Have More Time to Relax with an Enterprise Search Engine

Article contributed by dtSearch®

What if you could find anything instantly across terabytes of "Office" files, email archives, and even web-based data formats? And what if you could do your data search from anywhere — and extend this search capability to all of your coworkers? Think of the time this would save. This article will break down the processes that go into enterprise search and then follow with some more advanced tips.

Indexed search for enterprise search. The key to instant search across terabytes is to let the search engine first build a search index. Enterprise search can include indexed or unindexed search. dtSearch®, for example, offers both. But while unindexed search lets you query data without the overhead of a search index, it is much slower for multi-user concurrent searching across terabytes of data.

So what goes into a search index? An index is just an internal search engine guide that stores each unique word and number and the location of each in the data. For the end-user, indexing is easy; just point to the folders and the like to index, and the search engine does the rest. A single index can hold up to a terabyte of text, and there are no limits on the number of indexes that the search engine can build and simultaneously search.

Building an index is resource intensive. Indexed searching is resource-light. There are no limits on the number of concurrent search threads that can query the same index in a network environment. Online, each search thread can operate in a completely stateless manner, making it very easy to scale on a busy site.

Data sets can continue to evolve. Our sample search engine supports automatically updating all indexes using the Windows Task Scheduler to accommodate file edits, new files, and file deletions. Updating indexes does not block out searching, so individual and concurrent searching can continue even while indexes update.

Different data formats for enterprise search. Ultimately, what makes enterprise search so useful is that a single search request can span multiple different data formats and different data repositories. Here is how that works.

File format specification. To view a file outside of a search engine, you typically pull up that file in its native application, such as viewing a Word document in Microsoft Word, an email in Outlook, etc.

Building an index in the search engine. That's fine for viewing individual files. But for a search engine to build its index efficiently across terabytes of data, the search engine needs a different approach. That approach is to view each file in its binary format, bypassing the native application approach entirely.

Indexed searching is resource-light. There are no limits on the number of concurrent search threads that can query the same index in a network environment.

The problem is that when you look at the majority of "Office" files and the like in binary format, they look like a mishmash of binary codes. The main text can range from hard to read to completely inscrutable. Effective filtering of the text requires the application of a file format specification.

File format specification. The file format specification for "Office" formats can be hundreds of pages long and varies across different file types. The Microsoft Word file format is very different from the Access format, which is, in turn, very different from the file format for Excel, PowerPoint, OneNote, PDFs, emails, HTML, XML, etc. Correctly determining the file format of each binary file is, therefore, critical.

One way to make that determination is through the file format extension: a .PDF extension would indicate a PDF file, a .DOCX extension would indicate a Microsoft Word file, etc. However, it is all too easy to misapply a file format extension, saving a PDF with a .DOCX file extension or saving a Word document with a .PDF extension. While a mismatched file format extension can be accidental, it can also result from a desire to hide a particular file from scrutiny.

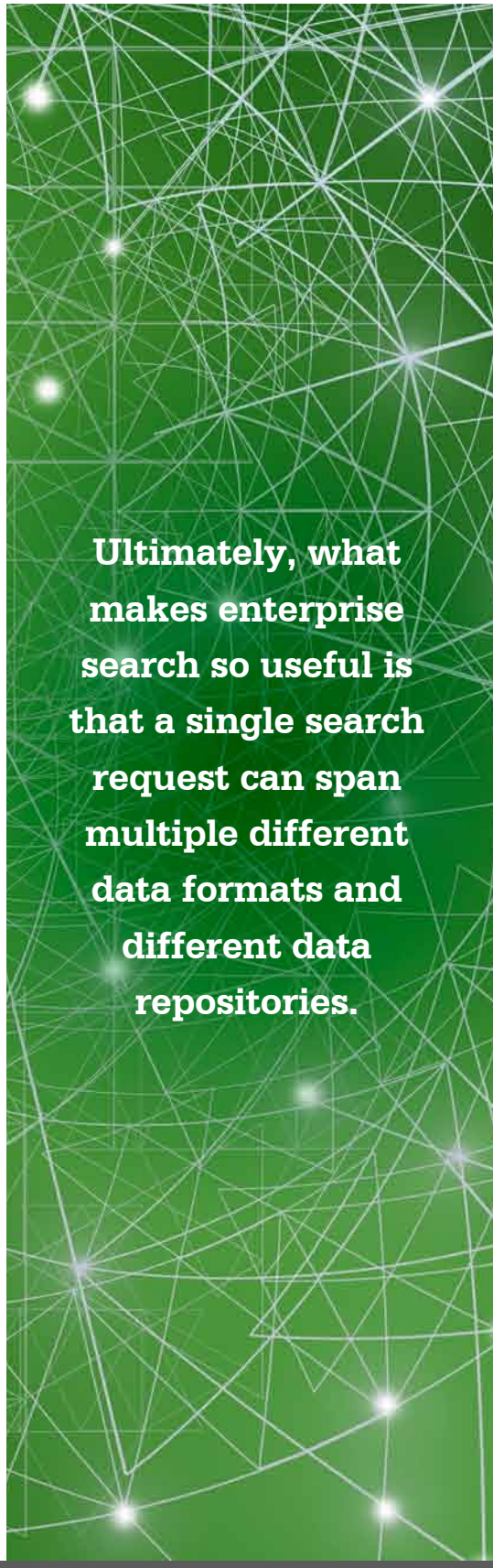
The surefire way to determine file format is for the search engine to look inside each binary file. After figuring out the file format from the binary file itself, the search engine can then apply the correct file format specification to parse the full-text and metadata of each item. Then the resulting information goes into building the index.

After indexing, the search engine will typically do a "mini-display" showing the search terms in context. The search engine can also show the full text of retrieved files as well with highlighted hits. To do so, the search engine will typically return to the binary format version and convert that to HTML for display inside a browser window inside the search engine, adding hit navigation for convenient browsing.

Types of indexed enterprise search engines. Because indexed searching is keyed off of a pre-built index, there are more than 25 different search options available for instant search. These include nearly any combination of word and phrase searching, Boolean and/or/not search expressions, and bilateral or unidirectional proximity searching. Search can cover the full text of indexed data or hone in on specific metadata, such as an email subject line.

Beyond word-oriented searching, an indexed search can also encompass numeric-oriented queries. A numeric-oriented query is like searching for specific numbers or numeric ranges and searching for specific dates or date ranges, even if the dates are in different formats, like 5/7/21 and June 11, 2022. The search engine can also find a different character and numeric configurations, including regular expression and digit character matching.

Article contributed by dtSearch®



Unicode. As the general standard for file text, Unicode covers hundreds of international languages, including English and other European languages, Asian languages, right-to-left languages like Hebrew and Arabic, and many more. Unicode lets any mix of languages coexist in a single document. All of that is in the binary format of a file and hence available to a search engine.

Advanced enterprise search engine tips. The description above represents the basics of how a search engine instantly searches terabytes. These are advanced tips.

Tip #1. Black writing against a black background, red writing against a red background, and the like can all but disappear in a file's native application view. However, because a search engine accesses files in binary format, all text is equally available to a search engine.

Tip #2. When viewing a file in its native application, it can take an enormous amount of clicking around in just the right sequence to even know that certain metadata is there. But all metadata is on an equal footing inside the binary format, making all metadata accessible to a search engine.

Tip #3. It is easy to forget when you are viewing a document in its final form that redlined edits may still exist in an alternate view of the document. If these are not eliminated entirely from a draft, such redlines will remain accessible to a search engine, both in the searching phase and in the file display phase.

Tip #4. Have you ever tried to copy what looks like words from a PDF file and gotten nothing when you tried to paste those words? This is what happens in an "image only" PDF. Such PDFs can be mixed in with other documents and are very hard to spot on their own. Since these are "image only," there is no digital text in them (other than filename and metadata). This means these are effectively blank to a text search engine. But search engines can flag "image only" PDFs at indexing time, letting you know that you need to run them through an OCR program like Adobe Acrobat – and then send them back to the search engine for full-text indexing.

Tip #5. Certain documents like emails and OCR'ed files can be full of typos. Setting fuzzy searching to a low level, like 1 or 2, will sift through common typographical errors. And fuzzy searching works on top of most other search options.

Tip #6. A search engine can flag certain personal information in files like credit card numbers. During the indexing process, the search engine can take a series of digits that may represent a credit card and run those digits through a credit card validation algorithm. Identifying where credit card numbers may appear in shared data lets you separately take steps to remediate the risk of such exposed personal information.

Tip #7. Normally, the search engine returns to the original source of the data to display it with highlighted hits. But if the original data is remote to where the search is running from, or the original data may disappear entirely, turning on caching will still allow file display with highlighted hits to work seamlessly. The disadvantage to activating caching is that it will make the index size much larger than otherwise.

Article contributed by dtSearch®