dtSearch Requests

dtSearch supports two types of search requests: natural language, and Boolean.

A natural language search is any sequence of text, like a sentence or a question. After a natural language search, dtSearch sorts retrieved documents by their relevance to your search request.

A Boolean search request consists of a group of words or phrases linked by connectors such as AND and OR that indicate the relationship between them.

For example:

- apple AND pear — Both words must be present
- apple OR pear — Either word can be present
- apple w/5 pear — “Apple” must occur within five words of “pear”
- apple NOT w/5 pear — “Apple” must not occur within five words of “pear”
- apple AND NOT pear — Only “apple” must be present
- name CONTAINS smith — The field name must contain “smith”

If you use more than one connector, you should use parentheses to indicate precisely for what you want to search.

For example, apple AND pear OR orange juice could mean (apple and pear) or orange, or it could mean apple and (pear or orange).

Words such as “if” and “the,” or noise words, are ignored in searches. Search terms may include the following special characters:

?	Matches any single character. For example: appl? matches “apply” or “apple.”
*	Matches any number of characters. For example: appl* matches “application.”
~	Stemming. For example: apply~ matches “apply,” “applies,” “applied.”
%	Fuzzy search. For example: ba%nana matches “banana,” “bananna.”
#	Phonic search. For example: #smith matches “smith,” “smythe.”
&	Synonym search. For example: fast& matches “quick.”
~~	Numeric range. For example: 12~~24 matches 18.
:	Variable term weighting. For example: apple:4 w/5 pear:1

Words and Phrases

You do not need to use any special punctuation or commands to search for a phrase. Simply enter the phrase the way it ordinarily appears. You can use a phrase anywhere in a search request.

For example: apple w/5 fruit salad

If a phrase contains a noise word, dtSearch will skip over the noise word when searching for it.

For example: a search for “statue of liberty” would retrieve any document containing the word “statue,” any intervening word, and the word “liberty.”

Punctuation inside of a search word is treated as a space.

For example:

- “can't” would be treated as a phrase consisting of two words: “can” and “t”
- “1843(c)(8)(ii)” would become “1843 c 8 ii” (four words)

Wildcards (* and ?)

A search word can contain the wildcard characters asterisk (*) and question mark (?). A question mark in a word matches any single character, and an asterisk matches any number of characters. The wildcard characters can be in any position in a word.

For example:

- *appl** would match “apple,” “application,” etc.
- **cipl** would match “principle,” “participle,” etc.
- *appl?* would match “apply” and “apple,” but not “apples.”
- *ap*ed* would match “applied,” “approved,” etc.

Use of the asterisk (*) wildcard character near the beginning of a word will slow searches.

Natural Language Searching

A natural language search request is any combination of words, phrases, or sentences. After a natural language search, dtSearch sorts retrieved documents by their relevance to your search request. Weighting of retrieved documents takes into account

- The number of documents in which each word in your search request appears (the more documents a word appears in, the less useful it is in distinguishing relevant from irrelevant documents)
- The number of times each word in the request appears in the documents
- The density of hits in each document; noise words and search connectors like NOT and OR are ignored

Synonym Searching

Synonym searching finds synonyms of a word in a search request.

For example, a search for “fast” would also find “quick.”

You can enable synonym searching for all words in a request, or you can enable synonym searching selectively by adding the ampersand (&) character after certain words in a request.

For example: fast& w/5 search.

Fuzzy Searching

Fuzzy searching will find a word even if it is misspelled.

For example, a fuzzy search for “apple” will find “appple.”

Fuzzy searching can be useful when you are searching text that may contain typographical errors. There are two ways to add fuzziness to searches:

1. Enable fuzziness for all of the words in your search request. You can adjust the level of fuzziness from 1 to 10.
2. You can also add fuzziness selectively using the percentage (%) character. The number of percentage characters you add determines the number of differences dtSearch will ignore when searching for a word. The position of the percentage characters determines how many letters at the start of the word have to match exactly.

   For example:

   - ba%nana will find words that begin with *ba* and have at most one difference between it and *banana.*
   - b%%anana will find words that begin with *b* and have at most two differences between it and *banana.*

Phonic Searching

Phonic searching looks for a word that sounds like the word you are searching for and begins with the same letter.

For example, a phonic search for “Smith” will also find “Smithe” and “Smythe.”

To ask dtSearch to search for a word phonically, put a pound sign (#) in front of the word in your search request.

For example: *#smith, #Johnson*

You can also check the Phonic searching box in the search form to enable phonic searching for all words in your search request. Phonic searching is somewhat slower than other types of searching and tends to make searches over-inclusive, so it is usually better to use the pound symbol to do phonic searches selectively.

Stemming

Stemming extends a search to cover grammatical variations on a word.

For example:

- “fish” would also find “fishing.”
- “applied” would also find “applying,” “applies,” and “apply.” There are two ways to add stemming to your searches:
    1. Check the Stemming box in the search form to enable stemming for all of the words in your search request. Stemming does not slow searches noticeably and is almost always helpful in making sure you find what you want.
    2. If you want to add stemming selectively, add a tilde (~) at the end of words that you want stemmed in a search.

       For example: apply~

Variable Term Weighting

When dtSearch sorts search results after a search, by default all words in a request count equally in counting hits. However, you can change this by specifying the relative weights for each term in your search request.

For example: apple:5 and pear:1 would retrieve the same documents as “apple” and “pear” but dtSearch would weigh “apple” five times as heavily as “pear” when sorting the results.

In a natural language search, dtSearch automatically weights terms based on an analysis of their distribution in your documents. If you provide specific term weights in a natural language search, these weights will override the weights dtSearch would otherwise assign.

AND Connector

Use the AND connector in a search request to connect two expressions, both of which must be found in any document retrieved.

For example:

- “apple pie” and “poached pear” would retrieve any document that contained both phrases.
- (apple or banana) and (pear w/5 grape) would retrieve any document that contained either “apple” or “banana,” and contained “pear” within five words of “grape.”

OR Connector

Use the OR connector in a search request to connect two expressions, at least one of which must be found in any document retrieved.

For example: “apple pie” or “poached pear” would retrieve any document that contained “apple pie,” “poached pear,” or both.

W/N Connector

Use the W/N connector in a search request to specify that one word or phrase must occur within a number of words of the other.

For example:

- “apple w/5 pear” would retrieve any document that contained “apple” within five words of “pear.”
- (apple or pear) w/5 banana
- (apple w/5 banana) w/10 pear
- (apple and banana) w/10 pear

Some types of complex expressions using the W/N connector will produce ambiguous results and should not be used.

For example:

- (apple and banana) w/10 (pear and grape)
- (apple w/10 banana) w/10 (pear and grape)

In general, at least one of the two expressions connected by W/N must be a single word or phrase or a group of words and phrases connected by OR.

For example:

- (apple and banana) w/10 (pear or grape)
- (apple and banana) w/10 orange tree

dtSearch uses two built in search words to mark the beginning and end of a file: xfirstword and xlastword. The terms are useful if you want to limit a search to the beginning or end of a file.

For example: “apple w/10 xlastword” would search for apple within ten words of the end of a document.

NOT and NOT W/N

Use NOT in front of any search expression to reverse its meaning. This allows you to exclude documents from a search.

For example: apple sauce AND NOT pear

NOT standing alone can be the start of a search request.

For example: “NOT pear” would retrieve all documents that did not contain “pear.”

If NOT is not the first connector in a request, you need to use either AND or OR with NOT.

For example:

- apple OR NOT pear
- NOT (apple w/5 pear)

The NOT W/ ("not within") operator allows you to search for a word or phrase not in association with another word or phrase.

For example: apple not w/20 pear

Unlike the W/ operator, NOT W/ is not symmetrical. That is, apple not w/20 pear is not the same as pear not w/20 apple. In the apple not w/20 pear request, dtSearch searches for apple and excludes cases where apple is too close to pear. In the pear not w/20 apple request, dtSearch searches for pear and excludes cases where pear is too close to apple.

Numeric Range Searching

A numeric range search is a search for any numbers that fall within a range. To add a numeric range component to a search request, enter the upper and lower bounds of the search separated by two tildes (~~).

For example: apple w/5 12~~17 would find any document containing “apple” within five words of a number between 12 and 17.

Numeric range searches only work with positive integers. A numeric range search includes the upper and lower bounds (so 12 and 17 would be retrieved in the above example).

For purposes of numeric range searching, decimal points and commas are treated as spaces, and minus signs are ignored.

For example: –123,456.78 would be interpreted as: 123 456 78 (three numbers).

Using alphabet customization, the interpretation of punctuation characters can be changed.

For example: 123,456.78 would be interpreted as 12345678.
