Uluslararası Sosyal Araştırmalar Dergisi / The Journal of International Social Research
Cilt: 12 Sayı: 63 Nisan 2019 *Volume: 12 Issue: 63 April 2019*
www.sosyalarastirmalar.com *Issn: 1307-9581*
Doi Number: http://dx.doi.org/10.17719/jisr.2019.3321

COMPARISON OF CALCULATING OPERATIONS PERFORMANCES ON DATABASE SERVER AND WEB SERVER

Yusuf Ziya AYIK[*]
Ferhat KAHVECİ[**]

Abstract

Many computer systems are combined to provide a service to users. Each system has specific tasks as part of the whole system. While some calculation operations can be processed on any server, some of them need a specific system platform to run. For the calculating operations which can be processed on any server, the main criteria will be related to the performance. If we have detailed information about the calculating performance of systems, we may set the calculating operations on the appropriate system. The aim of this study is the comparison of calculating operation on the database server and web server. Bubble sort and pi number algorithm were selected for the calculating operation. These algorithms' runtime results are obtained both by a database server and a web server. Algorithm functions with the same flow processes have been used for both server systems. The variables are predefined to prevent data transfer delay in systems. Thus, the performance results are focused on calculating performances on the servers. The results show, there is an appreciable difference between database and web server in terms of calculation time. The web server was the more performant than the database server, and the implementation of algorithms was simpler in the web server.

Keywords: Calculation, DBMS, Performance, Server, UDF.

1. Introduction

A specific service need of the users can meet by a single system. On the other hand, many systems which are integrated have the capability to provide various services. Most of the applications, especially web applications, need at least a web server and a database server. Web applications are a sort of server-side applications. The user request is sent to the related server as a task. The results of the server systems are presented by the Web applications.

Some of the tasks only can run on a specific server system, but on the other hand, there are tasks or calculations that can be run over any of the servers. At that point, programmers should focus on computation performance. If we know that the performance differences of the computing over the different systems, we may accomplish the selection of system properly.

2. Conceptual Framework

2.1. Server Systems

The server systems are designed to provide a certain service to many users. To provide this they have high hardware and software capabilities (Marinescu, 2013).

One of the most important specifications for a server system is the hardware. Generally, they have more than one CPU unit, high storage capabilities, high-speed network connection, recovery and backup systems, etc. Also, they are in a special room or in a data center. Hardware is a critical part for server systems, and a failure in hardware may affect all users (Lensu, 2013).

On the other hand, the server software is also as critical as hardware. The software should administrate the infrastructure properly. Also, clients communicate with server software to access related services. Therefore, the necessary software should be installed and configured. This software also defines the server type. Example, database server, web server, database server, network server, file server, application server, etc. (Kumar, 2011; Pressman, 2001)

[*] Associate Professor is with the Department of Computer Technology, Erzurum Vocational School, Atatürk University, Erzurum, Turkey (e-mail: ziyaayik@atauni.edu.tr).

[**] Ph.D. student is in Management Information Systems, Social Sciences Institute, Atatürk University, Erzurum, Turkey (e-mail: ferhatkahveci@hotmail.com).

2.2. Database Server

In daily life, every moment new data is being emerged. We need to store it in a proper environment for future use. In this way, whenever or wherever needed we can access it. To handle this task Database Management Software (DBMS) are used. Most of the DBMS use relational architecture to organize data. Also, technological developments are causing big data. To overcome in big data on-relational architecture started to be used. (Kumar, 2011; Pressman, 2001)

DBMS not only manages data, but it also manages security and performance. As well as, it has data manipulation tools like functions, procedures, packages, sequences, views, indexes, etc. (Kumar, 2011; Raphaely, 2013)

2.3. Web Server

Web servers store multimedia documents and publish over the network. These documents are called Web pages. They are coded with the web programming languages. The web server should support the web programming language of the web page. Source codes of web pages are compiled into the form of the users can understand. Usually, web servers need database servers. Web servers store data on the database servers and also can manipulate data in the database. When data is needed request data from the database server. There is high data traffic between them(Kumar, 2011).

2.4. Functions

Functional programming increases the quality of software in all computer programming languages. They return a result when they are called with the given variables. The calculations which are used frequently can be defined by functions. Thus, recurrences of the same code structures are avoided. When needed simply used with the name of functions and necessary variables. This also avoids code complexity. Depend on the software development platform, functional programming is implemented in a different layout for the same calculations. Because, each server software system has its particular structure for functional programming (Lorentz, 2005; Ashdown & Kyte, 2015; IEEE, 2014)

User-Defined Functions (UDF) move calculations from the application layer into the database. Usually, they are written in C or C++ as high-level languages for data mining and data analysis calculations. A UDF is called by a Structured Query Language (SQL) query and executed inside the database (Sundlöf, 2010; Crotty, et al., 2015; Ordonez, 2010; Tran, Diao, Sutton, & Liu, 2013).

2.5. Algorithm

Algorithms have the ability to solve a group of problems with the sequential calculations according to the coding flow. Calculation performance metrics usually measured by minimum memory, CPU and time usage. For the same problem, there are a variety of algorithms. Each one has its own advantages and disadvantages. While an algorithm can run in an optimal state by many variables, sometimes, unexpected a few variables can disturb the optimality of the algorithm. There are many well-known algorithms for common problems like sorting, finding, changing. Especially, data mining models rely on algorithms.

2.6. Computing performance

Computing operation usually executed over the system where it is used. If we have a possibility of executing computation over the different system, it will be suitable to do a performance comparison. Performance analyzes vary according to the system. Database servers have the capability of cost-based performance analysis. The cost information is related to input/output, processor, memory, and resource consumption. Web server computation performance measured by analysis of execution time. (Ayık & Kahveci, 2017).

3. Related Works

In the area of in-database computations, the master thesis of Sundlöf (Sundlöf, 2010) covers the analysis of computations inside the database. The thesis is focusing on user-defined computation operations inside the database. Also available a comparison between the UDF inside the database and C++ application outside the database. Both are computing a matrix algorithm which includes incremental factorization. The implementation in the database has a high complexity compared with C++ implementation. On the other side, computations inside the database resulted in better performance.

Ordonez (2010) has an introduction and reference monograph related to UDF subject. The article is comparing four fundamental computations with UDF, SQL query, and C++ with. These calculations are (1) linear regression, (2) PCA, (3) clustering, and (4) Naïve Bayes. The two summary matrices, which are the quadratic sum of cross-products points, and the linear sum of points are shown in all models mathematically. In the experiment, by analyzing exported files he compares UDFs and SQL queries inside the database and C++. In the research, the result is pointed out, UDFs, and SQL queries were faster than C++

considering computation times. UDFs highlighted their efficiency with the linear scale of precomputed summary matrices.

Broad knowledge about UDFs is presented in the paper of Tran, Diao, Sutton, and Liu (2013). They introduce data management, UDFs, and a learning approach based on Gaussian processes. An online algorithm which is also showing errors of computation is also devised. They claim that their proposed Gaussian processes can outperform the state of the art sampling approach for a variety of UDFs. Briefly, new approaches and applications in data management are introduced in the article.

Schrijver (2000) is proposed a well polynomial-time algorithm which is minimizing a submodular function. The algorithm is not including any linear programming method. He is presenting a polynomial in the size of the underlying set. The complexity of the values of function does not need any set for priority. At the same subject also, Iwata, Fleischer, and Fujishige (2001) presented as an algorithm minimizing a submodular function.

In the work of Iwata, Fleischer and Fujishige (2001) are presenting a good combinatorial polynomial-time algorithm which is minimizing submodular functions. They used a scalable scheme in a flow graph. The scaled parameters define the capacity in each graph set. The papers' main subject is running the algorithm in the time bounded by a polynomial. They are assuming that, a strongly polynomial-time which in the size of the set. Both studies of Schrijver (2000), and Iwata, Fleischer, and Fujishige (2001)are helpful for understanding algorithms from the view of the polynomial time.

Crotty, Galakatos, Dursun, Kraska, Binnig Cetintemel, and Zdonik(2015) are providing an architecture which is compiling UDFs automatically. They are also emphasizing several optimization techniques. They are generating UDF code step-by-step together with hardware criteria. The study shows performance improvement compared to alternative systems. However, scope primarily focuses on workflows of UDFs, on hardware configuration, and on dataset size. All kind of UDFs including well-known algorithms with optimized workflows rather than supporting specific UDFs are used.

Yan, Cheung, Yang, and Lu (2017) are offering techniques which reduce query time of database up to 91%, and response time up to 98%. They present a guide which needs less programming efforts for a performant web application development. To examine interactions between web applications and databases they are using a static analyzer tool. Static analyzer tool provides a general code review over the source codes of the web application. On the other hand, real-time interaction with a database cannot be examined.

4. Method

4.1. Purpose of Research

This paper covers algorithm performance tests both on the database server and on the web server. According to results, obtained programmers will have an opinion to run a proper algorithm on the proper server system.

4.2. Research model

In the research, the most known algorithms which can be run over both database and web server selected. As algorithm performance criteria, process complete time is used. The least process complete time is showing up the best performance.

Two kinds of algorithms are used in the test. One of them is the bubble sorting algorithm. This algorithm based on the comparison. Thus, it can be observed how long time the servers complete the comparative transactions. The other algorithm calculates the first six digits of the decimal part of pi. In this algorithm, it is selected to measure the calculation times of the two servers.

4.3. Data Collection

The cost-based analysis tool is used for the data collection from the data server, and the PHP function, which gives the given function`s execution time, is used for the data collection from the web server. Test codes of PHP bubble sort algorithm which is used in the study are given in Table 1.

Table 1: Php Bubble Sort Algorithm

```
$start timer = microtime(true);
$arr = array(0-999);
$counter= count($arr)-1;
for ($i=0; $i<$counter; $i++) {
for ($j=0; $j<$counter-$i; $j++) {
$k = $j+1;
      if ($arr[$k] <$arr[$j]) {
      list($arr[$j], $arr[$k]) = array($arr[$k], $arr[$j]);
    }
  }
}
      $stop timer = microtime(true);
      $timer = $stop_timer - $start_timer;

print("Bubble sort");
print_r($arr);
echo "Process Time: {$timer}";
```

Test codes of PHP Pi number calculation algorithm which is used in the study are given in Table 2.

Table 2: Php Pi Number Calculation

```
<?php
$plus = 0; $n = 1; $minus = 0;
$start_timer = microtime(true);

for ($i=1;$i<1000000;$i++) {
 $plus = $plus + 4 / $n;
 $minus = $minus + 4 / ($n + 2);
 $n = $n + 4;
}
$stop_timer = microtime(true);
$timer = $stop_timer - $start_timer;

print("Pi number");
print_r($plus-$minus);
echo "Process time: {$timer}";
?>
```

Test codes of Oracle bubble sort algorithm which is used in the study are given in Table 3.

Table 3: Oracle Bubble Sort Algorithm

```
DECLARE
Type arr1 IS TABLE OF NUMBER(3) INDEX BY BINARY_INTEGER;
arr arr1;
changed BOOLEAN;
tmp VARCHAR2(24);
start timer TIMESTAMP;
stop timer TIMESTAMP;
BEGIN
start_timer:= SYSTIMESTAMP;
arr(0-999) := 0-999;
LOOP
changed := false;
FOR i IN 2 ..arr.LAST LOOP
 IF arr(i - 1) >arr(i) THEN
   tmp := arr(i);
   arr(i) := arr(i - 1);
   arr(i - 1) := tmp;
   changed := true;
 END IF;
END LOOP;
EXIT WHEN NOT changed;
END LOOP;
FOR i in arr.FIRST .. arr.LAST LOOP
dbms_output.put_line(arr(i));
END LOOP;
sure bitir:= SYSTIMESTAMP;
dbms_output.put_line(stop_timer-start_timer);
END;
```

Test codes of Oracle Pi number calculation algorithm which is used in the study are given in Table 4.

Table 4: Oracle Pi Number Calculation

```
DECLARE
Counter integer;
Plus float;
n float;
minus float;
start_timer TIMESTAMP;
stop_timer TIMESTAMP;
BEGIN
start_timer := SYSTIMESTAMP;
plus := 0;
n := 1;
minus := 0;

  FOR counter in 1 .. 1000000 LOOP
   Plus := plus + 4 / n;
   Minus := minus + 4 / (n + 2);
  n := n + 4;
  END LOOP;

stop_timer := SYSTIMESTAMP;
dbms_output.put_line(stop_timer - start_timer);
dbms_output.put_line(plus - minus);
END;
```

In the bubble sorting algorithm, 1000 numbers with different value and mixed order are listed in ascending order. For calculation of pi value, one of the best-known formula discovered by Leibniz (1946 - 1716) is used.

$$\frac{\pi}{4} = \sum_{k=0}^{\infty} \frac{(-1)^k}{2k+1}$$ (Schrijver, 2000)

In the formula,∞a value considered as 1000000 to calculate the first six digits of the decimal part of pi which is 3,141592.

4.4. Findings

Bubble sort and pi number calculation algorithms results are given in this section. Figure 1, shows the execution time in seconds for algorithms. There is a clear calculation time difference between database and web server.

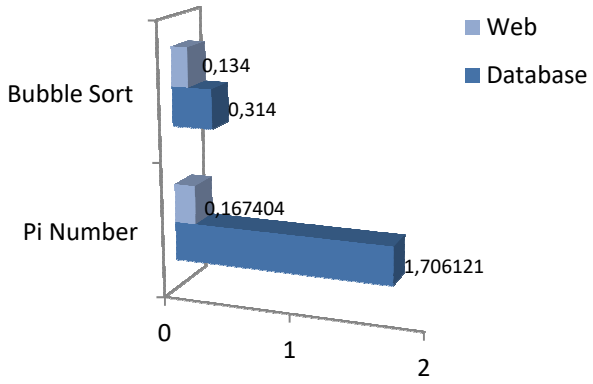

Figure 1: The execution time in Seconds for algorithms

5. Discussion and Conclusion

The calculating operation performed on the database server and the web server are compared in the study. For the calculating operations which can be processed on any server, the main criteria will be related to the performance. Bubble sort and pi number calculation, well-known algorithms, for the calculating operation are selected. These algorithms' runtime results are obtained both by a database server and a web server. Algorithm functions with the same flow processes have been used for both server systems. The variables are predefined to prevent data transfer delay in systems. Thus, the performance results are focused on calculating performances on the servers.

The graphic in Figure 1 shows that there is an appreciable difference between database and web server in terms of calculation time. The execution time in seconds for Buble Sort and Pi number algorithms are 0.134, and 0.167404 respectively for the web server. On the other hand, 0.314, and 1.706121 respectively for the database server. The web server was the more performant than the database server, and the implementation of algorithms was simpler in the web server as seen in Table 1 and Table 2.

We also noticed that there is another issue that may be subject to another study. While calculation difficulty increases, as in pi number calculation, the calculation time difference between the web and the database server is increasing. This shows that the database server can be used for only basic calculations. For the complex calculations, the web server has more stable calculation performance than the database server.

REFERENCES

Ashdown, L., & Kyte, T. (2015). *Oracle database concepts 11g Release 2 (11.2).* Oracle.
Ayık, Y. Z., & Kahveci, F. (2017). Improving effects of materialized view in database query processing and specific applications. *Ataturk University Journal of Economics & Administrative Sciences, 31*(5), 1057-1067.
Chirkova, R., & Yang, J. (2011). Materialized views. *Databases, 4*(4), 295-405.
Crotty, A., Galakatos, A., Dursun, K., Kraska, T., Binnig, C., Cetintemel, U., & Zdonik, S. (2015). An architecture for compiling udf-centric workflows. *Proceedings of the VLDB Endowment, 8*(12), 1466-1477.
Date, C. J. (2006). *The Relational Database Dictionary: A Comprehensive Glossary of Relational Terms and Concepts, with Illustrative Examples.* Sebastopol, California, USA: O'Reilly Media.
DB-Engines. (2016). *DB-Engines Ranking of Relational DBMS*. Retrieved 10 14, 2016, from DB-Engines: http://db-engines.com/en/ranking/relational+dbms
Ergüder, L. (2013, 06 25). *Materialized View*. Retrieved 10 11, 2016, from In java we trust: http://www.injavawetrust.com/oracle-ders-40-views-05-materialized-view/
Goldstein, J., & Larson, P.-A. (2001). Optimizing Queries Using Materialized Views: A Practical, Scalable Solution. *ACM SIGMOD Record, 30*(2), 331-342.
Gupta, A., & Mumick, I. S. (1995). Maintenance of Materialized Views: Problems, Techniques, and Applications. *The Bulletin of the Technical Committee on Data Engineering, 18*(2), 3-18.
Han, J., Kamber, M., & Pei, J. (2012). *Data Mining Concepts and Techniques 3rd Edition.* Waltham, Massachusetts, USA: Elsevier Inc.
IEEE. (2014). *SWEBOK version 3.0.* Piscataway, New Jersey, USA: IEEE Computer Society Products and Services.
Iwata, S., Fleischer, L., & Fujishige, S. (2001). A combinatorial, strongly polynomial-time algorithm for minimizing submodular functions. *Journal of the ACM (JACM), 48*(4), 761-777.
Jagadish, H. V., Mumick, I. S., & Silberschatz, A. (1995). View maintenance issues for the chronicle data model. *ACM SIGACT-SIGMOD-SIGART Symposium on Principles of database systems*. San Jose, California, USA.
Karde, P. P., & Thakare, V. M. (2010). Selection of materialized views using query optimization in database management: An efficient methodology. *International Journal of Database Management Systems (IJDMS), 2*(4), 116-130.
Kumar, B. A. (2011). Thin Client Web-Based Campus Information. *International Journal of Software Engineering & Applications (IJSEA)*, 13-26.
Kumari, N. (2012). SQL server query optimization techniques. *International Journal of Scientific and Research Publications, 2*(6), 1-4.
Lensu, V. (2013). *Building a secure IT server room.* Laurea Leppävaara: Laurea University of Applied Sciences.
Lorentz, D. (2005). *Oracle database online documentation, 10g Release 2 (10.2).* Redwood City, California: Oracle.
Marinescu, D. C. (2013). *Cloud computing theory and practice.* Waltham, USA: Morgan Kaufmann.
MEB. (2012). *Veritabanında Sorgular.* Ankara, Turquia: Ministry of Education.
MEB. (2013). *Veritabanı Yönetimsel Fonksiyonları.* Ankara, Turquia: Ministry of Education. MySQL. (2016). *Understanding the Query Execution Plan*. Retrieved 10 14, 2016, from MySQL: http://dev.mysql.com/doc/refman/5.7/en/execution-plan-information.html
Oracle. (2016). *Optimizer Statistics Concepts*. Retrieved 10 14, 2016, from Database SQL Tuning Guide: https://docs.oracle.com/database/121/TGSQL/tgsql_statscon.htm#TGSQL351
Oracle. (2016). *Query Optimizer Concepts*. Retrieved 10 14, 2016, from Database SQL Tuning Guide: https://docs.oracle.com/database/121/TGSQL/tgsql_optcncpt.htm#TGSQL192
Ordonez, C. (2010). Statistical model computation with UDFs. *IEEE Transactions on Knowledge and Data Engineering, 22*(12), 1752-1765.
Özkan, Y. (2013). *Veri madenciliği yöntemleri.* Istanbul, Turquia: Papatya Yayıncılık Eğitim.
Pilecki, M. (2007). Optimizing SQL Server Query Performance. *TechNet Magazine*.
Pressman, R. S. (2001). *Software engineering a practitioner's approach.* New York, USA: The McGraw-Hill Companies.
Rana, I. K. (2004). *History Of Pi.* Bombay, India: Department of Mathematics, Indian Institute of Technology.
Raphaely, D. (2013). *Oracle Database PL/SQL packages and types of reference, 11g Release 2 (11.2).* Oracle.
Schrijver, A. (2000). A combinatorial algorithm minimizing submodular functions in strongly polynomial time. *Journal of Combinatorial Theory, Series B, 80*(2), 346-355.
Sundlöf, C.-F. (2010). *In-database computations.* School of Computer Science and Communication, KTH Royal Institute of Technology.
Tran, T. T., Diao, Y., Sutton, C., & Liu, A. (2013). Supporting user-defined functions on uncertain data. *Proceedings of the VLDB Endowment, 6*(6), 469-480.
Uysal, A. Ö. (2011). Veritabanı sistemlerinde sorgu optimizasyonlarının veri analiz teknikleriyle geliştirilmesi. *Yıldız Technical University, Graduate School of Natural and Applied Sciences*.
Yagoub, K., & Gongloor, P. (2007). *SQL Performance Analyzer.* Redwood Shores, California, USA: Oracle Corporation.
Yan, C., Cheung, A., Yang, J., & Lu, S. (2017). Understanding database performance inefficiencies in real-world web applications. *CIKM*.