

Coveo Enterprise Search

Improving Query Performance

Applies to CES 5 and later

Certain elements, such as indexing speed and hard disk usage, are important when planning the deployment of a *Coveo Enterprise Search* (CES) system. Ultimately though, the goal of the system is to allow users to perform queries in order to access information in a precise and timely manner. Slow query speed can adversely affect the user experience and is sometimes difficult to diagnose.

This document presents advice on achieving better query performance in CES. It does not detail all possible tweaks for a particular system since it is often dependent on a particular setup; however, it attempts to cover the topics that apply to all systems and give instructions that can easily be applied to your environment.

Hardware Resources

Hardware is an important factor affecting the performance of a CES system and queries are no exception. The more powerful a CES server is, better the query performance is.

▶ CPU

Query processing can be very CPU-intensive; particularly the ranking phase, which sorts the results and displays the most relevant ones first. This phase requires CES to scan a large result set and perform calculations in order to quickly pinpoint the best results. Having a system with multiple cores helps query performance; especially if the CPUs are dedicated to CES (e.g. no other service is running on the CPU cores).

Starting with the upcoming version 6.0 of CES, the ranking phase will improve its use of multiple CPU cores on modern servers to boost performance even further. Installing this version on a multicore system will thus improve the query performance, especially for queries that return a large number of results.

▶ Memory

The amount of memory available for CES can affect query performance because CES uses several levels of caching to avoid having to access the hard disk as much as possible during the query process. The amount of memory available for the caches depends on the memory available on the server as well as the CPU architecture. A larger memory pool results in even more frequently-used data in the cache, thus improving performance. Using 64-bit systems is also highly recommended, as they allow CES to better use large amounts of memory and allocate even more memory for caches and such.

For more information concerning the configuration of the amount of memory used for caches, refer to the Cache Size section.

▶ Hard Disk

CES is a very I/O intensive application. Both indexing and querying processes need to frequently access the hard disk because the amount of data in a CES index is usually very large. As such,

faster hard drives will result in improved indexing and querying speeds. For optimal performance, SAS drives or SANs are recommended over SATA drives.

Performances are also affected by the number of services used by a particular hard drive. If CES shares a hard drive with another I/O intensive service, performance will adversely be affected. This applies to CES itself; when the CES system has more than one slice, each slice should store its index on a separate hard drive for best performance.

Mirrors and Web Front-Ends

Balancing the load of a heavily-used system across different servers is a well-know technique to improve performance and reduce the risk of data loss or the recovery time. The CES system is no different and the following explains the different ways to proceed:

▶ Mirroring

CES allows the index to be replicated on multiple mirrors. Mirror services are configured exactly like the main server (including slices); however, they do not perform indexing per se. Instead, when a transaction is applied to the index, the main server transfers new indexing data to the mirror.

Deploying mirrors in a CES system can be used to reduce risk of downtime in case of a computer crash and balance the load of queries across multiple servers; hence, resulting in better performances, as each server has fewer queries to process. The number of mirrors required depends on the load generated by the queries; therefore, it is different for each environment. Mirroring is not free of charge because data is transferred from the main service to the mirrors when transactions are applied, so deploying many mirrors at once is not recommended. Since mirrors can be added at any moment without limitations, it is recommended to deploy them on a needs basis.

Also, it is important to consider that even though the main server can also serve queries, it performs much more processing than mirrors (converting documents, crawling sources, etc.) As such, serving queries exclusively on a mirror can sometimes improve overall system performance. Note that this is heavily dependent on the environment; therefore, it is not advisable to use such a configuration in all cases.

▶ Web Front-Ends

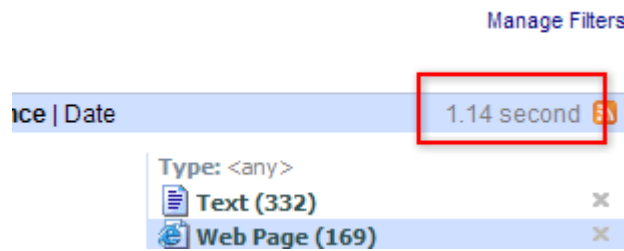
While mirrors can be used to balance the query load from a back-end perspective, multiple Web front-ends can be used to balance the load of Web access. The CES services' performance affects query speed, but user interaction is first and foremost performed via a Web page that is served by an IIS service.

If queries are slow when searching from a Web interface, it can be difficult to determine which part of the system is underperforming: the IIS or CES back-end service. Determining which service to tweak is essential to improve performance. There are many techniques that can be used to diagnose such problems.

To solve performance problems, analyze the Web server logs. If requests that do not hit the CES back-end are also slow, it can mean that the Web server itself has performance issues.

Enable query profiling in CES and compare the time it takes CES to process the query with the request processing time in the Web front-end. If both times are similar, then the performance problem is probably in the CES service; if the time displayed in the result page is significantly greater than the time it took CES to return results, then the problem is probably on the Web server side.

Note: Enabling query profiling is not free of charge and should only be performed with the help of the Coveo Technical Support.



When the Web server is overloaded, it is possible to balance the load on multiple Web front-ends in order to improve performance. Because it does not affect the performance of the CES service, it is important to determine beforehand where the performance problem lies; tests and instrumentations are recommended.

Note: The Coveo Web Service is hosted on the same Web server as the front-end; thus, performance problems in the Web Service can be corrected in a similar manner. There should be no performance gap between the Web front-end and the Web service.

Configuration

Aside from the effects of hardware and mirroring setup, many query performance problems can be traced to a particular CES configuration. Because many configuration properties can affect query performance, this section tries to cover the most important ones and gives tips on how to configure the CES system in order to maximize query performance while maintaining the same level of service.

▶ Hard Disk Usage

A CES system uses the hard disk to store multiple pieces of information. Aside from the contents of the index proper, data such as log files and connector data files are stored on the hard disk and in the default configuration; they are stored on the same drive as the CES index.

It is, however, possible to change the default configuration to store log files and connector data files in other locations. Storing the index, the log files and connector data files on separate drives improves the overall performance of CES and query performance in particular, since accessing index data for queries can be done simultaneously with connector access to their data or with log file activity.

▶ Cache Size

As discussed in the Memory section, query processing uses caching to keep in memory the data most frequently accessed, such as search terms, refine field information, etc. This speeds up queries since this information is often reused.

Caches are limited in the amount of memory they use to ensure that the rest of the CES system does not run out of memory. The amount of memory allocated to the caches can be configured in the **Advanced** page (Index > Advanced) of the Administration Tool.

⊙ Optimize for indexing and querying

Memory cache size MB (20.0 % of 2.0 GB of memory available for CES.) [?](#)

Default Aliases

Increasing the amount of memory allocated to the caches can increase query performance; however, it can also result in memory issues if the caches use too much memory, and not enough is left for other processing tasks. As a general rule, this setting should only be changed on systems with a large amount of available memory and even then, no more than 30-40% of the memory available for CES should be used for caching.

Important: If the CES system has multiple slices configured as separate *Windows* services on the same server, each slice is allocated the amount of memory specified in the configuration. Therefore, if for example, the system is configured to use 25% of memory for caching and there are 4 slices running on the server, the total amount of memory used for caching is 100%, which most likely causes out-of-memory problems. On such systems, the cache memory size should be lowered in the configuration so that each slice can have access to a memory cache without taking up all the memory.

▶ Refine Fields

Refine fields offer a great search user experience. It allows users to drill down in their search results very quickly; however, supporting such fields at query time is not free of charge. A portion of the query process is dedicated to processing the refine fields and the time it takes to complete this portion increases linearly with the number of refine fields used. Thus, adding a large number of refine fields can affect query performance. It is advised to perform proper testing and instrumentation when adding refine fields. Also, there are ways of minimizing the performance impact of such additions:

- Processing fields containing strings with multiple words is more expensive than fields containing a single value. However, when adding a new refine field, it is possible to specify a field to group on and another to display which can be different: the *field to group on* is used to fetch the refine field values and select the ones to return, while the *field to display* is used to present the refine value to the user. If a refine field containing multiple words has an equivalent field stored as a single value (for example, a @authorid field could match @authorname), grouping by the field with single values is faster, and the user can still see the author name in the search interface.

Edit Refine by Field

Title

Field to Group On

Field to Display

- When including refine fields that can contain multiple words, if the user clicks on a field value, a new query is generated including the exact value. If the value contains multiple words, the results in an exact phrase query take more time to process. It is possible to speed up the process if CES indexes this field as a **Nextwords** field. When indexing the values of a **Nextwords** field, CES indexes the field as pairs of words (for example, *Coveo Enterprise Search* would be indexed as *Coveo Enterprise* and *Enterprise Search*). *Exact phrase* queries on such fields are faster

because of this; therefore, if a refine field containing multiple words is created, the field name could be included in the list of **Nextwords** fields to speed up queries originating from such a refine field in the search interface. To configure the list of fields to index as *nextwords*, use the **NextwordsFields** element in the CES config file (stored in the root of the CES index: *Config\config.txt*).

Also, it is worth mentioning that adding search scopes does not impact query performance at all, since using a particular scope does not generate more queries than using the default one.

Important: Manual editing of the CES config file is a delicate process. Backup the original config file and make sure to understand the changes. When in doubt, contact Coveo Technical Support for help.

▶ Performance Mode

The **Advanced** page (Index > Advanced) of the Administration Tool contains a setting named **Performance mode**. This setting is used during the indexing process and has an effect on query performances. It controls the level of fragmentation¹ tolerated in search terms during the indexing process. Tolerating more fragmentation speeds up the indexing process; however, it slows down querying because the term needs to be defragmented each time it is loaded from disk. On the other hand, tolerating less fragmentation gives better query performances, but indexing is considerably slower.

The default value, **Optimize for indexing and querying**, is a good middle-ground. However, on most systems, there is a better usage scenario, especially if the system involves first creating an initial index that is quite large, and then simply refreshing the index with less changes afterwards:

1. Set the performance mode to **Optimize for indexing**.
2. Perform the original indexing of all sources. Let it run to completion.
3. Set the performance mode to **Optimize for querying**.
4. Compact the index.

By doing so, the initial index is optimized. Future indexing processes will be a bit slower because of the performance mode setting, but since it involves less updated documents, the performance hit will not be noticeable and the queries will run faster (the progress depends on the environment).

It is worth noting that compacting an index can always improve query performance, since index fragmentation can mean more I/O during query processing. The default compacting schedule (once per week) is a good setting to ensure the index stays optimized.

Example

Important: This example assumes CES 5.2 is installed. Future versions can include improvements that could change the following recommendations. If installing a later version of CES, contact Coveo Technical Support or your Coveo representative.

In this section, query performances are maximized in a more complex environment of the CES setup. This example uses techniques discussed previously and explains how to implement them.

The sample environment is used to index corporate emails, files, as well as a corporate intranet. It is expected to index up to 20 million documents and can be accessed via the intranet's search tool.

¹ This is not related to the fragmentation level of the index itself, which is related to the number of deleted documents in it.

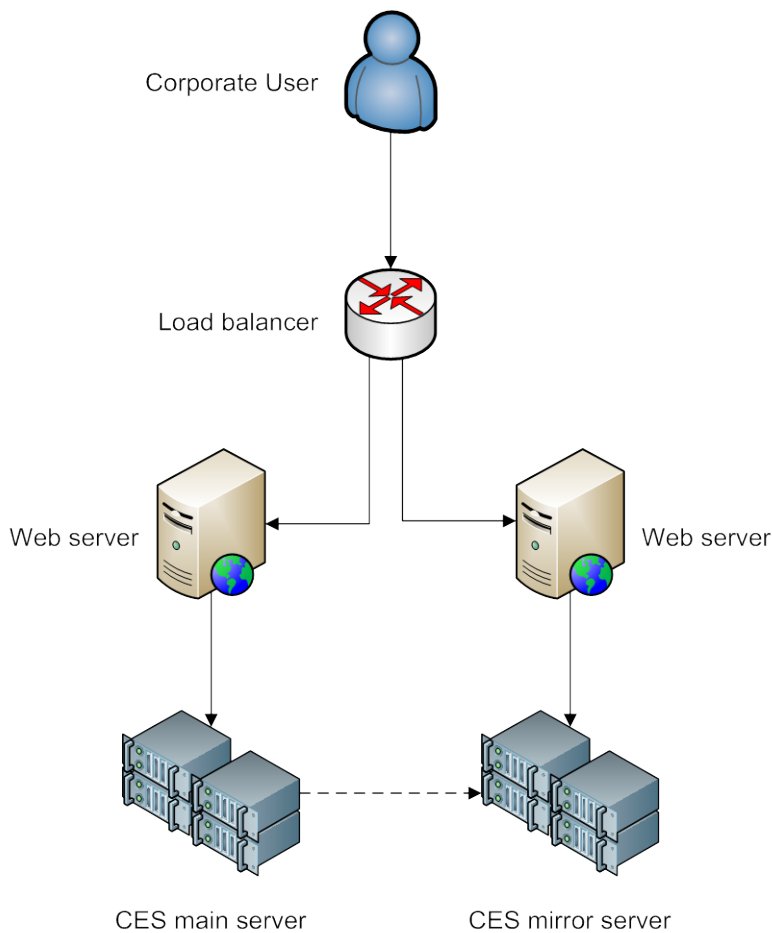
▶ **Mirrors, Slices and Web Front-Ends**

The number of documents present in this environment requires the use of multiple slices. A total of four slices means that each slice would host around 5 million documents, which is within the recommended use. In order to speed up the querying process, a mirror will also be deployed. This will ensure an uptime if a problem occurs on one of the main or mirror servers.

The Web front-ends will be embedded in the intranet's page. Since the corporate intranet is currently being served by two Web servers via a load balancer, the CES search interface will be installed directly on these servers. Monitoring of the Web servers shows only a light load, so the current hardware configuration of such servers is sufficient to host the search interfaces without changes. Each search interface will be configured to connect to one of the two CES back-end servers (main and mirror). Since the load is balanced across Web servers, the query load on the back-end servers will be load-balanced at the same time.

▶ **Hardware**

To ensure optimized performances at indexing and querying time, the CES system is hosted on dedicated servers. The main server must host four CES services (the main CES service and three slice services); therefore, a dedicated quad-core server with 16 GB of memory is used. The server is equipped with four hard drives, allowing each slice to sit on its own drive, which means that all services are able to perform I/O simultaneously. Also, the mirror uses a similar server, because it has the same slice configuration.



► Configuration

Important: Installing slice services on a mirror server is not done automatically with the CES installation kit. For more information, contact Coveo Technical Support

Once the CES services are all installed (main, mirror and all slices), the Administration Tool can be used to create the CES index on the main server. Afterwards, the mirror and external slices are created in the Administration Tool before performing the initial indexing, so that the data can be balanced properly across the slices.

Note: The index settings are not presented here, but are dependent on the environment.

Before initial indexing is performed, the performance mode is set to **Optimize for indexing**. Also, the memory cache size is set to 800 MB – this allows the caches to use 3.2 GB of memory on each server (since there are four slices).

Performance Mode

Optimize for indexing
 Optimize for querying
 Optimize for indexing and querying

Memory cache size MB

When the configuration is set, collections and sources are created and the initial indexing is started. Depending on the environment, it can take several days to complete.

Once the initial indexing is completed, the performance mode is switched to **Optimize for querying**. Refresh schedules are configured for each source and the suggested weekly optimization schedule is used. The index is also optimized once after the initial indexing; therefore the index is as fast as possible to handle queries.

Index Performance

Optimizing Index

0% 100%

Stop Optimizing

Conclusion

This example is a simple walkthrough of the decisions and configurations required to ensure good query performance in a CES environment. Remember that each environment is different; however, the content of this document should be interpreted as useful advice rather than a definitive guide.