows 10 Machine, https://www.znetlive.com/blog/how-to-install-apache-php-and-mysql-on-windows-10-machine/
[17] "About the Apache HTTP Server Project". Apache Software Foundation.Archived from the original on 7 June 2008.Retrieved 2008-06-25.
[18] "July 2016 Web Server Survey". Netcraft.Netcraft.Archived from the original on 10 August 2016.Retrieved 10 August 2016.
[19] MySQL Workbench, https://downloads.mysql.com /docs/workbench-en.a4.pdf
[20] Dave Stokes, MySQL Workbench, http://www.oracle.com/technetwork/mysql-hands-on-lab-403032.pdf
[21] Microsoft IIS Server Operation Guide for Windows, https://documentation.arcserve.com/Arcserve-RHA/Available/R16/ENU/Bookshelf_Files/PDF/XO_MS_IIS_W_ENU.pdf
[22] Using Visual Web Developer, http://ptgmedia. pearsoncmg.com/images/9780672327384/samplechapter/0672327384_CH03.pdf
[23] SQL Management Studio for SQL Server, https://www.sqlmanager.net/download/msstud