IX INTERNATIONAL CONFERENCE FOR YOUNG RESEARCHERS

TECHNICAL SCIENCES . INDUSTRIAL MANAGEMENT

PROCEEDINGS

Scientific-Technical Union of Mechanical Engineering
108 Rakovski str., 1000 Sofia, BULGARIA
tel./fax (+359 2) 986 22 40, tel. (+359 2) 987 72 90, Mobile: +359 888003582
office@youngconference.com, www.youngconference.com/

S C I E N T I F I C
P R O C E E D I N G S

OF THE SCIENTIFIC-TECHNICAL UNION
OF MECHANICAL ENGINEERING

Year XXIII **Volume 18/181** **SEPTEMBER 2015**

IX INTERNATIONAL CONFERENCE FOR YOUNG RESEARCHERS

TECHNICAL SCIENCES AND INDUSTRIAL MANAGEMENT

07 – 10.09.2015 BURGAS, BULGARIA

ISSN 1310-3946

C O N T E N T S

PERFORMANCE BENCHMARK OF PHP FRAMEWORKS WITH DATABASE SELECT METHODS

Mr. Student Eng. Janev M., Prof. PhD Delipetrev B., Eng. Ristova S.
Faculty of Computer Science, University „Goce Delchev“ - Shtip, Macedonia
mile.janev89@yahoo.com, blagoj.delipetrev@ugd.edu.mk, suzanaristova@yahoo.com

Abstract: *PHP is one of the most popular programming languages for web applications backend development. In order to quickly develop an application, many companies have created their own frameworks. Today, there are hundreds of frameworks in PHP, but all are not equally popular. Only a few of them stand on top of the most popular frameworks (Zend, Codeigniter, Yii, CakePHP, Symfony, Laravel). Each of these frameworks has its own advantage over others, some are faster in database operations, other are better in security, or allow rapid application development. The choice of which framework will be used for development depends from application needs. One of the most important aspects when we are choosing framework is speed, no one wants to wait long for loading some web page. Even Google have clearly stated that all those sites which needs more than 2 seconds to load will be rated negative. In this article we elaborated three of the most used PHP frameworks (Yii, Zend and Codeigniter) and we test them how quickly will retrieve data from database.*

Keywords: PHP FRAMEWORK, BENCHMARK, YII FRAMEWORK, CODEIGNITER, ZEND

1. Introduction

PHP [8] is used in almost 82% of all internet web sites as a backend programming language. PHP frameworks are used in 40% of all web sites [1]. There are many frameworks in PHP [16], some created for companies needs, others for general use. But all of them have the same goal: to speed up the process of application development and to increase the security level.

With increasing security mechanisms the speed of application is reduced, and this directly affects the interest of visitors. Nobody wants to wait longer than 2-3 seconds to read some information.

This is one of the reasons why is necessary to make an appropriate balance between all of these factors. Mostly used frameworks [11] [15] succeed to achieve this balance. Here we will point out three of them which are more and more used. These are Zend [2] [7], Yii [4] [5] and Codeigniter [3] [6].

On the Internet you can find more speed benchmarks [13] [14] [17] [18], but much less common are those who make analyzing speed with database.

Almost all web applications use database, which means it is very important how quickly some framework can communicate with the database. Every framework has their own classes and methods for extracting data from database. These classes are also called Active Record classes. They have embedded security mechanisms (filters) and some other methods, thereby reduced the speed of data extraction.

To develop this benchmark we have created application where we have tested every method from all frameworks for extracting data on same database. Application can be found on the following link http:/www.SiMYan.info/performance-benchmark.

The project has clearly defined two objectives:

- The first is to make internal comparison of all methods in given framework and to determine the fastest method for selecting data from database.
- The other is to determine which of these frameworks is the fastest.

2. Test Environment

All tests are made on local machine on LAMP server, Linux Ubuntu operating system. Hardware and software specifications are the following:

- Processor: Intel Core i5-2430M (2 Cores, 4 Threads, 2.4GHz)
- RAM memory: 6GB
- Operating system: Linux Ubuntu 12.04 LTS, 32-bit
- Apache server, version 2.2.22
- MySQL database, version 5.6.21
- PHP version 5.3.10

Framework versions are:

- Yii version 1.1.15
- Zend version 1.11.11
- Codeigniter version 2.2.0

This is one of the reasons why is necessary to make an appropriate balance between all of these factors. Mostly used frameworks [11] [15] succeed to achieve this balance. Here we will point out three of them which are more and more used. These are Zend [2], Yii[4] and Codeigniter[3].

Tables are with following sizes:

- s_post – 10 000 records
- m_post – 100 000 records
- l_post – 300 000 records

All tests were done with disable caches on server. Operation environment is completely same in everyone.

As auxiliary tools are used Xdebug [9] for errors detecting, NetBeans editor [12], GitHub [10]. Code can be downloaded from the following link https://github.com/mile-janev/frameworks.git.

Each method has run 900 seconds or 300 seconds on each table. For example, we tested how many times method findAll() in Yii will be performed in 300 seconds on each table.

All information displayed further will be expressed in number of executions of the method in one minute.

3. Methods Description

Every method in each framework has its own purpose why is there, although sometimes more methods may be used for the same purpose. Some methods are faster, some easier to create dynamic queries. Programmer should always choose the right method according to the needs. You should always choose the method that will be able to complete the required task but in same time will be the fastest one.

All frameworks have user guides on his web pages. User guide for Yii can be found on http://www.yiiframework.com/doc/guide/, for Zend on http://framework.zend.com/manual/1.12/en/reference.html, while for Codeigniter the link is http://www.codeigniter.com/user_guide/.

In this chapter we will briefly explain every function for selecting database record in each framework individually, what parameters they receive and what is returned as result.

3.1 Yii

Yii has eight methods for extracting data from the database. Four of them select one entry from the base, and four select all records that will meet the specified requirement.

Methods containing abbreviation "All" in its name return an array of objects that satisfy the given condition, while those that do not contain return one row from database mapped into object.

The methods are:

- *find()* – Accept an object from class CDbCriteria as parameter.
- *findByPk()* – Accept an integer (database primary key for the record) as parameter.
- *findByAttributes()* – Accept an array in format "key"=>"value" as parameter, where "key" is column name and "value" is the given condition.
- *findBySql()* – Accept one or two parameters. The first is manually defined query in which all conditions should be replaced with some strings, while the second one is an array where on each string will be given some value. For example, *findBySql("SELECT * FROM s_post WHERE id = :id", array(":id"=>5).*
- *findAll()* – Accept one parameter or not receive parameters. If parameter is not given all records will be returned. If you like to add some condition (parameters) should be an object from class CDbCriteria.
- *findAllByPk()* – Accept an array of integers (primary keys for records in database) as parameter.
- *findAllByAttributes()* – Accept an array in format "key"=>"value", where "key" is the column name from database, and "value" is the given condition**.**
- *findAllBySql()* – Accept one or two parameters. The first is manually defined query in which all conditions should be replaced with some strings, while the second one is an array where on each string will be given some value. For example, *findBySql("SELECT * FROM s_post WHERE id = :id", array(":id"=>5).*

3.2 Zend

Zend have four ways of extracting data from database. Two of them (find and fetchRow) return one result, while two (fetchAll and Zend_Db_Select) return all records that satisfy the given condition.

These methods are:

- *find()* – Accept an integer (primary key) or array of integers as parameter. Returns an object from class Zend_Db_Table_Rowset_Abstract.
- *fetchRow()* – Accept a string as query condition (example: "id=".$id). Returns an object from class Zend_Db_Table_Row.
- *fetchAll()* – Accept a string as query condition (example: "id=".$id). Returns an object from class Zend_Db_Table_Rowset_Abstract.
- *Zend_Db_Select* – Represents a SQL SELECT condition in query. Class has methods for partial adding parameters to query.

3.3 Codeigniter

In Codeigniter all methods return an object from class CI_DB_mysql_result.

- *get_where()* – Accept four parameters. First is table name (string), the second is array in format "key"=>"value", where "key" is column name and "value" is our condition. Third and fourth parameters are limit and offset, and they are not mandatory.
- *get()* – Partial query.
- *query()* – Manually defined query in string format ready for direct execution in the database.

4. Results and Discussion

In testing process all server caches were completely disabled.

Finally, complete content and organizational editing before formatting. Please take note of the following items when proofreading spelling and grammar:

4.1 Yii

The diagram on Figure 1 shows all methods in Yii with number of executed operations in one minute, grouped by table size.

From the diagram we can note that the fastest method is findBySql() and findAllBySql(), while the slowest is findAll(). With increasing the content in tables positions of methods are not changed, findBySql() remains on top and findAll is slowest.

The advantage of findAll() and higher usage than findBySql() and other methods is that he receives an object from class CDbCriteria as parameter, who allows creating partial queries, while findBySql() and findAllBySql() accepts only manually created query in string format.

On figure 2 is shown a diagram where are represented only methods who return one record. On this diagram grouping by columns is made by one method on different table size, and it is easier to distinguish the execution speed on some method when the number of records in table size is increased.

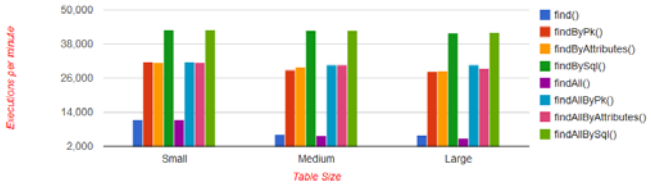

Fig. 1 *Number of queries performed in one minute on small, medium and large database table in Yii*

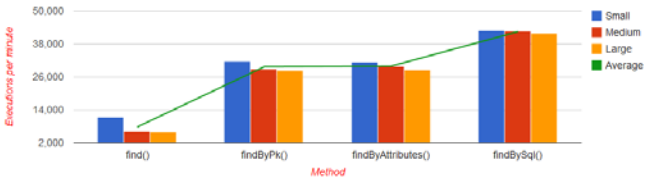

Fig. 2 *Number of queries performed in one minute grouped by same method on different table, only with methods who return one record in Yii*

The biggest difference in deceleration when number of records in table are increased is in find() method. For table with 10 000 records method is performed 11,377 times per minute, while for table with 100,000 records 6,265 times or almost in half. But with further increasing records in table we can notice that speed is stabilized again, and for 300,000 records we have 5,873 executions.

The situation is similar in other methods too, which means that bigger difference is between small and medium table than medium and large. Most stable method is findBySql() where is noted slightest difference when records in table will be increased.

We must underline that in our case the difference between the smallest and the largest table is around 300,000 records so the difference is not very large, but if it would be a few million, then the difference will be much higher.

The methods that return more records grouped by methods are presented in Figure 3. The results are similar to the methods for selecting one record. Method findAll() with 6,602 average

executions in one minute is the slowest one, and findAllBySql() with 42,674 is the fastest.

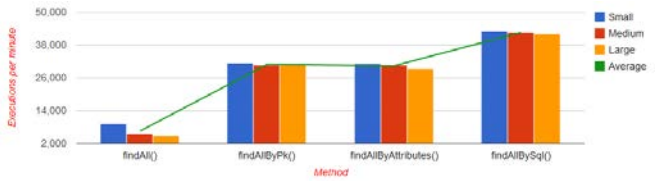

Fig. 3 *Number of queries performed in one minute grouped by same method on different table, only with methods who return multiple record in Yii*

4.2 Zend

Similar as in Yii, in Zend also will be represented two images with diagrams, where in the first diagram methods will be presented regarding to the execution speed on one table, and in the second will be grouped by method so will be easier to distinguish the execution speed on some method when the number of records in table size is increased.

In the first case, in Figure 4 we see that the fastest method is fetchAll(), while the slowest is selecting records with Zend_Db_Select. With increasing the number of records in table position of methods is not changed.

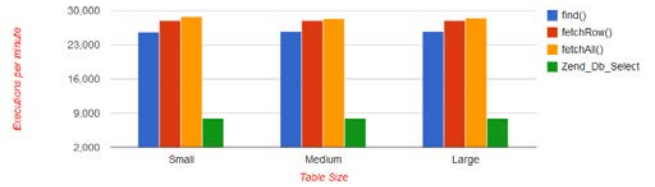

Fig. 4 *Number of queries performed in one minute on small, medium and large database table in Zend*

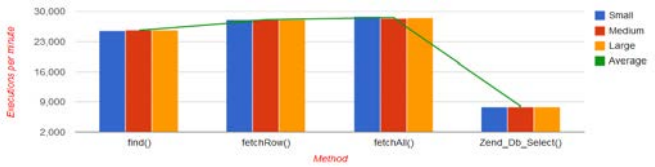

Fig. 5 *Number of queries performed in one minute grouped by same method on different table in Zend*

Figure 5 presents the second case, where we can see an interesting situation in which almost is no difference in the speed of method no matter the number on records has been increased. This means that Zend can great handle with large databases, and is suitable for large projects. We can note some cases when some methods are quickly performed in large table, but this difference is very small and insignificant, probably caused probably because processor had other processes in that time.

4.3 Codeigniter

In figure 6 we have presented all methods grouped regarding the execution speed on small, medium and large table. From the diagram we can easily conclude that the fastest method is query(), followed by get_where() and the slowest one is get().

From Figure 7 where we make grouping by method, some minimal difference in all methods can be noted between small and medium table, while the difference between medium and large table is minimal, which means that Codeigniter is equally suitable for small and large projects.

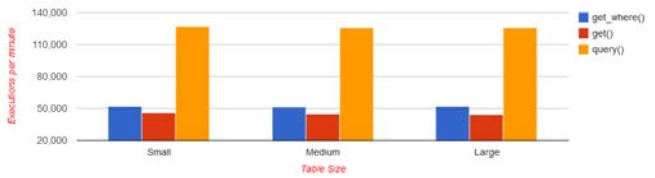

Fig. 6 *Number of queries performed in one minute on small, medium and large database table in Codeigniter*

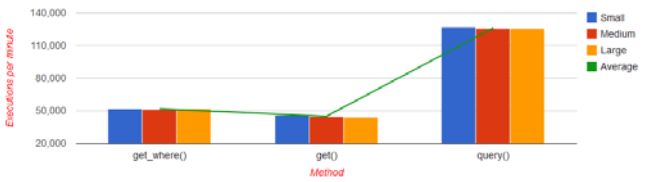

Fig. 7 *Number of queries performed in one minute grouped by same method on different table in Codeigniter*

4.4 All Frameworks Comparation

To make a general comparison between all frameworks, we make a group of all execution times in each framework and then we found average execution time. This is showed on Figure 8, where each bar represents a number of selection made in one minute on small, medium and large table, while the green line showing the average number of executions on all tables together.

From Figure 8 we can conclude that Codeigniter is the fastest framework, followed by Yii, and Zend is last.

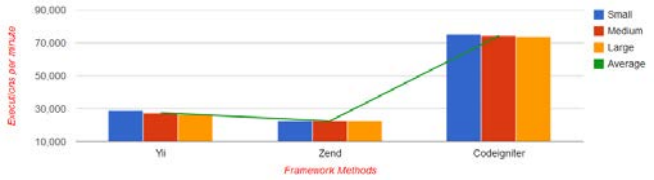

Fig. 8 *Comparation of all methods in Yii, Zend and Codeigniter*

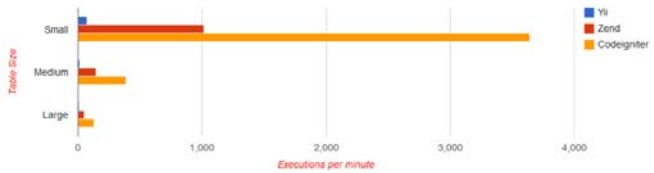

Fig. 9 *Selecting all records from table without conditions*

Another conclusion that we can bring is that while in Zend almost performance degradation cannot be noted when data in database is increased, in Yii and Codeigniter we can see slightly better performances in smaller tables.

The number of executions when all data from one table is selected is presented in Figure 9. From here we can note that in all frameworks has performance degradation when data is increased, but Codeigniter is fastest again, followed this time by Zend, and Yii is last.

Figure 10 shows a line diagram where is presented by one method from all frameworks which is used for selecting multiple records from database. Those methods are findAll() in Yii, fetchAll() in Zend and get() in Codeigniter, who are aslo the most used methods among developers.

Codeginiter with get() won the battle again, Zend with fetchAll() is second and findAll() in Yii is slowest.

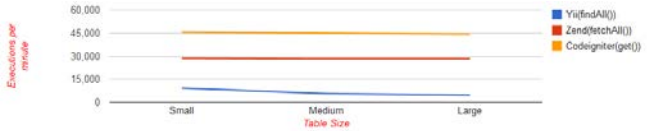

Fig. 10 *Selecting all records for given condition*

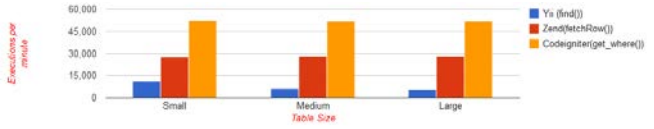

Fig. 11 *Selecting one record*

Figure 11 shows a diagram where is represented by one method from all frameworks which is used for selecting one record from database. Yii and Zend have methods that return exactly one record, while in Codeigniter is no such method but we took as his

representative get_where(), because is logical for this situations to be used this method. In Yii there are several methods that return one result, but we took find() as commonly used, while Zend representative is fetchRow().

The conclusion is that get_where() in Codeigniter is fastest, followed by fetchRow() in Zend and find() in Yii.

5. Conclusion

Driven by the need to increase the development speed and security in applications, some PHP communities and companies created their own frameworks. Qualities that are offered in each of them bring them to the top. In this text we have processed three of the most popular frameworks: Yii [4], Zend [2] and Codeigniter [3].

Considering the fact that every framework has safety filters, speed of selecting data is one of the main aspects that need to be taken, because now almost every application has database.

From our analyses and tests we can bring a few conclusions.

Best methods from the aspect of performances in Yii are findBySql() and findAllBySql(), followed by findByPk() and findAllByPk(), findByAttributes() and findAllByAttributes(), and at the end find() and findAll(). In Zend order is as follows: fetchAll(), fetchRow(), find(), Zend_Db_Select. In Codeigniter: query(), get_where(), get(). If you want to improve the performance in your application, it is recommended to follow the specified order and use those methods ahead of the list which will be able to finish the required task.

In Codeigniter average speed is raised by query() method which is not very useful, taking the fact that receive manually defined query as parameter. Unlike Codeigniter, in Yii and Zend are methods which are quite faster than the average and are very favorable to use, such as findByPk() or findByAttributes() in Yii and the increased use of these methods can accelerate application largely. However, the objective analysis shows that average execution of methods in Codeigniter is fastest, so we proclaim Codeigniter as the winner.

6. References

[1] Lancor, L., & Katha, S. (n.d.). *Analyzing PHP Frameworks for Use in a Project-Based*. Retrieved 10 10, 2014, from Grinell Campus: http://db.grinnell.edu/sigcse/sigcse2013/Program/viewAcceptedProposal.pdf?sessionType=paper&sessionNumber=116.

[2] *Zend Framework*. (n.d.). Retrieved 11 05, 2014, from Zend Framework: http://www.zend.com/

[3] *CodeIgjniter*. (n.d.). Retrieved 11 05, 2014, from CodeIgniter Web Framework: http://www.codeigniter.com/

[4] *Yii*. (n.d.). Retrieved 11 05, 2014, from Yii PHP Framework: http://www.yiiframework.com/

[5] *Class Reference*. (n.d.). Retrieved 10 28, 2014, from Yii PHP Framework: http://www.yiiframework.com/doc/api/

[6] *Codeigniter User Guide*. (n.d.). Retrieved 11 14, 2014, from Codeigniter: http://www.codeigniter.com/user_guide/

[7] *Zend Framework Reference*. (n.d.). Retrieved 11 09, 2014, from Zend Framework: http://framework.zend.com/manual/1.12/en/reference.html

[8] *PHP: Hypertext Preprocessor*. (n.d.). Retrieved 10 25, 2014, from PHP: http://www.php.net/

[9] *Xdebug – Debugger and Profiler Tool for PHP*. (n.d.). Retrieved 10 24, 2014, from Xdebug: http://xdebug.org/

[10] *GitHub – Build software better, together*. (n.d.). Retrieved 10 22, 2014, from GitHub: https://github.com/

[11] *Sitepoint - Best PHP Frameworks for 2014*. (n.d.). Retrieved 10 26, 2014, from Sitepoint: http://www.sitepoint.com/best-php-frameworks-2014/

[12] *NetBeans IDE* (n.d.). Retrieved 10 20, 2014, from Netbeans IDE: https://netbeans.org/downloads/index.html

[13] *System Architect – Performance benchmark of popular PHP frameworks* (n.d.). Retrieved 10 17, 2014, from System Architect: http://systemsarchitect.net/performance-benchmark-of-popular-php-frameworks/

[14] *Yii PHP Framework | Performance* (n.d.). Retrieved 10 18, 2014, from Yii PHP Framework: http://www.yiiframework.com/performance/

[15] *CodeGeekz* (n.d.). Retrieved 10 18, 2014, from CodeGeekz: http://codegeekz.com/20-best-php-frameworks-developers-august-2014/

[16] *PHP Frameworks* (n.d.). Retrieved 10 15, 2014, from PHP Frameworks: http://phpframeworks.com/

[17] *PHP Framework MVC Benchmark* (n.d.). Retrieved 11 25, 2014, from PHP Framework MVC Benchmark: http://www.ruilog.com/blog/view/b6f0e42cf705.html

[18] *PHP Framework Comparison* (n.d.). Retrieved 11 27, 2014, from PHP Framework Comparison: http://www.livingwithphp.com/php-framework-comparison-2014/