DATABASE (HTTPS://WWW.CLOUDWAYS.COM/BLOG/PHP/DATABASE/)
PHP (HTTPS://WWW.CLOUDWAYS.COM/BLOG/PHP/)

Exclusive MySQL Performance Tuning Tips For Better Database Optimization

Shahroze Nawaz (https://www.cloudways.com/blog/author/shahroze-nawaz/) -

March 19, 2019 ◦ 7 Min Read

mysql performance tuning

Structured Query Language (SQL) is a special-purpose programming language used to store, manipulate, and retrieve data from the database. It has found applications in many relational database systems including MySQL, Postgres, Oracle, SQL Server, and others.

By using SQL statements, developers easily carry out various functional database operations such as creating, updating, and deleting data.

As data volumes grow, and technology becomes increasingly complex, it is becoming more important to optimize MySQL databases properly to deliver end-user experience and to lower infrastructure costs. MySQL performance tuning tools can help database professionals quickly identify bottlenecks, target insufficient operations through a review of query execution plans, and eliminate any guessing games.

☐ Table of Content

f. The ORDER BY clause is mandatory in SQL if you expect to get a sorted result

3. Don’t Use MySQL as a Queue
4. Understand the Four Fundamental Resources
5. Pagination Queries
6. Optimizing MySQL Subqueries
7. MySQL Query Cache
8. Use Memcached for MySQL Caching
9. Wrapping up!

With the added complexity of growing data volumes and ever-changing workloads, database performance tuning and MySQL query optimization are now necessary to maximize resource utilization and system performance.

There are several reasons which make SQL tuning a bit complex for developers. Firstly, it requires extensive technical expertise to write and understand different execution plans. While writing clean and complete SQL statements is the responsibility of the one who garners thorough knowledge of it.

Besides its complexity, tuning is very time-consuming. Because when you have a large number of SQL statements to sort through, it brings a bit of uncertainty to find out which statements must you tune up and which one should you leave. And while every statement is different, their tuning approach also varies according to their respective functionalities.

In this tutorial, I will discuss how to improve MySQL performance using some handy performance tuning tips. So, let’s see them in detail below:

The Benefits of MySQL Performance Tuning

The major advantage of identifying the performance driving factor for database allows you to avoid over-provisioning and reducing cost by right-sizing your servers. It also gives you insights into whether moving data storage or adding server capacity will bring improvement in performance or not, and if so, then how much will it be.

The tuning database for MySQL query performance optimization doesn’t come with pale challenges. However, once tuned properly, the database gives worthwhile performance results with great functionalities. It not only lowers unwanted task load but also optimizes the MySQL database for faster data retrieval.

You Might Also Like: PHP Performance Tips to Optimize Your Websites (https://www.cloudways.com/blog/php-performance/)

Optimize Queries With MySQL Query Optimization Guidelines

Follow these best practices for your MySQL performance tuning and optimizing database speed.

First of all, ensure indexing of all the predicates in WHERE, JOIN, ORDER BY, and GROUP BY (https://dev.mysql.com/doc/refman/8.0/en/group-by-optimization.html) clauses. WebSphere Commerce strongly emphasizes on indexing of predicates to augment SQL performance. Because improper indexing of SQL queries can cause table scans, which eventually lead up to locking problems and other issues.

Therefore, I highly recommend indexing all predicate columns so that database can experience MySQL query optimization.

You Might Also Like: Laravel Performance Optimization Guide (https://www.cloudways.com/blog/laravel-performance-optimization/)

Avoid using functions in predicates

The database doesn't use an index if it has some function predefined in the column.

For example:

```
1.  SELECT * FROM TABLE1 WHERE UPPER(COL1)='ABC'Copy
```

Because of the UPPER() function, the database doesn't utilize the index on COL1. If there isn't any way to avoid that function in SQL, you will have to create a new function-based index or have to generate custom columns in the database to improve performance.

Avoid using a wildcard (%) at the beginning of a predicate

The predicate LIKE `'%abc'` causes a full table scan. For example:

```
1.  SELECT * FROM TABLE1 WHERE COL1 LIKE '%ABC'Copy
```

In most cases, this wildcard usage (https://www.w3schools.com/sql/sql_wildcards.asp) brings major performance limitations.

Avoid unnecessary columns in SELECT clause

Instead of using 'SELECT *', always specify columns in the SELECT clause to improve MySQL performance. Because unnecessary columns cause additional load on the database, slowing down its performance as well whole systematic process.

Use inner join, instead of outer join if possible

Use outer join only when it is necessary. Using it needlessly not only limits database performance but also limits MySQL query optimization options, resulting in slower execution of SQL statements.

Use DISTINCT and UNION only if it is necessary

Using UNION and DISTINCT operators without any major purpose causes unwanted sorting and slowing down of SQL execution. Instead of UNION, using UNION ALL brings more efficiency in the process and improves MySQL performance more precisely.

The ORDER BY clause is mandatory in SQL if you expect to get a sorted result

The ORDER BY keyword sorts the result-set in predefined statement columns. Though the statement brings advantage for the database admins for getting the sorted data, it also produces a bit performance impact in the SQL execution. Because the query first needs to sort the data for producing the final result-set, causing a bit complex operation in the SQL execution.

You Might Also Like: How to Join Two Tables in MySQL (https://www.cloudways.com/blog/how-to-join-two-tables-mysql/)

Don't Use MySQL as a Queue

Queues can affect your database performance right from the core and could enter in your app databases without your knowledge. For instance, if you are setting up a status for a particular item so that a 'relevant process' can access it, you are unintentionally creating a queue. What it does is that it builds up extra load time to access the resource without any major reason.

Queues cause problems for two major reasons. They serialize your workload, preventing completion of tasks in parallel, and they often result in a table that contains work in process as well as historical data from already completed jobs. It not only adds latency to the application but also adds hindrance to the MySQL performance tuning.

You Might Also Like: How to Use Redis for Queuing (https://www.cloudways.com/blog/redis-for-queuing-in-laravel-5/)

Understand the Four Fundamental Resources

You need four fundamental resources to make database functions. CPU, disk, memory, and network. If anyone of these doesn't function correctly, it ultimately affects the database server and results in poor performance.

To understand the fundamental resources properly, you need to focus on two particular areas i.e choosing the right hardware and troubleshooting problems with it.

Always ensure to use all-round performance components when choosing hardware for the MySQL database. Not only opt for the best among the stack but also make sure that there should be the right balance between them. We have often seen that organizations tend to select servers with fast CPUs and large disks, but they get mistaken with starved memory which eventually kills performance.

In some scenarios, adding memory becomes highly substantial for improving performance when it comes to the magnitude. It does look a bit counterintuitive, but in most cases, the overutilization of disks affects directly to the database performance. As the deficiency of enough memory to hold the server's data proves costly in derailing database performance.

When it comes to troubleshooting, always keep in check the performance of all four fundamental resources. Validate qualitatively that they are performing as per the needs improvement in the norms. Taking this audit into regular consideration will quickly solve major occurring problems.

Pagination Queries

Applications that paginate (https://www.cloudways.com/blog/vue-pagination-in-laravel/) tend to bring the server down. In showing you a page of results, with a link to go to the next page, these applications typically group and sort in ways that can't use indexes, and they employ a LIMIT and offset function that causes the server to do a lot of work generating, then discarding rows.

You can find optimizations within the user interface itself. Instead of showing the exact number of pages in the results and links to an individual page, you can just show a link to the next page. You can also prevent people from going to irrelevant pages.

On the query side, instead of using LIMIT with offset, you can select one more row than you need, and when the user clicks the “next page” link, you can designate that final row as the starting point for the next set of results. For example, if the user has viewed a page with rows 101 through 120, you will have to select row 121 as well; to render the next page, you’d query the server for rows greater than or equal to 121, limit 21.

Optimizing MySQL Subqueries

The most important advice I can give you about subqueries is that you must prefer a join where possible, at least in current versions of MySQL.

Subqueries are the subject of intense work by the optimizer team, and upcoming versions of MySQL may have more subquery optimizations. Do keep a check on which of the optimizations will end up in released code, and how much difference they’ll make. My point here is that “prefer a join” is not future-proof advice. The server is getting smarter all the time, and the cases where you have to tell it how to do something instead of what results to return are becoming fewer.

Mysql Query Cache

One of the most important aspects of measuring performance is caching the content. MySQL provides database query caching which caches the SELECT statement text and the retrieved result. Hence, whenever you make a duplicate database, you call MySQL query cache, it will respond to you and show the result from the cache, and no call will be parsed repeatedly. In this way, you can maximize the MySQL cache optimization process.

To set up MySQL query cache, you must add a few settings to MySQL. First of all, you must check if query cache is available or not with the following command:

```
mysql> SHOW VARIABLES LIKE 'have_query_cache';
```

This will show the result, YES. This means MySQL cache is working fine.

```
+------------------+-------+

| Variable_name    | Value |

+------------------+-------+

| have_query_cache | YES   |

+------------------+-------+
```

Now, you can set up the MySQL query cache size and type. Remember the minimum default size is 40KB. The maximum size can be 32MB. You can set up the MySQL query_cache_size by using the following command:

```
1. mysql> SET GLOBAL query_cache_size = 40000;
```

Query cache type can determine the behavior of all connections. You can also disable the Query cache for queries like:

```
1. mysql> SET SESSION query_cache_type = OFF;
```

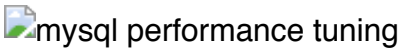
You can also set values like 0,1 and 2 for setting up the connection status.

Use Memcached for MySQL Caching

Memcached is a distributed memory caching system. It speeds up websites having large dynamic databases by storing database object in Dynamic Memory to reduce the pressure on a server, whenever an external data source requests a read. A Memcached layer reduces the number of times the database makes a request.

Memcached stores the values (v) with the key (k), and retrieves the values (v) with the key (k) without even parsing the database queries and stays away from all these hassles.

To read more about Memcached, you can read the guide on how to set up Memcache in php (https://www.cloudways.com/blog/memcached-with-php/).



Wrapping up!

This article provides in detail, the account of the best practices for database optimization and handy MySQL performance tuning tips every developer must know. It’s a complete guide for those backend devs, who are uncertain about their poor database performance and need some handy techniques to optimize MySQL database from the core.

If you want to add your thoughts on the topic or want to ask some questions regarding it, feel free to write your comments in the comments section.

Share This Article

(http

s://t

witt

er.c

om/i

nten

t/tw

eet?

text

=Ex

clusi

ve

MyS

QL

Perf

orm

anc

e

Tuni

ng

Tips

For

Bett

er

Dat

aba

se

Opti

miz

atio

n,&

url=

http

s://

ww

ww

w.cl

oud

way

s.co

m/bl

og/

mys

ql-

perf

orm

anc

e-

tuni

ng/

&via

=clo

udw

ays)

Shahroze Nawaz (https://www.cloudways.com/blog/author/shahroze-nawaz/)

Shahroze is a PHP Community Manager at Cloudways - A Managed PHP Hosting (https://www.cloudways.com/en/php-cloud-hosting.php) Platform. Besides his work life, he loves movies and travelling. You can email him at shahroze.nawaz@cloudways.com

THERE'S MORE TO READ.

(HTTPS://WWW.CLOUDWAYS.COM/BLOG/ADD-FACEBOOK-LOGIN-IN-PHP/)

PHP

How To Add Facebook Login to PHP Website (https://www.cloudways.com/blog/add-facebook-login-in-php/)

(HTTPS://WWW.CLOUDWAYS.COM/BLOG/CONTINUOUS-DEPLOYMENT-TOOLS/)

PHP

8 Powerful Continuous Deployment Tools to Consider in 2021 (https://www.cloudways.com/blog/continuous-deployment-tools/)

(HTTPS://WWW.CLOUDWAYS.COM/BLOG/INTEGRATE-CONTENTFUL-CMS-WITH-CLOUDWAYS/)

PHP

Integrate Contentful with Your Web Application Using Cloudways (https://www.cloudways.com/blog/integrate-contentful-cms-with-cloudways/)

PRODUCT & SOLUTION

WordPress Hosting (https://www.cloudways.com/en/wordpress-hosting.php)

Magento Hosting (https://www.cloudways.com/en/magento-hosting.php)

PHP Cloud Hosting (https://www.cloudways.com/en/php-hosting.php)

Laravel Hosting (https://www.cloudways.com/en/laravel-hosting.php)

Drupal Hosting (https://www.cloudways.com/en/drupal-hosting.php)

Joomla Hosting (https://www.cloudways.com/en/joomla-cloud.php)

PrestaShop Hosting (https://www.cloudways.com/en/prestashop-hosting.php)

WooCommerce Hosting (https://www.cloudways.com/en/woocommerce-hosting.php)

Cloudways Platform (https://platform.cloudways.com/signup)

Cloudways API (https://developers.cloudways.com/)

Breeze – Free WordPress Cache (https://www.cloudways.com/en/free-wordpress-cache-plugin-breeze.php)

Add-ons (https://www.cloudways.com/en/addons.php)

CloudwaysCDN (https://www.cloudways.com/en/cdn-services.php)

CloudwaysBot (https://www.cloudways.com/en/cloudwaysbot.php)

COMPANY

SUPPORT

QUICK LINKS

Follow Us On




(https://www.facebook.com/cloudways) (https://twitter.com/cloudways) (https://www.linkedin.com/company/cloudways) (https://www.cloudways.com/blog/upda

(https://www.cloudways.com/en/)

52 Springvale, Pope Pius XII Street Mosta MST2653, Malta