Chapter 3

Influential Parameters to the Database Performance — A Study by Means of Design of Experiments (DoE)

Eduardo Batista de Moraes Barbosa and
Messias Borges Silva

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/56546

1. Introduction

The growth of the Organization’s data collections, due to the development of new materials and advanced computing devices, puts the database (Db) technology at the forefront. Consequentially to its popularity, different options of database management systems (DBMS) can be found on the market to store one of the most important Organization’s assets, the data. Among the factors that may influence its choice (advanced features, interoperability, etc.), could be highlighted the cost-benefit provided by fierce competition between different software philosophies – proprietary (Oracle, MS SQL Server, IBM DB2, etc.) and public domain (PostgreSQL, MySQL, Firebird, etc.) – and, also the system performance in critical computing environments.

Performance measurements in the computer systems area (processors, operating systems, compilers, database systems, etc.) are conducted through benchmarks (a standardized problem or test used as basis for evaluation or comparison) widely recognized by the industry. There are a lot of the benchmarks consortia with specific evaluation criteria, metrics, pricing and results communication, highlighting: Open Source Database Benchmark (OSDB) (http://osdb. sourceforge.net), System Performance Evaluation Cooperative (SPEC) (http://www.spec.org) and Transaction Processing Performance Council (TPC) (http://www.tpc.org).

In the academic scope, the TPC benchmarks are widely recognized [1; 2; 3; 4; 5; 6; 7] due its exactness for definition of tests implementations, price measurements and results reporting. The TPC began from two *ad hoc* benchmarks formalization (DebitCredit and TP1) which resulted in the TPC BM^{TM} and the TPC BM^{TM} B [1]. Currently, with the benchmarks advance, its possible performs complex queries, batch and operational aspects of systems for transac-

INTECH
open science | open minds

tions processing through different benchmark standards, such as TPC-C (simulates a complete computing environment where a population of users executes transactions against a Db), TPC-DS (models several generally applicable aspects of a decision support system), TPC-E (uses a database to model a brokerage firm with customers who generate transactions related to the business), TPC-H (consists of a suite of business oriented *ad-hoc* queries and concurrent data modifications), TPC-VMS (leverages the TPC benchmarks by adding the methodology and requirements for running and reporting performance metrics for virtualized databases) and TPC-Energy (contains the rules and methodology for measuring and reporting an energy metric in TPC benchmarks).

In this chapter we intend to present a study on the Db performance by means of statistical techniques for planning and analysis of experiments (DoE), applied in the computing scope. The objective of this study is use two DoE techniques (2^k full factorial and 2^{k-p} fractional factorial) to investigates the influence of different parameters (related with Db memory tuning) in the Db performance. The DoE methodology will be applied at the case study, where the Db parameters will be simultaneously combined and tested and its results analyzed by means of full factorial and fractional factorial designs, and to assist in the investigations to determine how each parameter may explain (or take influence in) the Db performance. Thus, will be also addressed a comparison of results between the both techniques chosen. It should be noted that, in the scope of this study, the Db technology will be used as a vehicle to demonstrate how the DoE methodology can help in the design of experiments and its analysis and its use as a promising tool in several scopes, like in the computing science field.

The paper is structured as follows: Section 2 shows an introduction in the benchmark technology, with emphasis on the TPC-H standard. The DoE methodology is introduced in Section 3, where can be found a overview over full factorial and fractional factorial designs. The Section 4 is devoted to the case study to investigate the influence of different Db parameters (from PostgreSQL DBMS) in its performance through DoE designs (2^k full factorial and 2^{k-p} fractional factorial). This Section also presents the analysis and comparison of results. Some related work are presented in Section 5 and the final considerations are in Section 6.

2. Benchmark overview

Performance tests in the computing scope are a valuable tool to assist the decision makers in the hardware and/or software settings. Such tests, usually called as benchmark, can ensure that software does not present problems or unavailability due to insufficient resources (i.e.: memory, processor, disk, etc.).

According to the Merriam-Webster dictionary (http://www.merriam-webster.com), benchmark is "*a standardized problem or test that serves as a basis for evaluation or comparison (as of computer system performance)*". In the computing scope, benchmark is typically a software to perform pre-defined operations and returns a metric (i.e.: workload, throughput, etc.) to describe the system behavior.

Some generic benchmarks have become widely recognized (i.e.: TPC), such that vendors advertise new products performance based in its results. Benchmarks are important tools in evaluating computer system performance and price/performance. However, to be really useful [1] prescribes some key criteria that should be considered during the benchmarks' choice and/ or use:

- Relevance (must measure the peak performance and price/performance of systems when performing typical operations within that problem domain);
- Portability (should be easy to implement on many different systems and architectures);
- Scalability (should apply to small and large computer systems); and
- Simplicity (must be understandable).

Benchmark softwares, usually, simulates scenarios from the real world (i.e.: TPC-C, TPC-DS, TPC-E, TPC-H and TPC-VMS) where systematic procedures are performed to test, collect and analysis the system performance. It's use is strongly recommended, because the cost of implementing and measuring specific applications on different systems/platforms is usually prohibitive [1].

2.1. TPC-H Standard

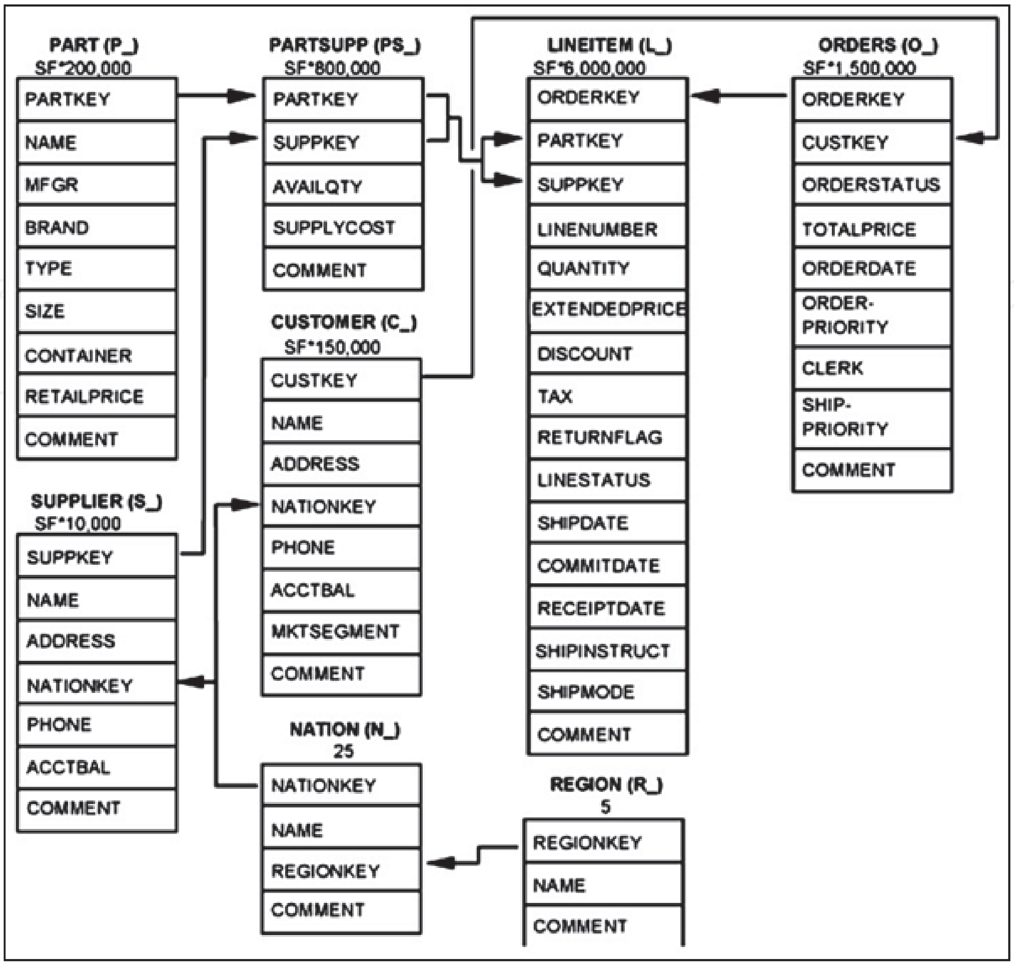
TPC-H benchmark represent a generic decision support benchmark, with the main features: capacity to manage very large amounts of data, the power to analyze it with a high degree of complexity and the flexibility to answers critical business questions.

Figure 1, illustrates the logical schema of the TPC-H specification and shows the business and application environment. According to the TPC official documentation [8], "*TPC-H does not represent the activity of any particular business segment, but rather any industry which must manage, sell, or distribute a product worldwide (i.e.: car rental, food distribution, parts, suppliers, etc.)*".

In this schema (Figure 1), the Db consists of eight tables simulating a realistic application involving customers, parts, lineitems, suppliers and orders. The prefix of the table columns is expressed into parentheses, the relationships between tables are represented by arrows and the number/formula below each table name are the cardinality (number of rows) of the table, factored by scale factor (SF). That is, the SF determines the size of the raw data outside the Db (i.e.: SF = 100 means that the sum of all base tables equals 100GB).

To be compliant with TPC-H benchmark, [8] recommends that the Db must be implemented using a commercially available DBMS, with support to queries and refresh functions against all tables on a 7 days of week, 24 hours by day (*7x24*). The minimum required to run the benchmark holds business data from 10.000 suppliers, with almost ten million rows representing a raw storage capacity of about 1GB (i.e.: SF = 1).

The performance metric reported by TPC-H is called TPC-H Composite Query-per-Hour Performance Metric (QphH), and reflects multiple aspects of the system's capability to process queries. TPC-H benchmark is composed by 22 *ad-hoc* business-oriented queries (16 of which carried from other TPC benchmarks) that include a variety of operators and selectivity



(SOURCE: TPC Benchmark^{TM} H documentation, URL: http://www.tpc.org)

Figure 1. TPC-H database schema.

constraints, whose the objective is to assist the decisions makers in the business analysis (pricing and promotions, supply and demand management, profit and revenue management, customer satisfaction study, market share study and shipping management) [3; 8]. A typical query (Table 1) uses tables "join" and, in most cases, aggregate functions and "group by" clause. The workload of benchmark consists of a data load, the execution of queries in both single and multi-user mode and two refresh functions.

3. Design of experiments overview

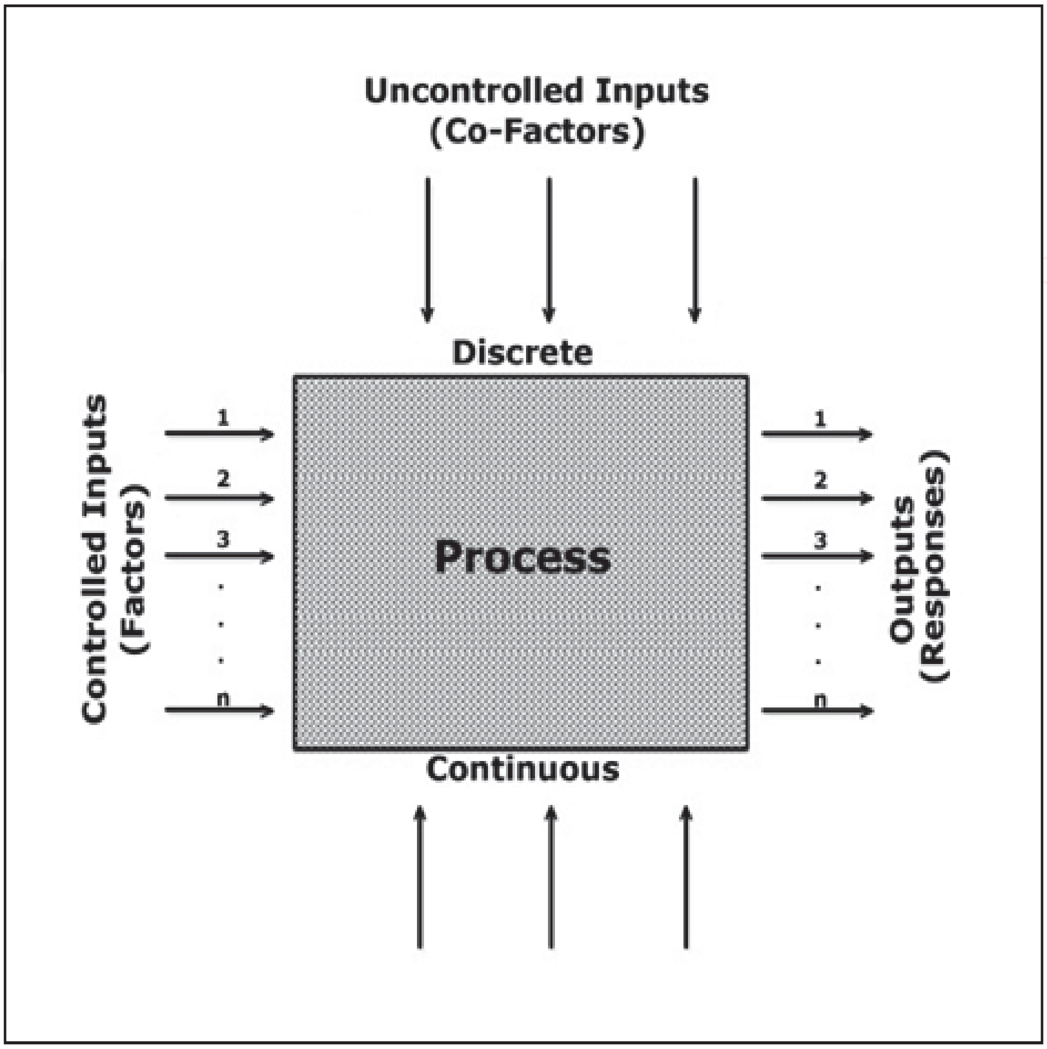
In science, the researchers' interest is focused on systems (processes) investigations where, usually, there are several variables for analysis. Often, the investigations are centered on individual changes produced by the variables, as well in their interactions (Figure 2).

Traditionally, in an investigation, experiments are planned to study the effects of a single variable (factor) in a process. However, the combined study of multiple factors represents a

Query	Join			Aggregate functions			Group by	Sub query
Avg()	Count()	Max()	Min()	Sum()				
Q1		✓	✓			✓	✓	
Q2	✓				✓			✓
Q3	✓					✓	✓	
Q4	✓		✓				✓	✓
Q5	✓					✓	✓	
Q6						✓		
Q7	✓					✓	✓	✓
Q8	✓					✓	✓	✓
Q9	✓					✓	✓	✓
Q10	✓					✓	✓	
Q11	✓					✓	✓	✓
Q12	✓					✓	✓	
Q13	✓		✓				✓	✓
Q14	✓					✓		
Q15	✓			✓		✓	✓	✓
Q16	✓		✓				✓	✓
Q17	✓	✓				✓		✓
Q18	✓					✓	✓	✓
Q19	✓					✓		
Q20	✓					✓		✓
Q21	✓		✓				✓	✓
Q22	✓	✓	✓			✓	✓	✓

Table 1. TPC-H queries' characteristics.

way to determine the main effects, as well as the interaction effects among the factors underlying the process. The DoE is a framework of statistical techniques, such as the results can produce valid and objective conclusions [9].



(SOURCE: NIST/Sematech, 2006)

Figure 2. A classical process.

DoE methodology have been used very successfully in the verification, improvement and reducing the processes variability with impacts on costs and development time [9; 10]. A relevant class of DoE techniques is called factorial design, whose goal is to study and analyze the results (effects) produced by multiple variables of a process.

The beginning of a factorial design is a careful selection of a fixed number of levels for each set of factors. The experiments should be performed with all combinations factor/level. For example, if there are l_1 levels to the first variable, l_2 for the second,..., l_k for the k-th factor, the full array of l_1, l_2,..., l_k plays will be classified as factorial design l_1 x l_2 x... x l_k.

The default schema for designs with two levels uses the notation "–" (negative) and "+" (positive) to denote the low and high levels of each factor, respectively [9; 10; 11]. For example, a 2x2 factorial design with two factors (X_1 and X_2) and two levels (low and high), requires four experimental plays (Table 2).

Exp.	X_1	X_2	Result
1	–	–	Y_1
2	+	–	Y_2
3	–	+	Y_3
4	+	+	Y_4

Table 2. Experimental matrix.

3.1. Full factorial design

When all combinations of factors are running at the same number of times for each level, the experiment is classified as 2^k *full factorial design*. Thus, the factorial design presented in Table 2 is classified as 2^k ($k = 2$) full factorial design [9; 10]. The most intuitive approach to study such factors would be vary the factors of interest in a full factorial design (trying all possible combinations of settings). For example, a 2^3 full factorial with three factors (X_1, X_2, and X_3) at two levels requires eight experimental plays (Table 3), while to study 5 factors at two levels, the number of runs would be $2^5 = 32$, and $2^6 = 64$, and so on. So, the number of runs required for 2^k full factorial design grows geometrically as k increases, and therefore even the number of factors is small, a full factorial design can become big quickly. In these circumstance, it is recommended [9; 10] to use *fractional factorials designs*.

Exp.	X_1	X_2	X_3	Result
1	–	–	–	Y_1
2	+	–	–	Y_2
3	–	+	–	Y_3
4	+	+	–	Y_4
5	–	–	+	Y_5
6	+	–	+	Y_6
7	–	+	+	Y_7
8	+	+	+	Y_8

Table 3. ($k = 3$) full factorial design.

3.2. Fractional factorial design

Fractional factorial designs represents one way where only a fraction of appropriate combinations required for 2^k full factorial designs is selected for execution. Fractional designs are commonly used when one wants to investigate k factors with smaller number (2^{k-p}) of experiments, where p is the reduction factor [9; 10]. For example, the 2^3 full factorial design (Table 3) can be re-written as a fractional factorial design$2_{III}^{3-1} = \frac{2^3}{2} = 4$, where 4 is the number of experimental plays required (Table 4). In the current example, the design is described as 2^{3-1} design of resolution III (three). This means that you study overall $k = 3$ factors, however, $p = 1$ of those factors were generated from the interactions of $2^{[(3-1) = 4]}$ full factorial design.

Exp.	X_1	X_2	X_3	Result
1	−	−	+	Y_1
2	+	−	−	Y_2
3	−	+	−	Y_3
4	+	+	+	Y_4

Table 4. ($k = 3$, $p = 1$) fractional factorial design.

The design does not give full resolution, that is, there are certain interaction effects that are confounded with (identical to) other effects. However, fractional designs requires a smaller number of plays as compared to the full factorial design, but they assumption implicitly that higher-order interactions do not matter. Therefore interactions greater than two-way particularly could escape of detection.

4. Case study

To illustrate the effectiveness of DoE methodology we will apply it in the Db scope, through a case study to know the influence of parameters in the Db performance. This case study will be divided by phases: the first, comprehends a study with a *full factorial design* at two levels, requiring 2^k ($k = 5$), 32 experimental plays. The second phase deals with a *fractional factorial design* 2^{k-p} ($k = 5$ and $p = 1$) resolution *V*, requiring 16 experimental plays. All proposed experiments will be performed at the same computing environment according to the techniques previously chosen.

Thus, this case study uses the PostgreSQL DBMS (version 8.4.11), through the implementation of a database of 1GB, populated with *dbgen* application (SF = 1) from TPC-H benchmark. Between the 22 queries provided by TPC-H benchmark, we choose to use four queries with a common SQL feature (i.e.: tables “join”, aggregate functions and commands to grouping and ordering data) and mostly related to the customer satisfaction study:

- Q10 – identifies customers who might be having problems with the parts that are shipped to them;
- Q13 – seeks relationships between customers and the size of their orders;
- Q18 – ranks customers based on their having placed a large quantity order; and
- Q22 – identifies geographies where there are customers who may be likely to make a purchase.

In this study, the parameters selected (Table 5), intuitively, looks be significant to the Db performance according to the queries characteristics.

	Parameters	Low (–)	High (+)
A	*shared_buffers*	32MB	1024MB
B	*temp_buffers*	8MB	64MB
C	*work_mem*	1MB	1536MB
D	*wall_buffers*	64KB	1MB
E	*effective_cache_size*	128MB	1536MB

Table 5. Design factors.

The PostgreSQL parameters (experiments factors, Table 5) were set according to the suggestions from the PostgreSQL official documentation (http://www.postgres.org), and the values from low level, are standards of installation, while the high level values, were customized according to the computing environment characteristics (a virtual machine implemented over Intel Core^{TM} i5 360 3.20GHz CPU, GNU/Linux i386 openSUSE 11.3, 2GB RAM and hard disc 50 GB). That is, *shared_buffers* (amount of memory reserved for data cache) was set to 50% of total memory; *work_mem* (amount of memory used for sort operations and hash tables) and *effective_cache_size* (amount of memory available for disk cache used by the operating system and Db) were set to 75% of total memory; *temp_buffers* (maximum number of temporary buffers used by each Db session) and *wal_buffers* (useful for systems with high need to write to disk) have 64MB and 1MB, respectively.

4.1. Phase I – 2^k full factorial design

The experiments performed in this phase were structured with five factors at two levels (Table 5), resulting in a 2^5 *full factorial* (32 experimental plays). Each experiment is composed of two replicates and a sample of the experimental matrix, whose the results are the execution time (in seconds) – average time of queries answers 2_{III}^{k-p} – is presented in Table 6. In this table (Table 6), each column contains – (negative) or + (positive) signs to indicate the setting of the respective factor (low or high, respectively). For example, in the first run of the experiment, set all factors A through E to the plus setting, in the second run, set factors A to D to the positive setting, factor D to the negative setting, and so on.

In Table 7 can be found the main effects of factors ($\left(\mu = \frac{1}{N}\sum_{i=1}^{N} t_i,\ N = 3\right)$, where E = effect, f = factor [$A..E$] and Q = query), measured from each query (Q10, Q13, Q18 and Q22). The effects of factors were calculated by the sum of multiplying levels (– and +) by execution time (Y) across all 32 rows. Thus, for query Q10, the effect of factors A and B were estimated as:

Exp.	Factors					Execution time			
	A	B	C	D	E	Q10	Q13	Q18	Q22
1	–	–	–	–	–	0.132	4.122	0.109	0.840
2	–	–	–	–	+	0.126	21.710	0.109	1.067
3	–	–	–	+	–	0.137	4.198	0.113	1.029
4	–	–	–	+	+	0.118	22.052	0.112	1.069
5	–	–	+	–	–	0.185	3.033	0.117	2.553
6	–	–	+	–	+	0.160	3.148	0.119	2.052
7	–	–	+	+	–	0.223	4.097	0.115	1.776
8	–	–	+	+	+	0.169	3.121	0.139	2.857
9	–	+	–	–	–	0.133	4.257	0.116	1.336
10	–	+	–	–	+	0.120	23.552	0.118	1.170
11	–	+	–	+	–	0.132	4.131	0.108	0.830
12	–	+	–	+	+	0.134	27.506	0.113	1.124
13	–	+	+	–	–	0.185	2.992	0.121	2.473
14	–	+	+	–	+	0.163	3.056	0.112	2.223
15	–	+	+	+	–	0.206	3.152	0.127	2.662
16	–	+	+	+	+	0.136	3.373	0.120	1.841
17	+	–	–	–	–	0.336	11.871	0.185	4.062
18	+	–	–	–	+	0.273	14.873	0.147	1.421
19	+	–	–	+	–	0.360	10.997	0.247	4.760
20	+	–	–	+	+	0.265	17.241	0.166	1.451
21	+	–	+	–	–	0.279	7.402	0.151	3.458
22	+	–	+	–	+	0.293	8.605	0.182	5.824
23	+	–	+	+	–	0.305	7.537	0.159	4.149
24	+	–	+	+	+	0.320	8.408	0.154	4.020
25	+	+	–	–	–	0.281	10.905	0.177	5.236
26	+	+	–	–	+	0.287	12.144	0.164	1.380
27	+	+	–	+	–	0.313	11.118	0.195	4.722
28	+	+	–	+	+	0.256	18.553	0.169	1.382
29	+	+	+	–	–	0.272	7.460	0.157	3.746
30	+	+	+	–	+	0.302	7.957	0.154	4.772
31	+	+	+	+	–	0.316	7.488	0.165	3.327
32	+	+	+	+	+	0.307	7.718	0.157	5.582

Table 6. Experimental matrix.

$E_{fi}Q_j$, $i=1..5$; $j=1..4$ Similarly, the same methodology was employed to estimate all others effects of factors.

We use the analysis of variance (ANOVA) to know the influence of factors in the Db performance. According to the ANOVA (Table 8) it appears that the effect of factor A is statistically significant ($p<.05$) for queries Q10, Q18 and Q22, and marginally significant ($p<.01$) for query Q13. It also stands out that the factors C and E are marginally significant for query Q10 and statistically significant for query Q13. However, such factors do not seems to show influence for query Q18. On the other hand, for query Q22 the factor C is statistically significant, while the factor E is marginally significant.

Factors	Q10	Q13	Q18	Q22
A (*shared_buffers*)	0.144	2.049	0.054	2.024
B (*temp_buffers*)	0.008	0.184	0.003	0.088
C (*work_mem*)	0.026	8.168	0.006	1.277
D (*wall_buffers*)	0.011	0.850	0.007	0.064
E (*effective_cache_size*)	0.023	6.141	0.008	0.483

Table 7. Effects of factors.

After estimate the effects of each factor and analyze them through the ANOVA, we used a analysis of sensitivity of factors (Table 9), suggested by [5], whose goal is create a rank of the factors. This methodology consists of a *"sorting method, where the effects are normalized with respect to the maximum effect, rounded to the first decimal point, and sorted in descending order"* [5].

For example, the sensitivity effect $E_AQ10 = \left| \frac{1}{N}\sum Y_+ - \frac{1}{N}\sum Y_- = \frac{4.764}{16} - \frac{2.460}{16} = 0.298 - 0.154 = 0.144 \right|$ and $E_BQ_{10} = \left| \frac{3.544}{16} - \frac{3.679}{16} = 0.008 \right|$. of factors A and B for query Q10 were estimated as $\left(S_{f_i}Q_j = E_{fi}Q_j / MAX(E_{f_i}Q_j),\ i=1..5;\ j=1..4 \right)$ and $S_AQ_{10}=0.144/0.144=1.0$. All others sensitivity effects were estimated in the same way (Table 8).

Once the sensitivity effect of factors was estimated, they can be rated with respect to its range of influence (Figure 3) based on number of factors studied (i.e.: 5, Table 5). According to this range, each factor has 0.2 units of influence and, therefore such factors with the same normalized effect can be assigned at the same rank. For example, factors with sensitivity effect 0.2 (factor E for query Q10) and 0.3 (factor A for query A13) will be at the same ranking.

The ranking of sensitivity effect of factors is presented in Table 10. These results corroborates with the ANOVA analysis (Table 8) and reveals that, factors statistically significant (i.e.: factor

		(a) Q10						**(b) Q13**			
Factors	**SS**	***df***	**MS**	**F**	***p***	**Factors**	**SS**	***df***	**MS**	**F**	***p***
A	0.498	1	0.498	188.789	0.000	A	100.717	1	100.717	4.550	0.036
B	0.002	1	0.002	0.653	0.421	B	0.814	1	0.814	0.037	0.848
C	0.016	1	0.016	6.100	0.015	C	1601.020	1	1601.020	72.331	0.000
D	0.003	1	0.003	1.058	0.306	D	17.343	1	17.343	0.784	0.378
E	0.013	1	0.013	4.821	0.031	E	905.090	1	905.090	40.890	0.000
Error	0.237	90	0.003			Error	1992.105	90	22.135		
Total SS	0.768	95				Total SS	4617.089	95			
		(c) Q18						**(d) Q22**			
Factors	**SS**	***df***	**MS**	**F**	***p***	**Factors**	**SS**	***df***	**MS**	**F**	***p***
A	0.069	1	0.069	89.169	0.000	A	98.355	1	98.355	65.706	0.000
B	0.000	1	0.000	0.304	0.583	B	0.188	1	0.188	0.126	0.724
C	0.001	1	0.001	1.192	0.278	C	39.146	1	39.146	26.152	0.000
D	0.001	1	0.001	1.733	0.191	D	0.099	1	0.099	0.066	0.797
E	0.001	1	0.001	1.924	0.169	E	5.592	1	5.592	3.736	0.056
Error	0.070	90	0.001			Error	134.720	90	1.497		
Total SS	0.143	95				Total SS	278.100	95			

Table 8. ANOVA table.

Factors	**Q10**	**Q13**	**Q18**	**Q22**
A	1.0	0.3	1.0	1.0
B	0.1	0.0	0.1	0.0
C	0.2	1.0	0.1	0.6
D	0.1	0.1	0.1	0.0
E	0.2	0.8	0.1	0.2

Table 9. Sensitivity effect of factors.

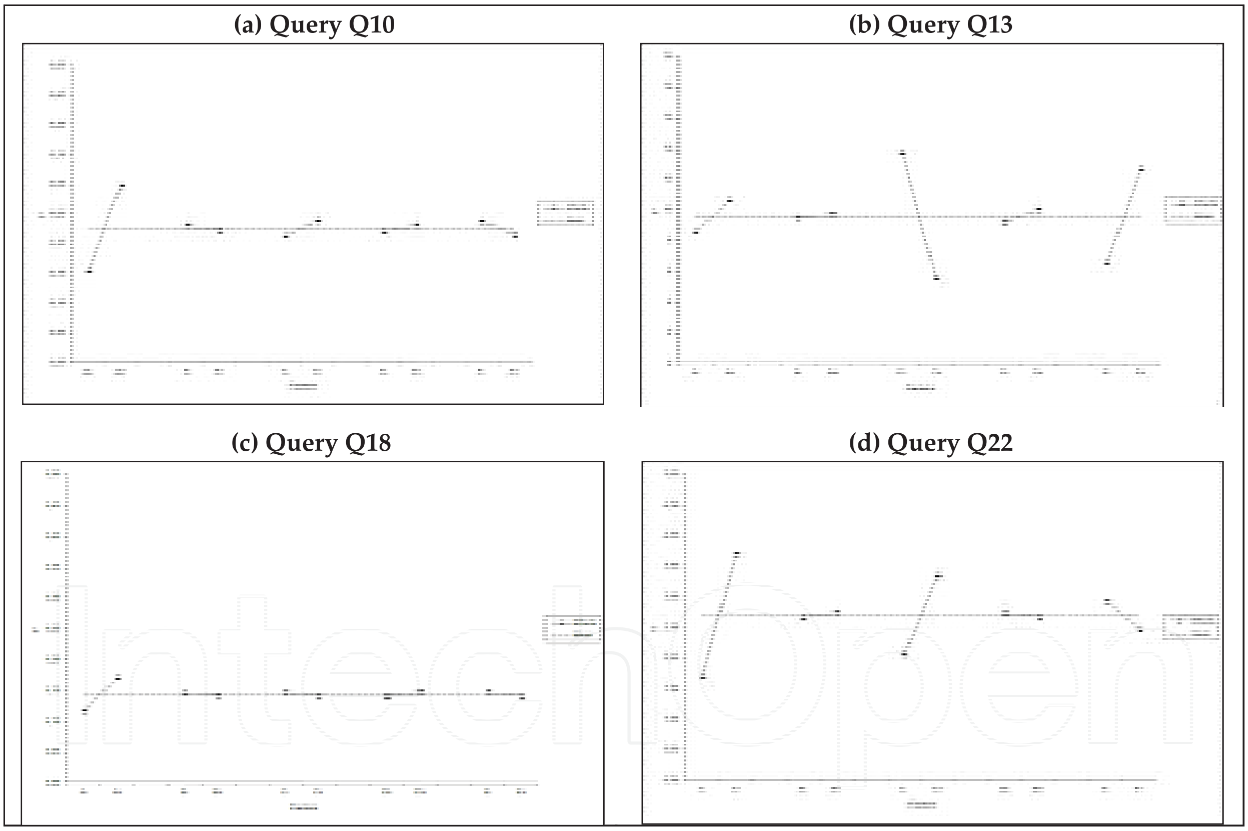
0.0 |– 0.2 |– 0.4 |– 0.6 |– 0.8 |– 1.0

Figure 3. Range of influence of factors.

Factors	Q10	Q13	Q18	Q22
A	1	4	1	1
B	5	5	5	5
C	4	1	5	3
D	5	5	5	5
E	4	2	5	4

Table 10. Sensitivity effect of factors.

A for queries Q10, Q18 and Q22) are the most sensitivity (close to 1.0), while those effects marginally significant (factors C and E for query Q10) have low influence (close to 0.0). Here also these factors do not looks have influence for query Q18.



(a) Query Q10

(b) Query Q13

(c) Query Q18

(d) Query Q22

Figure 4. Effect of factors.

Such observations can be highlighted by graphs of the changes of effects versus factors levels (Figure 4). They confirms the hypothesis that the factor A is the most significant for queries Q10, Q18 and Q22. Through a visual inspection, it should be noted that the factors classified with low influence are very close to the average (i.e.: factors B, C, D and E for query Q18). The graphs also confirms that factors C and E are significant for queries Q13 and Q22.

4.2. Phase II – 2^{k-p} fractional factorial design

The second phase of this study case uses a $S_B Q_{10}$=0.008/0.144=0.1 *fractional factorial* (16 experimental plays). Here, it is employed the concept of design resolution, such as the study overall k = 5 factors, however, p = 1 of those factors were generated from the interactions of a full $2^{[(5-1)=4]}$ factorial design. As result, the design does not give full resolution, that is, there are certain interaction effects that are confounded with other effects (i.e.: factor E, generated as result of one-way interactions between factors A, B, D and D).

As mentioned before, the experiments are composed of two replicates performed at the same computing environment used during the Phase I (Section 4.1). A sample of the experimental matrix with execution time (in seconds) – average time of queries answers – is presented in Table 11.

Exp.	Factors					Execution time			
	A	B	C	D	E	Q10	Q13	Q18	Q22
1	–	–	–	–	+	0.126	21.710	0.109	1.067
2	–	–	–	+	–	0.137	4.198	0.113	1.029
3	–	–	+	–	–	0.185	3.033	0.117	2.553
4	–	–	+	+	+	0.169	3.121	0.139	2.857
5	–	+	–	–	–	0.133	4.257	0.116	1.336
6	–	+	–	+	+	0.134	27.506	0.113	1.124
7	–	+	+	–	+	0.163	3.056	0.112	2.223
8	–	+	+	+	–	0.206	3.152	0.127	2.662
9	+	–	–	–	–	0.336	11.871	0.185	4.062
10	+	–	–	+	+	0.265	17.241	0.166	1.451
11	+	–	+	–	+	0.293	8.605	0.182	5.824
12	+	–	+	+	–	0.305	7.537	0.159	4.149
13	+	+	–	–	+	0.287	12.144	0.164	1.380
14	+	+	–	+	–	0.313	11.118	0.195	4.722
15	+	+	+	–	–	0.272	7.460	0.157	3.746
16	+	+	+	+	+	0.307	7.718	0.157	5.582

Table 11. Experimental matrix.

In Table 12 are presented the effects of factors for each query (Q10, Q13, Q18 and Q22). These effects were calculated with the same methodology used in the Phase I (Section 4.1), but here the sum of multiplying levels (– and +) with execution time (Y) across all 16 rows.

Factors	Q10	Q13	Q18	Q22
A (*shared_buffers*)	0.140	1.708	0.052	2.008
B (*temp_buffers*)	0.000	0.113	0.004	0.027
C (*work_mem*)	0.021	8.295	0.001	1.678
D (*wall_buffers*)	0.005	1.182	0.003	0.173
E (*effective_cache_size*)	0.018	6.059	0.003	0.344

Table 12. Effects of factors.

The influence of factors were studied with ANOVA (Table 13). It's noteworthy that factor A is statistically significant for queries Q10, Q18 and Q22, but it does not seems influential for query Q13. By the other hand, the factors C and E are statistically significant for query Q13. We also note that, with the fractional factorial experiments, there are no factors marginally significant for query Q10, so this query, as well as query Q18 have only one factor significant (factor A), while for query Q22 the factors A and C are statistically significant.

		(a) Q10						**(b) Q13**			
Factors	**SS**	***df***	**MS**	**F**	***p***	**Factors**	**SS**	***df***	**MS**	**F**	***p***
A	0.237	1	0.237	91.951	0.000	A	34.996	1	34.996	1.283	0.264
B	0.000	1	0.000	0.000	1.000	B	0.154	1	0.154	0.006	0.941
C	0.005	1	0.005	2.016	0.163	C	825.760	1	825.760	30.268	0.000
D	0.000	1	0.000	0.127	0.723	D	16.758	1	16.758	0.614	0.438
E	0.004	1	0.004	1.527	0.223	E	440.609	1	440.609	16.150	0.000
Error	0.108	42	0.003			Error	1145.839	42	27.282		
Total SS	0.354	47				Total SS	2464.116	47			
		(c) Q18						**(d) Q22**			
Factors	**SS**	***df***	**MS**	**F**	***P***	**Factors**	**SS**	***df***	**MS**	**F**	***p***
A	0.033	1	0.033	42.577	0.000	A	48.394	1	48.394	31.541	0.000
B	0.000	1	0.000	0.195	0.661	B	0.009	1	0.009	0.006	0.939
C	0.000	1	0.000	0.035	0.853	C	33.790	1	33.790	22.022	0.000
D	0.000	1	0.000	0.170	0.682	D	0.361	1	0.361	0.235	0.630
E	0.000	1	0.000	0.183	0.671	E	1.417	1	1.417	0.924	0.342
Error	0.033	42	0.001			Error	64,.442	42	1.534		
Total SS	0.066	47				Total SS	148.413	47			

Table 13. ANOVA table.

After known the ANOVA results, we also employee the analysis of sensitivity of factors to rank them (Table 14) according to the range of influence (Figure 3). The results reveals that such factors classified as statistically significant by ANOVA (Table 13) (i.e.: factor A for queries Q10, Q18 and Q22) are the most sensitivity. It stands out that A is the only factor that seems influential for queries Q10 and Q18.

Factors	Q10	Q13	Q18	Q22
A	1	4	1	1
B	5	5	5	5
C	5	1	5	2
D	5	5	5	5
E	5	3	5	4

Table 14. Sensitivity effect of factors.

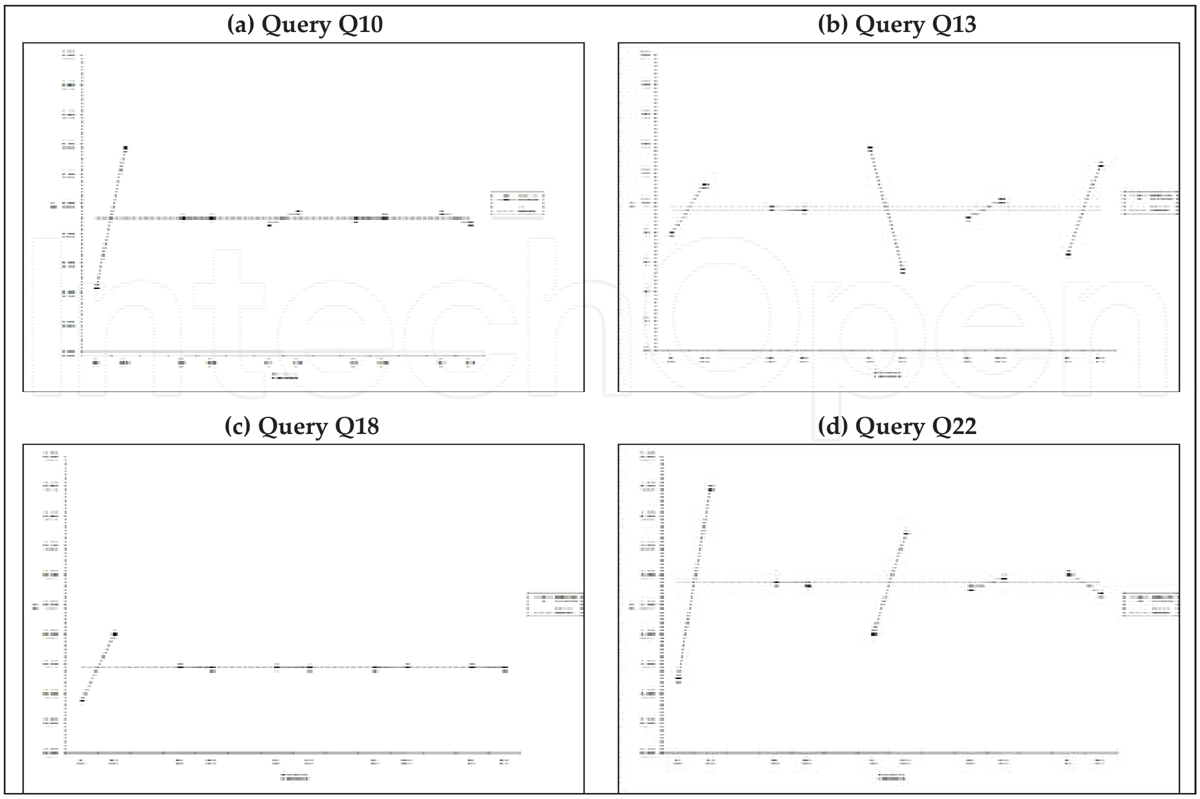
Through the graphs of the changes of effects versus factors levels (Figure 5) it can be noted that the factor A is the most significant for queries Q10, Q18 and Q22. The a visual inspection stands out the factors classified with low influence are very close to the average (i.e.: factors B, C, D and E for queries Q10 and Q18). The graphs also highlights that factors C and E are significant for queries Q13 and Q22.

4.3. Analysis of results

Through the present study were made several experiments using two DoE techniques (2^k full factorial design and 2^{k-p} fractional factorial design) to investigates how different Db parameters can influence in its performance.

The Phase I (Section 4.1) comprehends a study with a 2^k full factorial design (k = 5), whose results highlighted the influence of the factors and rated them in concordance with its sensitivity. According to the ANOVA (Table 8) there are factors statistically significant for one query, but marginally significant for others. So, we employed the analysis of sensitivity (Table 10), that corroborated with the ANOVA results. Thus, according to the queries characteristics used in this case study, the results suggests the factor A as the most significant, followed by factors C and E rated as very significant and significant, respectively, while the others (factors B and D) looks have low influence in the Db performance.

In the Phase II (Section 4.2) a same research was conducted, but using a 2^{k-p} fractional factorial design (k = 5, p = 1). With the fractional design we come to the results with half of the work required by full design and, through them, we also know the influence of each factor (Table 14). The results were similar to those succeeded before (Section 4.1) and rated the factor A as the most significant, followed by factors C and E as very significant and significant, respectively, while the others (factors B and D) with low influence in the Db performance.



(a) Query Q10

(b) Query Q13

(c) Query Q18

(d) Query Q22

Figure 5. Effect of factors.

To perform comparison of results between both techniques, it's noteworthy that even with similar results, full design is more accurate. For example, it stands out such factors classified as marginally significant by ANOVA (Table 8) from full design (i.e.: C and E for query Q10), do not appear in fractional design. However, despite the accuracy, the full design is more laborious and, therefore should require more resources (depending on the number of factors). Anyway, we could state that, in this study, both techniques proved to be effective for identification and classification of influential factors (parameters) to the Db performance.

In this study, we assume that all queries have same importance. So intuitively it seems that factor A (*shared_buffers*, amount of memory reserved for data cache) is one of the most significant to improve the Db performance, as it appears as rated first for three queries out of five with both techniques chosen. The factor C (*work_mem*, amount of memory used for sort operations and hash tables) also seems very significant, as it was rated as first and second for two queries (Q13 and Q22, respectively), while the factor E (*effective_cache_size*, amount of memory available for disk cache used by the operating system and Db) seems marginally significant. Since their rate can vary according to the query. Another interesting observation is that factors B and D (*temp_buffers*, related to the maximum number of temporary buffers used by each Db session, and *wall_buffers*, useful for systems with high need to write to disk, respectively) never seems important for the individual queries. Therefore, the results suggests that these two parameters should have low impact to improve the Db performance. All results are summarized in Table 15.

Influence	Parameter
High	shared_buffers work_mem
Medium	*effective_cache_size*
Low	temp_buffers wall_buffers

Table 15. Parameters influence.

5. Related work

A quick search on the contemporary literature reveals some works addressing to the use of DoE in the several scopes. At the computer science area, the use of DoE is explored by [12] through a comprehensive study about techniques for software engineering experimentation. Other works [13; 14] are devoted to the algorithmic optimizations by means of DoE.

There are also a lot of works approaching the Db performance subject. For example, [15] approaches the optimizations of Db systems through a statement of a new problem, that is the Web-based interactive applications. [16] report a performance study with different Db architectures and provide useful information for the enterprise architects and database administrator in determining the appropriate Db architecture. Techniques to automate the setting of tuning parameters in specifics software applications could be found in [17], as well as in [18]. The importance of best practices and the database administrator knowledge for autonomic Db tuning is pointed by [19]. In [20] is introduced a algorithm to select a small subset of Db statistics, such that it can improve the benefits over maintaining base-table statistics. To [21] the challenge in making Db systems truly self-tuning is a tall task, due the different abstractions for workloads and different constraints on the desired solution (i.e.: the complexity of internal components of the Db architecture). In [22] is discussed a way to avoid the trial and error process of SQL tuning, through by choosing a small set of experiments in order to reach a satisfactory plan to execute queries.

The o use of DoE techniques is formally explored in the Db scope. In the [5; 6], the database performance evaluation was studied by a statistical approached. The authors define a statistical methodology, which may be useful to minimize the effort related with database tuning activities. Following in this line, the study presented by [7] describes a software application, whose purpose is to automate the database parameters configuration by means of DoE.

In summary, in the Db performance area there are much of the effort to take the tests and results comparison, however only a little portion of the studies uses the DoE methodology to planning and analysis of the experiments.

6. Final considerations

This chapter presented a study to investigate the influence of Db parameters in its performance. Two DoE techniques (full factorial design and fractional factorial design) were applied in a case study with PostgreSQL DBMS and TPC-H benchmark, to assist in the investigations of how each parameter may influence in the Db performance.

To the case study, were selected five parameters mostly related with the Db tuning and conducted different experiments with the DoE techniques chosen. At the Phase I, we studied the influence of parameters through a 2^k full factorial design, whose the analysis suggests, with high degree of confidence, the parameter shared_buffer as the most significant, while work_mem and effective_cache_size can be classified as very significant and significant, respectively. According to the analysis, the others parameters (temp_buffers and wall_buffers) looks have low influence in the Db performance. The Phase II comprehended a similar study, but using 2^{k-p} fraction factorial design. The results were very similar with those suggested in Phase I, that is the shared_buffer, work_mem and effective_cache_size looks have influence in the Db performance, while the others not. It stands out in Phase II, that those parameters marginally significant with full design, do not appear in fractional design. This characteristics leading us to conclude that, although being the most laborious, full design is more accurate. But, by the other hand, according to the analysis of case study results, it is also feasible to reach the same conclusions with fractional design using half of the work required by the full design.

It should be noted that Db technology was used in this study as a vehicle to demonstrate how the DoE methodology can help in the design of experiments and its analysis and used as tool in several scopes, like in the computing science field. We also emphasizes that this study did not aim to close the subject about the use of DoE in the computing scope, instead it we sought disclose the effectiveness of this methodology applied in this context. Thus, we can conclude that DoE methodology is a promising to assist in quantitative analysis, for example in the investigation of influential parameters in the Db performance.

Author details

Eduardo Batista de Moraes Barbosa[1*] and Messias Borges Silva[1,2]

*Address all correspondence to: ebmb@yahoo.com

1 São Paulo State University “Júlio de Mesquita Filho” – UNESP, School of Engineering at Guaratinguetá – FEG, Guaratinguetá, SP, Brazil

2 University of São Paulo – USP, School of Engineering at Lorena – EEL, Estrada Municipal do Campinho, s/n – Lorena, SP, Brazil

References

[1] Gray, J. (Ed.) The Benchmark Handbook for Database and Transaction Systems. 2nd. Edition. Morgan Kaufmann, 1993.

[2] Leutenegger, S. T.; Dias, D. A Modeling Study of the TPC-C Benchmark. ACM SIGMOD Record, Vol. 22, Issue 2, p. 22-31, 1993.

[3] Poess, M., Floyd, C. New TPC Benchmarks for Decision Support and Web Commerce. ACM SIGMOD Vol. 29 Issue 4, p. 64-71, 2000.

[4] Hsu, W. W.; Smith, A. J.; Young. H. C. I/O Reference Behavior of Production Database Workloads and the TPC Benchmarks – An Analysis at the Logical Level. ACM Transactions on Database Systems (TODS), Vol. 26, Issue 1, p. 96-143, 2001

[5] Debnath, B. K.; Lilja, D. J.; Mokbel, M. F. Sard: A Statistical Approach for Ranking Database Tuning Parameters. In: IEEE 24th International Conference on Data Engineering Workshop (ICDEW), pp. 11-18, 2008.

[6] Debnath, B. K.; Mokbel, M. F.; Lilja, D. J. Exploiting the Impact of Database System Configuration Parameters: A Design of Experiments Approach. In: IEEE Data Eng. Bull., pp. 3-10, 2008.

[7] Duan, S.; Thummala, V.; Babu, S. Tuning Database Configuration Parameters with iTuned. In: Proceedings of Very Large Database Endowment (PVLDB), pp. 1246-125, 2009.

[8] TPC Benchmark™ H (Decision Support) Standard Specification Revision 2.8.0. URL: http:// www.tpc.org (Accessed: Dec., 2012).

[9] NIST/SEMATECH. e-Handbook of Statistical Methods. 2006. URL: http:// www.itl.nist.gov/ div898/handbook. (Accessed: Dec., 2012)

[10] Montgomery, D. C. Design and Analysis of Experiments (5th Edition). John Wiley & Sons Inc., 2001.

[11] Box, G. E. P.; Hunter, W. G.; Hunter, J. S. Statistics for Experimenters (1 Ed.). John Wiley & Sons, 1978.

[12] Juristo, N.; Moreno, A. M. Basics of Software Engineering Experimentation. Springer, 2001.

[13] Adeson-Diaz, B.; Laguna, M. Fine-Tuning of Algorithms Using Fractional Experimental Designs and Local Search. Operations Research, Vol. 54, No. 1, p. 99-114, 2006.

[14] Arin, A.; Rabadi, G.; UNAL, R. Comparative studies on design of experiments for tuning parameters in a genetic algorithm for a scheduling problem. Int. Journal of Experimental Design and Process Optimisation, Vol. 2, N. 2, p. 103-124, 2011.

[15] Florescu, D., Kossmann, D. Rethinking Cost and Performance of Database Systems. ACM SIGMOD Record, Vol. 38, No. 1, p. 43–48, 2009.

[16] Chen, S., Ng, A., Greenfield. P. Concurrency and Computation: Practice and Experience. Concurrency Computat.: Pract. Exper. doi: 10.1002/cpe.2891. 2012.

[17] Stillger, M.; Lohman, G.; Markl, V.; Kandil, M. Leo – DB2's Learning Optimizer. In: Proceedings of Very Large Database Endowment (PVLDB), p. 19-28, 2001.

[18] Badu, S. Towards Automatic Optimization of MapReduce Programs. In: Proceedings of the 1st ACM symposium on Cloud computing (SoCC '10), p. 137-142, 2010.

[19] Wiese, D.; Rabinovitch, G; Reichert, M; Arenswald, S. Autonomic Tuning Expert – A framework for best-practice oriented autonomic database tuning. In: Proceedings of the 2008 conference of the center for advanced studies on collaborative research: meeting of minds (CASCON '08), p. 3:27-3:41, 2008.

[20] Bruno, N., Chaudhuri, S. Exploiting Statistics on Query Expressions for Optimization. In: Proceedings of the 2002 ACM SIGMOD international conference on Management of data, p. 263-274, 2002.

[21] Chaudhuri, S., Narasaaya, V. Self-Tuning Database Systems: A Decade of Progress. In: Proceedings of the 33rd international conference on Very large data bases, p. 3-14, 2007.

[22] Herodotou, H.; Badu, S. Automated SQL Tuning through Trial and (sometimes) Error. In: Proceedings of the Second International Workshop on Testing Database Systems (DBTest '09), article No. 12, 2009.