Corpus Middelnederlands application manual

Table of contents

Introduction

This manual describes the corpus exploitation environment for the *Corpus Middelnederlands*. The corpus application is developed by the Dutch Language Institute (Instituut voor de Nederlandse Taal or INT). The backend of the application is the BlackLab Lucene based search engine developed for corpora with token-based annotation (https://blacklab.ivdnt.org/). The web-based frontend is a further development of the corpus-frontend application developed by INT (https://github.com/instituutnederlandsetaal/blacklab-frontend). Its design is inspired by the first version of the OpenSoNaR user interface by Tilburg University and Radboud University (https://github.com/Taalmonsters/WhiteLab2.0).

Information about the corpus

The Corpus Middelnederlands in the current release is a collection of 372 documents from the period of 1300-1550. The main sources for this corpus are the rhyming texts and prose texts from the CD-ROM Middle Dutch, compiled by the INL/INT and published in 1998 by the Flemish Standaard Uitgeverij and the Dutch Sdu. (The CD-ROM texts, taken from the corpus Gysseling have not been included here, for these texts please consult the online Corpus Gysseling.).

This corpus is in part an extension of the Corpus Middelnederlands that was integrated in the Nederlab portal in 2018. Two completely French texts were removed. The metadata has also been corrected.

The *Corpus Middelnederlands* presented in this application contains classical works of Middle Dutch literature like *Beatrijs*, *Van den vos Reynaerde*, the *abele spelen*, the stories about King Arthur or about Charlemagne, all texts from the famous Gruuthuse manuscript (including the Egidius song), but also many of the lesser known or less researched texts, such as prose adaptations of the rhyming knight's tales (the so-called 'chapbooks'), collections of songs such as the Antwerp Songbook, several Bible translations, hagiographies, books of prayer, chronicles, and all kinds of religious, didactic and scientific treatises, medical manuals and recipes.

A number of texts belonging to the so-called *artes literature* have been added to these source texts, such as the Hattem manuscript (C5) and *Van der proprieteyten der dinghen* ('On the Properties of Things') by Bartholomeus Anglicus – both published and made available by the Werkgroep Middelnederlandse Artesliteratuur (WEMAL) – and the *Circa Instans* and the *Trotula*.

GiGaNT Lexicon service

To make the *Corpus Middelnederlands* more accessible, suggestions for query expansion are given, using the INT lexicon service with the historical computational lexicon GiGaNT-HILEX.

The current version of GiGaNT-HILEX in the lexicon service contains the lexicon modules based on the *Dictionary of the Dutch Language* (*Woordenboek der Nederlandsche Taal*, WNT) and the *Dictionary of Middle Dutch* (*Middelnederlandsch Woordenboek*, MNW).

If you want to make use of this service, please contact Katrien Depuydt (katrien.depuydt@ivdnt.org).

Metadata categories

The *Corpus Middelnederlands* has been enriched with an elaborate set of metadata categories. These metadata are described below. In the corpus application it is possible to limit a search by filtering on metadata categories.

Date

Witness Year

For each document in this corpus, we indicate the period in which the manuscript, providing us the text, was written. Witness Year does not necessarily refer to the period in which the text itself was written. It only concerns the carrier of the text.

Witness Year cannot be stated with the same accuracy for every document. For example, the manuscript with the *Cyrurgie* of Jan Yperman can be dated exactly to 1351, while the source in which the text of *Wrake van Ragisel* is included, originated between 1340-1360.

Permissive / strict

It is possible to do a permissive and strict search for Witness Year. What exactly is the difference between the two options? An example can clarify this. Suppose you want to investigate sources that were written between 1425 and 1450.

If you choose to do a Strict search by Witness Year, the search query will only result in manuscripts that were produced later than 1425 but before 1450. Among the 11 results is a manuscript from the period 1434-1436 in which the *Spiegel der sonden* is included.

If, on the other hand, you choose the option Permissive, no less than 42 documents are found, one of which is Geert Grote's *Getijdenboek*, which was handed down in a manuscript dating from the period 1450-1470.

Text type

Fictionality

The documents have been divided into fictional texts (fiction) and non-fictional texts (non-fiction). However, there are cases in which we will classify a text as fiction and medieval people as non-fiction. For this corpus, we applied the medieval point of view.

Genre

The texts are divided into two main genres: *prose* and *verse* and one mixed genre *prose+verse*, used for texts where prose and verse approximately balance each other.

Subgenre

The texts can be sorted out using one or more of the twenty-nine different subgenre labels. These subgenre labels may indicate a general text category as well as a more specific one; they may touch either the content of a text or its form (e.g. *stanzaic*).

- *Alchemy*: texts dealing with alchemical knowledge or information

- *Artes*: non-religious and non-fictional texts written for utilitarian and instructive purposes
- *Artifices*: descriptions of how to devise and use specific expedients or how to perform certain tricks
- *Astronomy / astrology*: texts describing the practical application of the medieval art or science of astronomy
- *Biblical texts*: Bible, lectionaria, diatessara
- *Biology*: texts with biological information
- *Chapbook*: printed books with popular literature, consisting chiefly of adaptations of chivalric romances, tales, ballads, tracts etc.
- *Chiromancy*: texts describing the art of telling the characters and fortunes of persons by inspections of their hands
- *Chivalric romance*: mostly versified stories about the idealized world of knights
- *Drama*: texts with one or more stage plays
- *Epic*: narrative literary texts
- *History*: texts with historical information
- *Legend*: texts offering a traditional story sometimes regarded as historical but not authenticated; a myth, a fable
- *Legendary hagiography*: descriptions of the life of saints
- *Lexicon*: text offering lexicographical information (a vocabulary or wordlist);
- *Linguistics*: texts offering linguistic information
- *Medicine*: texts with medical information or instructions
- *Metallurgy*: texts with information on (the use) of metals
- *Natural history*: texts providing in a popular form the scientific study of nature in general or animals, plants, even stones, in particular
- *Philosophy / ethics*: texts with life lessons, instructions of how to behave in life and social traffic, based on specific philosophical or secular grounds
- *Physiognomy*: texts describing the art of telling the characters and fortunes of persons by inspections of their physiognomy
- *Prayer*: texts containing a collection of prayers; also: book of hours
- *Religious*: texts dealing with religion or religious matters
- *Secular*: texts dealing with non-religious matters
- *Science*: texts offering scientific information
- *Song*: texts containing a collection of songs
- *Stanzaic*: texts composed in the form of stanzas
- *Technic*: texts with technical information or instructions
- *Theology / ethics*: texts with life lessons, instructions of how to behave in life and social traffic, based on theological or religious grounds
- *Travelogue*: travel books

Title and author

Words in title

Every document has a title. Usually this is the title as we know it from the editions used for this corpus.

This search field is provided with a list, which contains suggestions for possible search terms in alphabetical order, based on the characters typed in.

Author

It is possible to search by author name. However, for most of the documents in this corpus the author is unknown or uncertain.

Application user manual

The language of the corpus application is set to Dutch by default. Press the globe icon in the top right corner to select English.

Getting started

Here are a few examples of what you can do with the corpus application (links will take you to the application):

- To search for a specific word form, use Simple Search or Extended search.
  - Simple search for Word *gesont*
  - Extended search for Word *gesont*
- To search for words satisfying a certain pattern, use *wildcards* in Simple Search or Extended Search, or *regular expressions* in Advanced Search and Expert Search.
  - words starting with *ver* and ending with *len* in Simple Search
  - words starting with *ver* and ending with *len* in Extended Search
  - words starting with *ver* and ending in *erin* with (mostly) one syllable in between in Advanced Search
  - words starting with *ver* and ending in *eren* with (mostly) one syllable in between in Expert Search
- To see which unique forms occur as a result of your search, use Group Results.
  - example Group by Annotation: different words following *lieve*
  - example Group by Annotation: different words preceding the word *huis*
- To explore the distribution of document properties in the corpus, use the Explore feature.
  - example: characteristics of the genres
  - example: subgenres in the corpus

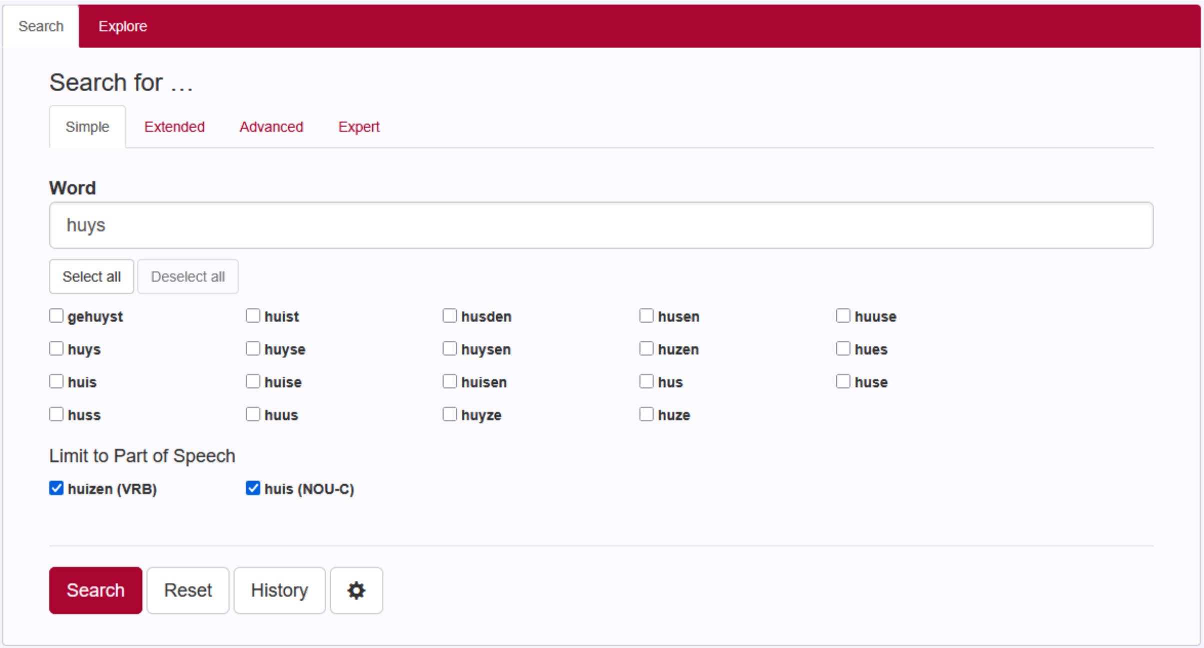
Searching the corpus

Simple search

Search

The Simple Search allows you to quickly search for specific word forms (e.g. *huys*). After entering a search term, a spinner briefly appears on the right side of the search bar. Based on the keyed in word, suggestions are given of possible variants of spelling and/or form from the GiGaNT-lexicon. If you know exactly which word you are looking for, you can also – while the wheel is spinning – press Enter directly. The search will then start immediately.

Based on the information in this lexicon all spelling variants of the search term found are suggested (see the screenshot below). You can then choose from the presented suggestions or select all at the same time (Select all). To make your search even more targeted, it is also possible to limit the search to the parts of speech that were found in the historic component of the GiGaNT-lexicon in connection to the search term.



Word

huys

□ gehuyst
□ huys
□ huis
□ huss

□ huist
□ huyse
□ huise
□ huus

□ husden
□ huysen
□ huisen
□ huyze

□ husen
□ huzen
□ hus
□ huze

□ huuse
□ hues
□ huse

Limit to Part of Speech

☑ huizen (VRB)
☑ huis (NOU-C)

It is also possible to enter a phrase: *bij haers vaders huys* or *princen des huys*. You will then find all occurrences of that exact phrase. Furthermore, you can search for different values simultaneously by separating them without spaces by a vertical line, e.g. *god|man|lief* or - with the use of wildcards - *god|aan*|hond*.

Note that in Simple Search the patterns will be matched case-insensitively**:** *ghemaket* for instance will deliver the same results as *GHEMAKET* or *Ghemaket*. See the paragraph Grouping results in Per Hit view to see how it is nevertheless possible to distinguish between uppercase and lowercase letters, or go to Extended Search.

Wildcards

In Simple Search, the use of wildcards can prove good service to search for specific word forms. A wildcard is a symbol used to replace or represent one or more characters. The following two wildcards are supported:

* The asterisk matches any character zero or more times. Therefore, searching for *a*n* matches all word forms that start with an *a* and end with a *n*, e.g. *an*, *anzoemen* and *anderen*.

? The question mark matches a single character once. Therefore, searching for *b?n* matches *only* three-letter values starting with an *b* and ending with a *n*, e.g. *ban*, *ben*, *bin*, *bon*, *bun* and *byn*.

This wildcard can be used more than once. Thus *b???n* matches *boven, buten*, *bouen*, *biden* and *began*.

Note that searching with wildcards is limited to Simple Search and Extended Search. [In Advanced Search and Expert Search you can use so-called regular expressions instead of wildcards.]

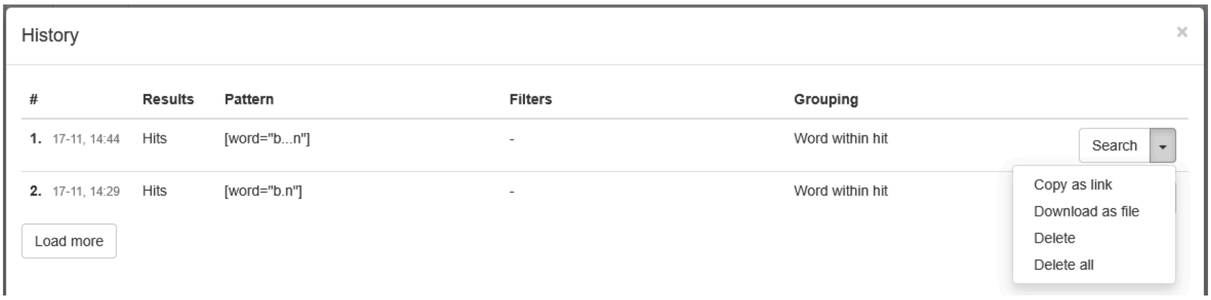
Reset

You can start a new search by pressing the Reset button. By doing so, both the search query and the hits found will be cleared. Your search history, however, will remain unchanged.

Note that it is also possible to start a new search by entering a new word or phrase in the search field Word.

History

The History button will display your query history. Per search query there are several possibilities (as shown in the screenshot below): you can perform the search query again (Search), you can copy the search query as a link (Copy as link), you can download the search query as a file (Download as file), you can delete a single search query (Delete) or delete all search queries (Delete all).



History

#	Results	Pattern	Filters	Grouping
1. 17-11, 14:44	Hits	[word="b...n"]	-	Word within hit
2. 17-11, 14:29	Hits	[word="b.n"]	-	Word within hit

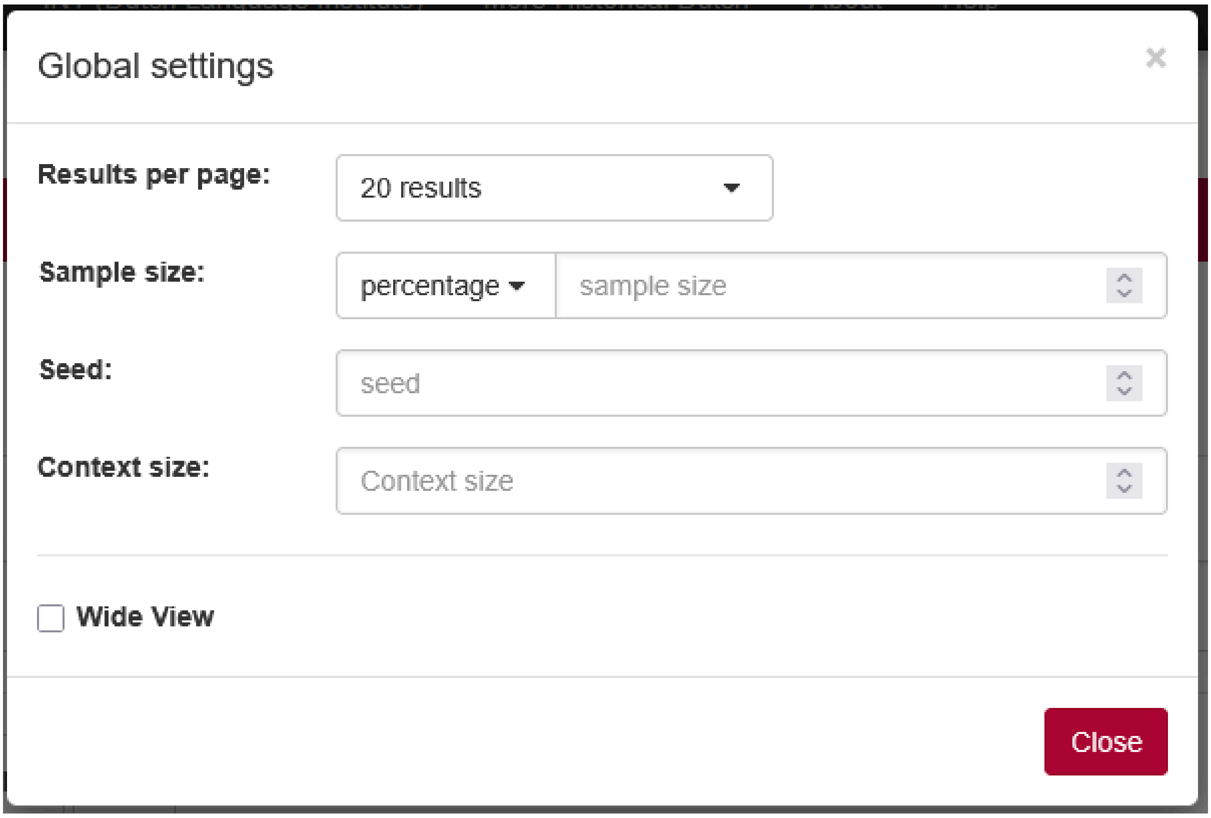
Every search query has its own url. If you copy this url via History (Copy as link) or directly from the address bar of your browser, you can send it to someone else who can import this link via Import from a link. It offers that person the possibility to run the search on his own computer.

Global settings

The Global settings dialogue, activated by pressing the wheel button, allows you to configure five settings: Results per page, Sample size, Seed, Context size and Wide View.

- *Results per page*: you can choose whether you want 20, 50, 100 or 200 results to be shown;

- *Sample size:* selecting a value here will instruct the search engine to return a random sample drawn from the complete result set. The sample size can be limited by
    - a percentage of the total number of search results (percentage);
    - the number of results displayed (count).
- *Seed:* a ‘random seed’ is a number used to initialize a so-called pseudo-random number generator. Keeping the same seed will ensure that two samples drawn from the same result set are identical. A new seed will most likely result in a different sample;
- *Context size*: by entering a number you can determine the number of words Before hit and After hit;
- *Wide View:* the default setting is ‘small view’; you can change to Wide View by ticking the checkbox.



Global settings

Results per page: 20 results

Sample size: percentage | sample size

Seed: seed

Context size: Context size

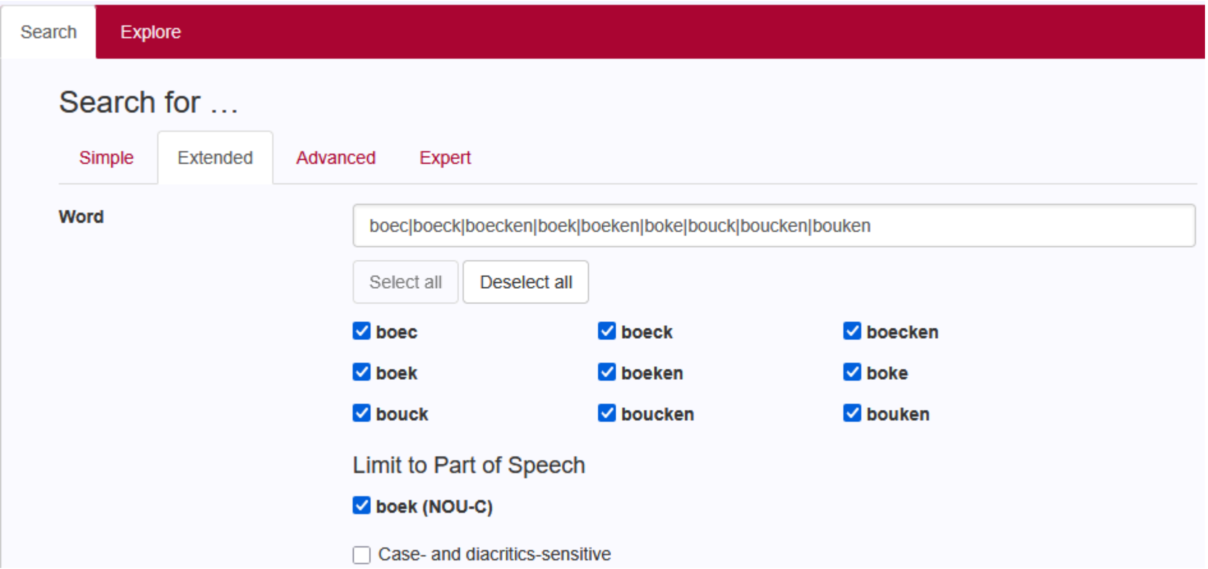
□ Wide View

Extended search

Like in Simple search, Extended Search allows you to quickly search for specific word forms or phrases. The search is performed in the same way as described for Simple Search.

After entering a search term in the search field Word, a spinner briefly appears on the right side of the search bar. Based on the keyed in word, suggestions are given of possible variants of spelling and/or forms from the GiGaNT-lexicon. If you know exactly which word you're looking for, you can also – while the wheel is spinning – press Enter directly. The search will then start immediately. Like in Simple Search, you can also enter a phrase here.

Based on the information in this lexicon all spelling variants of the search term found are suggested To make your search even more targeted, it is also possible to limit the search to certain parts of speech in connection to the search term. You can then choose from the presented suggestions or select all at the same time (Select all). In the screenshot below, all options have been selected.



Word: boec|boeck|boecken|boek|boeken|boke|bouck|boucken|bouken

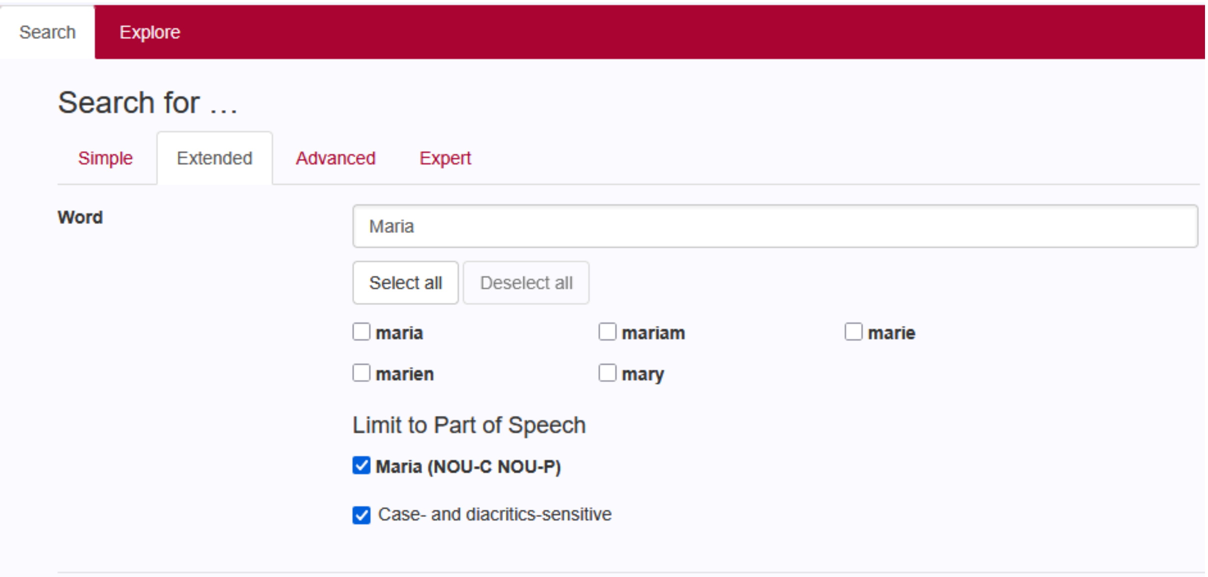
☑ boec ☑ boeck ☑ boecken
☑ boek ☑ boeken ☑ boke
☑ bouck ☑ boucken ☑ bouken

Limit to Part of Speech

☑ boek (NOU-C)

□ Case- and diacritics-sensitive

Extended Search allows to search case- and diacritics-sensitive. Note that the default setting for search is case- and diacritics-insensitive. For example, searching for the Word *maria* will result in 3141 occurrences of this name. By ticking the box Case- and diacritics-sensitive you will only find 1795 occurrences of the Word *maria*, but none of *Maria, MAria* or *MARIA*. In order to directly find only occurrences of the Word (form) *Maria* (1328x), use the search term *Maria* and tick the box Case- and diacritics-sensitive under the search field Word (as shown below).



Word: Maria

□ maria □ mariam □ marie
□ marien □ mary

Limit to Part of Speech

☑ Maria (NOU-C NOU-P)

☑ Case- and diacritics-sensitive

Like in Simple Search, wildcards are supported in Extended Search. (See for a short explanation of wildcards Simple Search.)

Filter search by

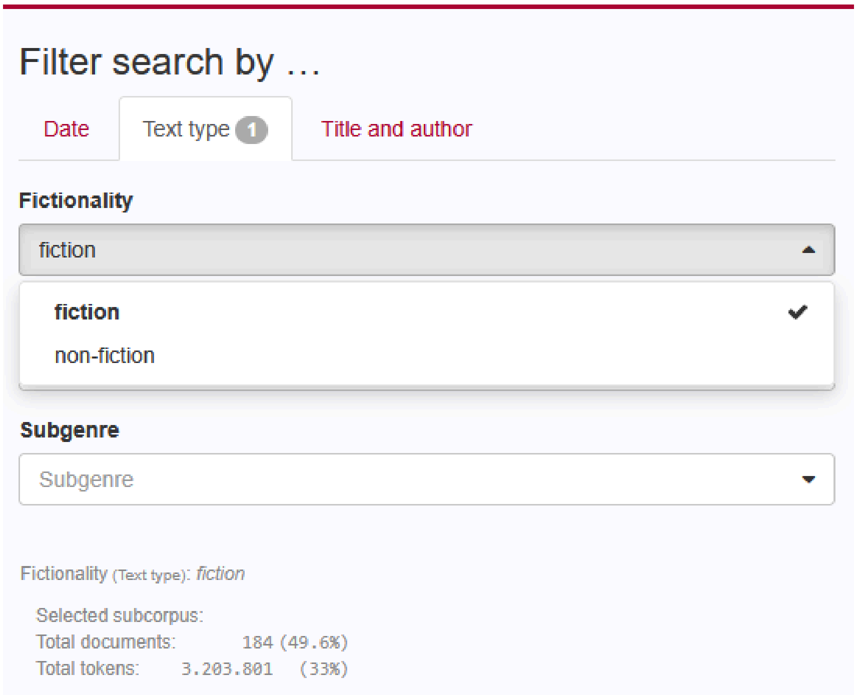
At the right side you will find the option to limit your query to a subset of documents with specific metadata values. You can apply different filters for Date (*Witness Year*), Text type (*Fictionality,* *Genre, Subgenre*) and Title and Author (*Words in title, Author*). To view the results for all documents simply leave the attributes in the filtering form empty.

Filter by date

You can select specific dates by entering the same data in the ‘from’ row as in the ‘to’ row. If you do not enter a date, the entire corpus is searched. If you want to filter by another date or another period, please press the reset button.

Filter by other metadata

There are two different ways to specify a filter, depending on the field type. You can either fill in a value yourself – beside Year above only for Words in Title – or choose one or more values from a drop-down list – for instance Fictionality. The drop-down list has been applied especially when the number of values to choose from is relatively small. Fictionality for instance has only two possibilities. You can pick one of these values by clicking on it; your choice will be marked with a tick. It is possible to choose several values. If you want to delete a selection, you can click on the corresponding line again. To close the drop-down list, you can either press the upward pointing arrow in the upper right corner or simply press escape.



Filter search by …

Date | Text type 1 | Title and author

Fictionality

fiction ▲

fiction ✓

non-fiction

Subgenre

Subgenre ▼

Fictionality (Text type): *fiction*

```
Selected subcorpus:
Total documents:        184 (49.6%)
Total tokens:     3.203.801   (33%)
```

By means of a number at the top of Filter search by, the number of values used to filter on, is displayed as can be seen in the screenshot above.

When on the other hand the set of possible values is relatively large (Words in title), you have to type a specific value in the search field. After entering a character in the search field, a list of possible values is suggested. Clicking on an auto-completed value will paste that value in the field. Note that this only works with a single word, like *Brabantsche*. In order to search for an exact phrase, i.e. a multiple word value, it must be surrounded by double quotes. For instance, in the field Words in title "*Brabantsche* *" will result in three documents containing the word *Brabantsche*.

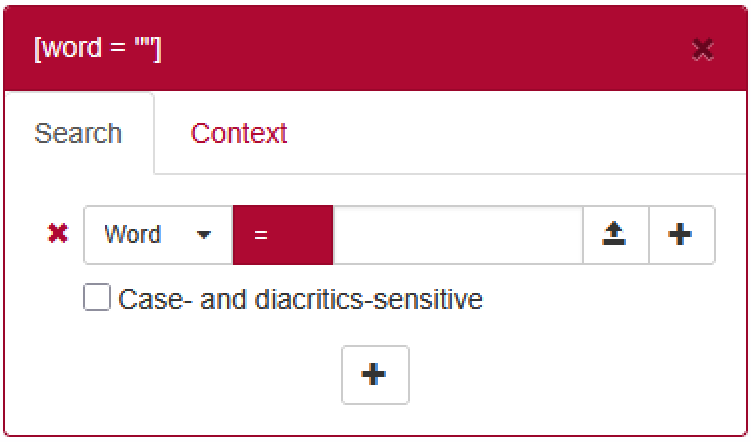
For a detailed description of the metadata, see the section Metadata categories.

Advanced search

The query builder

The basic building block in the query builder is the *token box* (see below). Each box represents a token – usually just a single word – or a simple repetition of tokens; when multiple tokens are used, they are matched in order from left to right.

You can use the query builder to create complex queries without writing CQL (here: Corpus Query Language). Therefore, it is easy to use.



[word = ""]

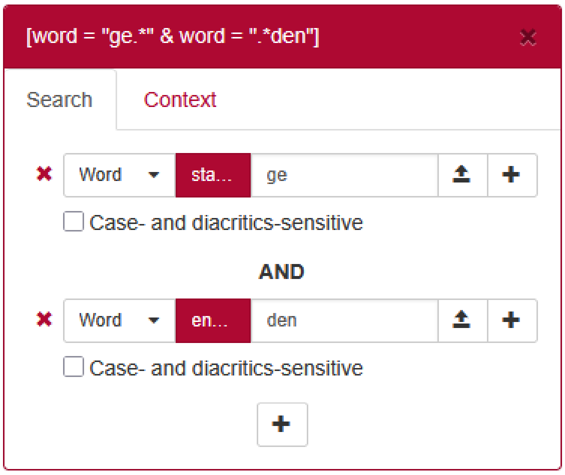
A token box in the querybuilder has two tabs: Search and Context.

The tab Search

The tab Search contains a set of attributes a token in the corpus must have to be matched by the query. By clicking the +-button on the right hand side of this token, you can add new attributes (see below). Then enter a value that the attribute must have for the token to be found. The search command Word 'starts with' *ge* and Word 'ends with' *den* for example results in both verbal forms (*gereden*, *geboden*, *gelden*) and plural and inflected nouns (*genaden*, *geboden*).

It is only possible to search by word forms. However, you can specify whether that word form should be equal or not equal to the entered search term. You can also specify whether or not a word should begin or end with a particular character combination.

The CQL query generated to match this token (the *token query*) in the corpus is displayed in the top bar of the box, to help you understand what is happening internally. The following applies to our example:



[word = "ge.*" & word = ".*den"]

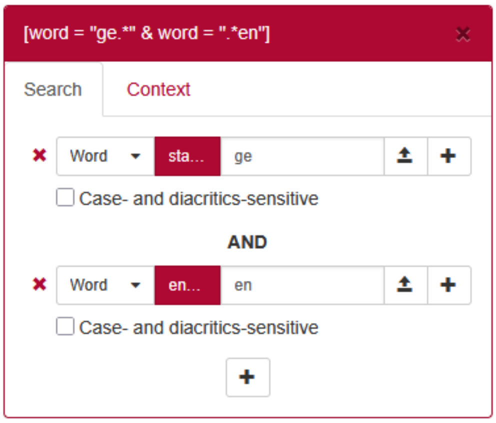
Token attributes

Specifying token attributes is similar to the Extended Search form. Select which attribute a token should have, and enter the value that the attribute must have for the token to be matched. Attributes in the query builder are interpreted as *regular expressions*. Note that this is different from the Extended Search, where token patterns use wildcards.

Going beyond single-attribute token queries, a token box also allows you to combine several attributes and to specify repetition options.

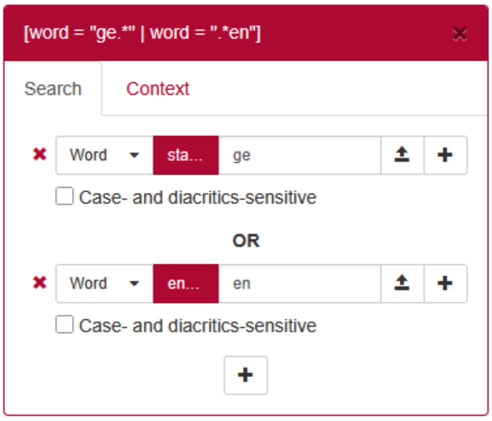
Adding attributes to a token box

Using the +-button, new attributes can be added. Two options exist: *AND* and *OR*.



[word = "ge.*" & word = ".*en"]

The *AND* option creates a new attribute restriction that a token must match in addition to the ones which were already there. As an example: suppose we want to match past participles of strong verbs. First, fill in the attribute Word 'starts with' *ge*, then click +, choose *AND*, and choose Word 'ends with' *en*.



[word = "ge.*" | word = ".*en"]

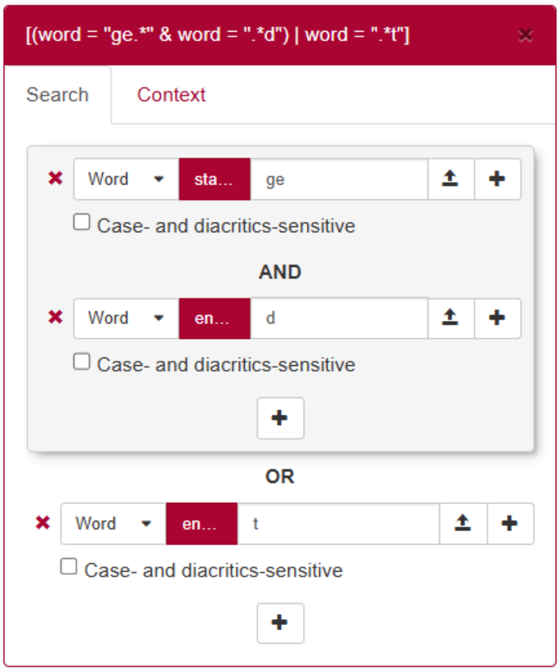
Similarly, creating a new attribute using *OR* will create a token query matching tokens that have the original attribute *or* the new attribute. For instance, enter Word 'starts with' *ge*, add a new attribute with the *OR* option and enter Word 'ends with' *en* to match tokens as *gedaen, gelijc* and *hebben, comen*.

Function of the two +-buttons in a token box

The difference between the +-sign on the right of an attribute and the one below it, is that the +-sign on the right keeps the newly added attribute ‘within a subclause’. This is most easily explained by means of an example.

Suppose we want to look for past participles of weak verbs, i.e. verbs that end in an -*d* or a -*t*. If we add the attributes in the order Word ‘starts with’ *ge* AND Word ‘ends with’ *d*, OR Words ‘ends with’ *t* using the +-signs **below** the attributes, as in the left screenshot below, we get the token query [(word = "ge.*" & word = ".*d") | word = ".*t"]. This will also match forms such as *stadt, niet*, so this is not what we were after.

If, on the other hand, we add OR Word ‘ends with’ *t* with the +-sign to the **right** of the attribute Word ‘ends with’ *d*, it will be inserted in a subclause, thus resulting in the correct query [word = "ge.*" & (word = ".*d" | word = ".*t")], as shown in the right screenshot below.



[(word = "ge.*" & word = ".*d") | word = ".*t"]

Word sta... ge

□ Case- and diacritics-sensitive

AND

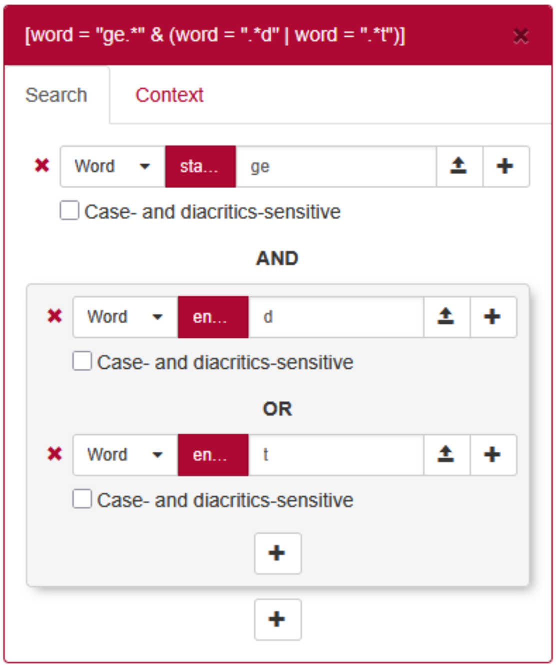
Word en... d

□ Case- and diacritics-sensitive

OR

Word en... t

□ Case- and diacritics-sensitive



[word = "ge.*" & (word = ".*d" | word = ".*t")]

Word sta... ge

□ Case- and diacritics-sensitive

AND

Word en... d

□ Case- and diacritics-sensitive

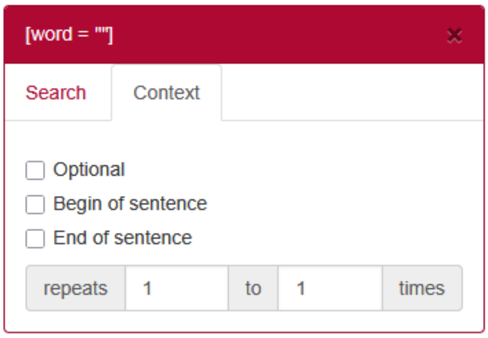
OR

Word en... t

□ Case- and diacritics-sensitive

The tab Context

The tab Context specifies the contextual properties, such as whether the token occurs at the end of a sentence, and the repetition pattern:



[word = ""]

□ Optional

□ Begin of sentence

□ End of sentence

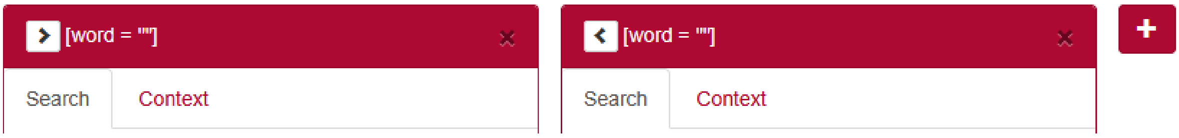
repeats 1 to 1 times

Managing sequences of token boxes

There are three ways to manage the sequence and the number of token boxes:

- *Rearrange* a token by clicking on the arrow in the top-left corner of a box (1). This arrow only appears if there are multiple token boxes.
- *Delete* a token by clicking the x in the top-right corner of a box (2).
- *Create a new token box* by clicking the +-button next to the upper right corner of the utmost right token box (3).

↓ (1) ↓ (2) ↓ (1) ↓ (2) ↓ (3)



Uploading value lists in the query builder

It is also possible to upload a list of values, separated by a white space. To do so, click the upload button (with the arrow pointing upwards) and select a text file. Tokens will then be matched for any of the values from the file.

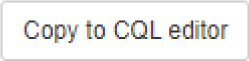
Note that this function only works for *.txt-files. If you are using a text editor like Word, you have to save your file as a *.txt file or you can copy and paste the values into a *.txt file first.

After uploading a file, the text can be edited by clicking the yellow marked file name in the text field. Editing the text is temporary and will not modify your original file.

To remove an uploaded file and go back to typing a value, click on the cross (x) next to the yellow text box. Another possibility to clear the uploaded values is by clicking the yellow marked text field and then pressing the Clear button on the bottom left corner of the Edit box. Using the Reset button will start a complete new search.

Copy to CQL editor

You can use the query builder to create complex queries without writing CQL. Any time a query is created in the querybuilder, it can be copied to the CQL editor, where you can further edit the query by hand. This will take you automatically to the Expert Search screen, after which you can start the search or adjust the query if desired.



Expert search

The Corpus Query Language (CQL) editor allows you to type your own CQL query, to copy your query into the query builder (in Advanced Search), to import a previously downloaded query and to upload a tab separated list of values to substitute for gap values (see below for further explanation).

CQL queries are expressions built up with the help of a few sequence operators and brackets from basic blocks enclosed by square brackets, in each of which one or more token attributes are specified.

In CQL, spaces only affect a search if they are included in quotes. Whether the search command is [word="man"] or [word = "man"] (or just "man") does not make any difference to the result. However, there is a difference between the queries [word="man"] and [word=" man"]. The first search results in over 17.000 hits, but the second one in zero!

Some examples:

- Simple: [word="man"], e.g. the attribute word matches the regular expression *man*; [word!="man"], e.g. the attribute word does **not** match the regular expression *man*. [word=".*man"] matches all word forms ending with *man*, including *man* itself. (Note that [word="*man"] will not give any results, because in Expert Search an asterisk is not a wildcard but a repetition operator.)
- Combination of attributes (combining operators are &, |, !), e.g. [word="huis|god|dier"] matches either the word *huis*, the word *god* or the word *dier*.
- The empty [] matches any token, e.g. [word="man"][]{3}[word="god"] matches a sequence of *man* followed by *god* with 3 arbitrary tokens in between.
- Operators |, & and parentheses () and the repetition operators (+, *, ? and {}) can be used to build complex sequence queries. Example: "goet" "wijf" | "quaet" "wijf", or even ("goet" "wijf" | "quaet" "wijf")+, matching any sequence of *goet wijf* or *quaet wijf*. Note that, while most queries up to this point could also have been constructed with the query builder, we really need the power of CQL from here on.

This short list does not cover all CQL features. For more detailed information on how to write CQL, please consult the short Appendix: Corpus Query Language, which contains further pointers.

Copy to query builder

When the query is relatively simple – like [word="schip"][word="den"] – it can also be imported into the querybuilder using the *Copy to query builder* button. This will take you automatically to the Advanced Search screen, after which you can start the search or adjust the query if desired.

A message will be displayed next to the button if the query couldn't be parsed.

Import query

If you have entered a search query, you can find it back by clicking the History button. On the right hand side you can select Download as file in the drop-down menu (default value is Search) and save the file. (For a more elaborate description of the History button see Simple Search.)

Previously saved queries can be used again by uploading them through the Import query button.

Gap filling

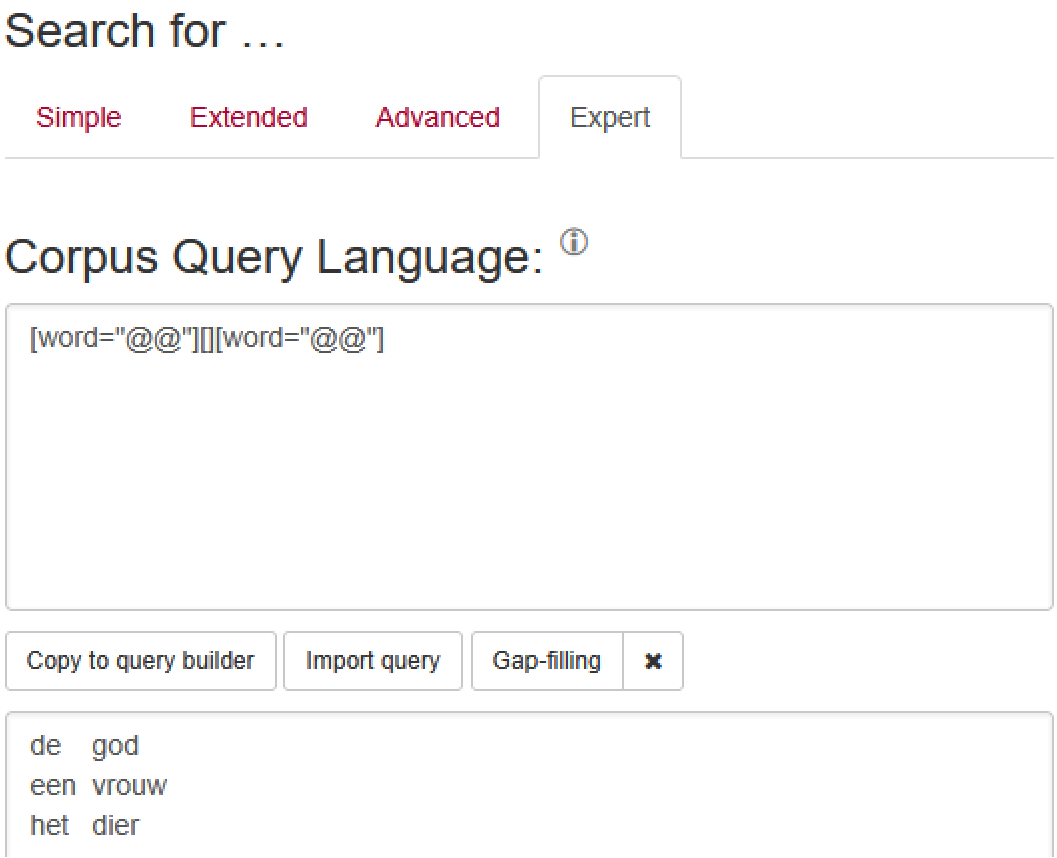
Use this button to upload a Tab Separated Values (TSV) file, which is a simple text format for storing data in a tabular structure. Each record in the table is one line of the text file. Each field value of a record is separated from the next by a tab character. It is also possible to upload a plain text file (.txt) that has the same properties.

A .tsv file or a comparable .txt file enables you to complete a query with marked gaps.

If, for instance, you are interested in the distribution of words that can be placed between two specific words you can create this query in the Corpus Query Language field:

[word="@@"][][word="@@"]

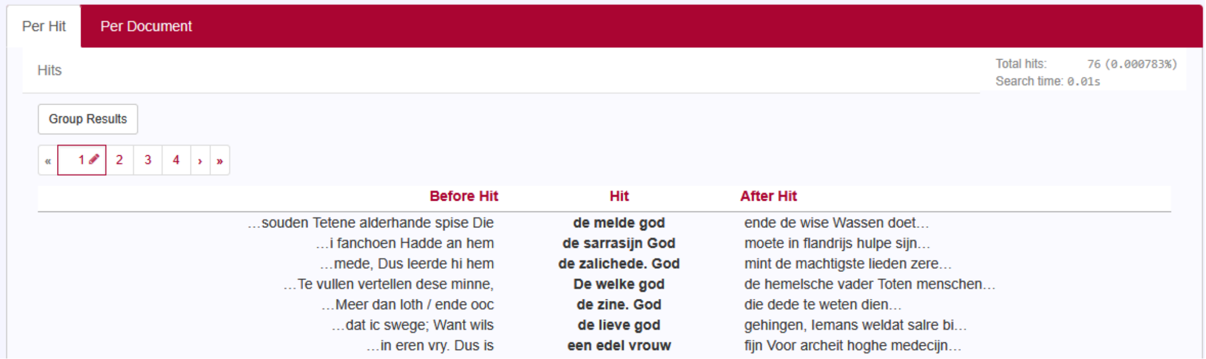
By clicking Gap-filling you can upload a file with a tab-separated list of values from your computer to substitute them for the gap values, i.e. the at signs (@@) in your query. After the upload your values will appear in a separate box:



[word="@@"][][word="@@"]

de	god
een	vrouw
het	dier

The values in the first column - *de, een, het* - will be entered at the position of the first gap (@@) and the values in the second column - *god, vrouw, dier* - at the position of the second gap. With these values, gap-filling yields the following results (titles are hidden):



Hits

Total hits: 76 (0.000783%)
Search time: 0.01s

Before Hit	Hit	After Hit
…souden Tetene alderhande spise Die	**de melde god**	ende de wise Wassen doet…
…i fanchoen Hadde an hem	**de sarrasijn God**	moete in flandrijs hulpe sijn…
…mede, Dus leerde hi hem	**de zalichede. God**	mint de machtigste lieden zere…
…Te vullen vertellen dese minne,	**De welke god**	de hemelsche vader Toten menschen…
…Meer dan loth / ende ooc	**de zine. God**	die dede te weten dien…
…dat ic swege; Want wils	**de lieve god**	gehingen, Iemans weldat salre bi…
…in eren vry. Dus is	**een edel vrouw**	fijn Voor archeit hoghe medecijn…

This mimics the functionality to upload a list of values in the Advanced Search interface.

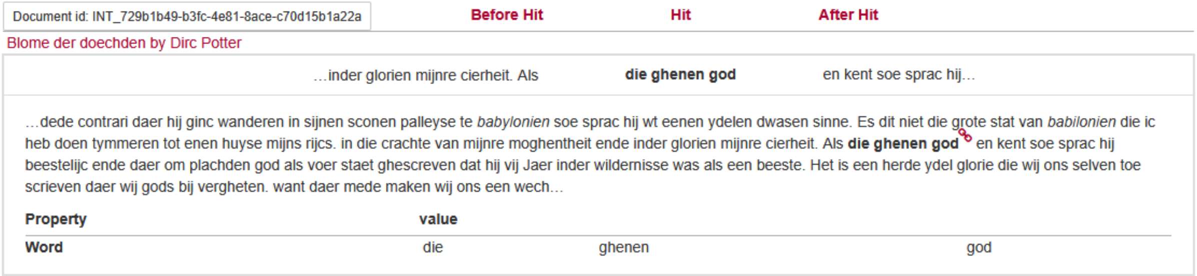
Please note that for this to work, you do need to enter @@ in the field where you want the substitution to take place. An empty field ([]) will match any term.

Viewing results

Results can be viewed in two ways: Per hit (hit is defined as one token or a group of tokens that matched the query), or Per document (each document listed contains at least one hit).

Per Hit view

Click a hit – i.e. a line with the bold word(s) in the column Hit – to display the properties and values of the hit (in the following example **die ghenen god)**. Click the hit again to close.



Document id: INT_729b1b49-b3fc-4e81-8ace-c70d15b1a22a

Blome der doechden by Dirc Potter

Before Hit	Hit	After Hit
…inder glorien mijnre cierheit. Als	**die ghenen god**	en kent soe sprac hij…

…dede contrari daer hij ginc wanderen in sijnen sconen palleyse te *babylonien* soe sprac hij wt eenen ydelen dwasen sinne. Es dit niet die grote stat van *babilonien* die ic heb doen tymmeren tot enen huyse mijns rijcs. in die crachte van mijnre moghentheit ende inder glorien mijnre cierheit. Als **die ghenen god** en kent soe sprac hij beestelijc ende daer om plachden god als voer staet ghescreven dat hij vij Jaer inder wildernisse was als een beeste. Het is een herde ydel glorie die wij ons selven toe scrieven daer wij gods bij vergheten. want daer mede maken wij ons een wech…

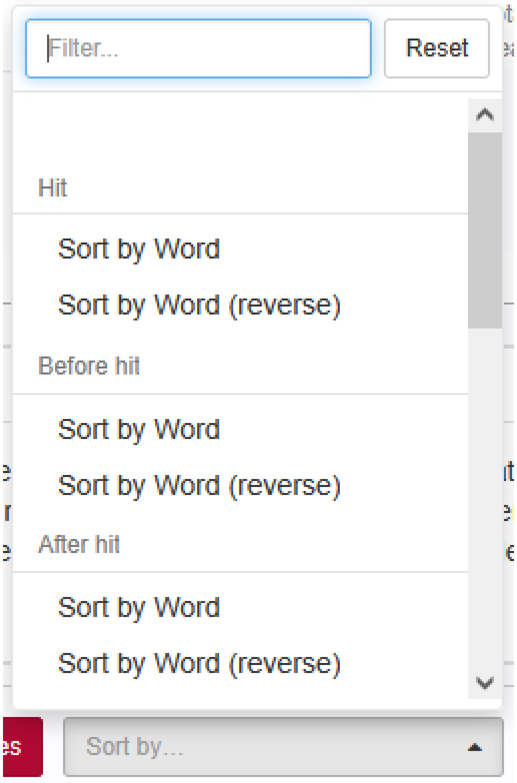
Property	value		
Word	die	ghenen	god

Hit rows are always preceded by a row containing the document title in which those hits occurred, in this case *Blome der doechden by Dirc Potter*. Document titles can be toggled on or off by using the Hide Titles (or Show Titles when titles are hidden) button at the bottom of the page. If you hover the mouse over the title, the identification number of the document appears, in this case: INT_729b1b49-b3fc-4e81-8ace-c70d15b1a22a.

Sorting results

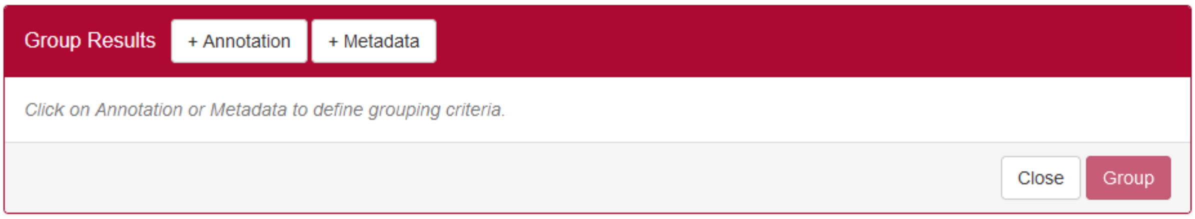
Click on any of the column headings to sort the hits on Words within that column, clicking again inverts the sorting.

You can also sort the results by means of the drop-down menu at the bottom of the page (Sort by…), which offers you the possibility to sort by various attributes for Hit, Before hit, After hit, Date, Text type and Title and Author.



Grouping results

It is possible to group the results by clicking on the button Group Results, after which the following menu appears:



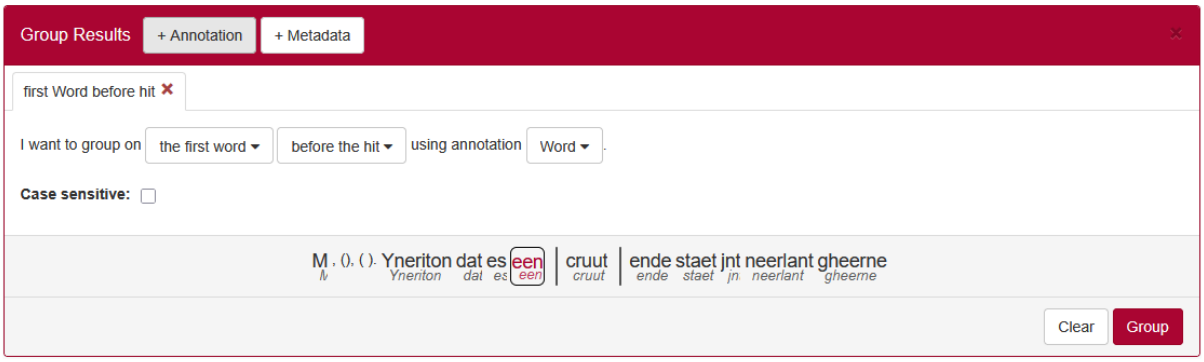
Click on Annotation or Metadata to define grouping criteria.

Results can be grouped by Annotation and by Metadata.

By clicking +Annotation you can group by the first word, by all words or by specific words, whether before the hit, within the hit or after the hit, and based on the annotation Word. When grouping by the first word or specific words, you can also group from the end of the hit. The default grouping is grouping all words within the hit using annotation Word. Clicking +Metadata allows you to group by metadata assigned to the document (Date, Text type, Title and Author).

By clicking the Case sensitive box it is possible to distinguish between case sensitive and case insensitive.

The example below is grouped by the first word before the hit. The example dynamically updates when the grouping options are changed.

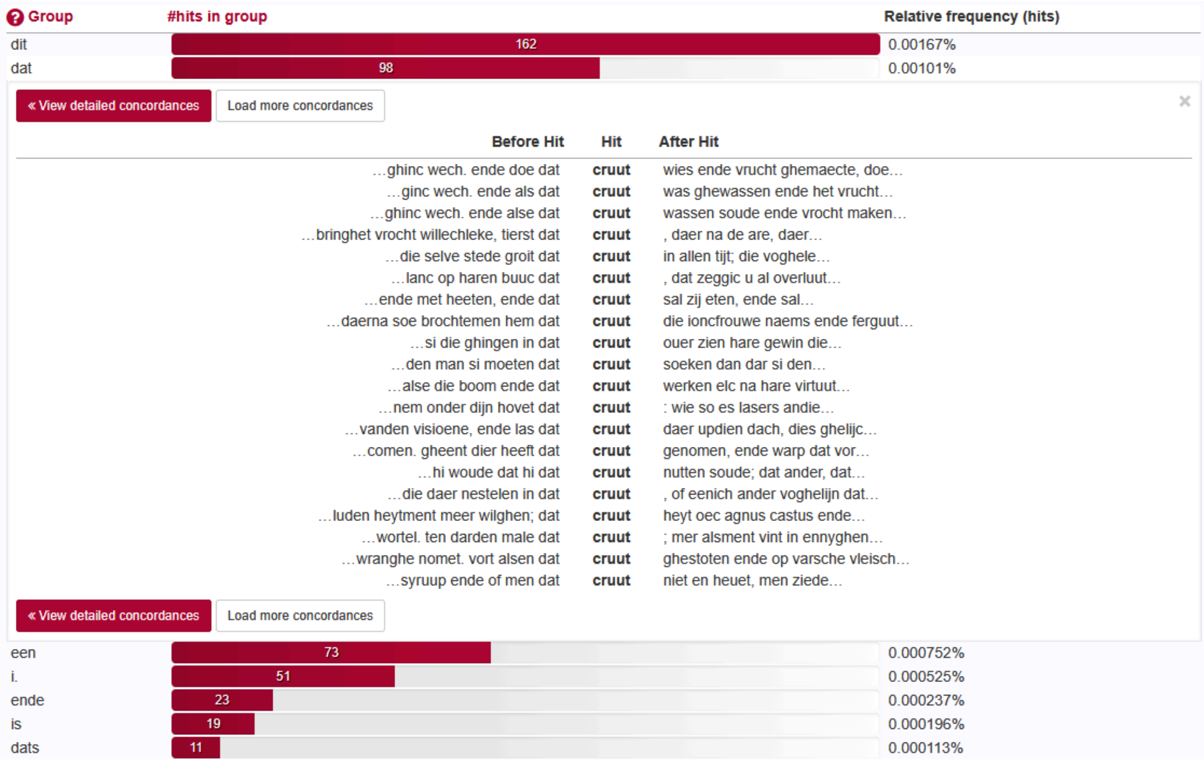


I want to group on the first word ▾ before the hit ▾ using annotation Word ▾.

Case sensitive: □

M . (). (). Yneriton dat es een | cruut | ende staet jnt neerlant gheerne

Click a group to show or hide hits within that group, as shown below. Click once more on the group to close it again. If more than twenty hits are found in a document, you can make them appear by clicking on Load more concordances.



Group	#hits in group	Relative frequency (hits)
dit	162	0.00167%
dat	98	0.00101%

Before Hit	Hit	After Hit
…ghinc wech. ende doe dat	cruut	wies ende vrucht ghemaecte, doe…
…ginc wech. ende als dat	cruut	was ghewassen ende het vrucht…
…ghinc wech. ende alse dat	cruut	wassen soude ende vrocht maken…
…bringhet vrocht willechleke, tierst dat	cruut	, daer na de are, daer…
…die selve stede groit dat	cruut	in allen tijt; die voghele…
…lanc op haren buuc dat	cruut	, dat zeggic u al overluut…
…ende met heeten, ende dat	cruut	sal zij eten, ende sal…
…daerna soe brochtemen hem dat	cruut	die ioncfrouwe naems ende ferguut…
…si die ghingen in dat	cruut	ouer zien hare gewin die…
…den man si moeten dat	cruut	soeken dan dar si den…
…alse die boom ende dat	cruut	werken elc na hare virtuut…
…nem onder dijn hovet dat	cruut	: wie so es lasers andie…
…vanden visioene, ende las dat	cruut	daer updien dach, dies ghelijc…
…comen. gheent dier heeft dat	cruut	genomen, ende warp dat vor…
…hi woude dat hi dat	cruut	nutten soude; dat ander, dat…
…die daer nestelen in dat	cruut	, of eenich ander voghelijn dat…
…luden heytment meer wilghen; dat	cruut	heyt oec agnus castus ende…
…wortel. ten darden male dat	cruut	; mer alsment vint in ennyghen…
…wranghe nomet. vort alsen dat	cruut	ghestoten ende op varsche vleisch…
…syruup ende of men dat	cruut	niet en heuet, men ziede…

Group	#hits in group	Relative frequency (hits)
een	73	0.000752%
i.	51	0.000525%
ende	23	0.000237%
is	19	0.000196%
dats	11	0.000113%

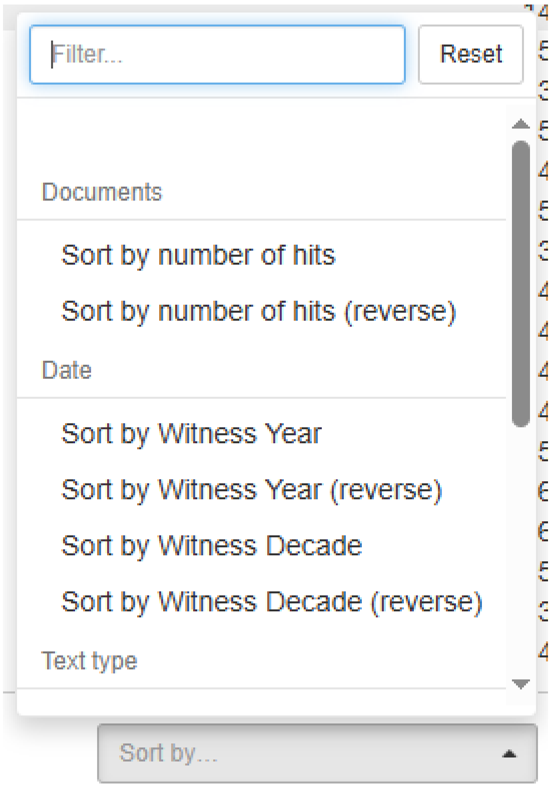
Click on View detailed concordances to go back to the normal hits view to see more detailed information for the hits in this group. The button Go back to grouped results brings you back to the list of groups.

Per Document view

Sorting results

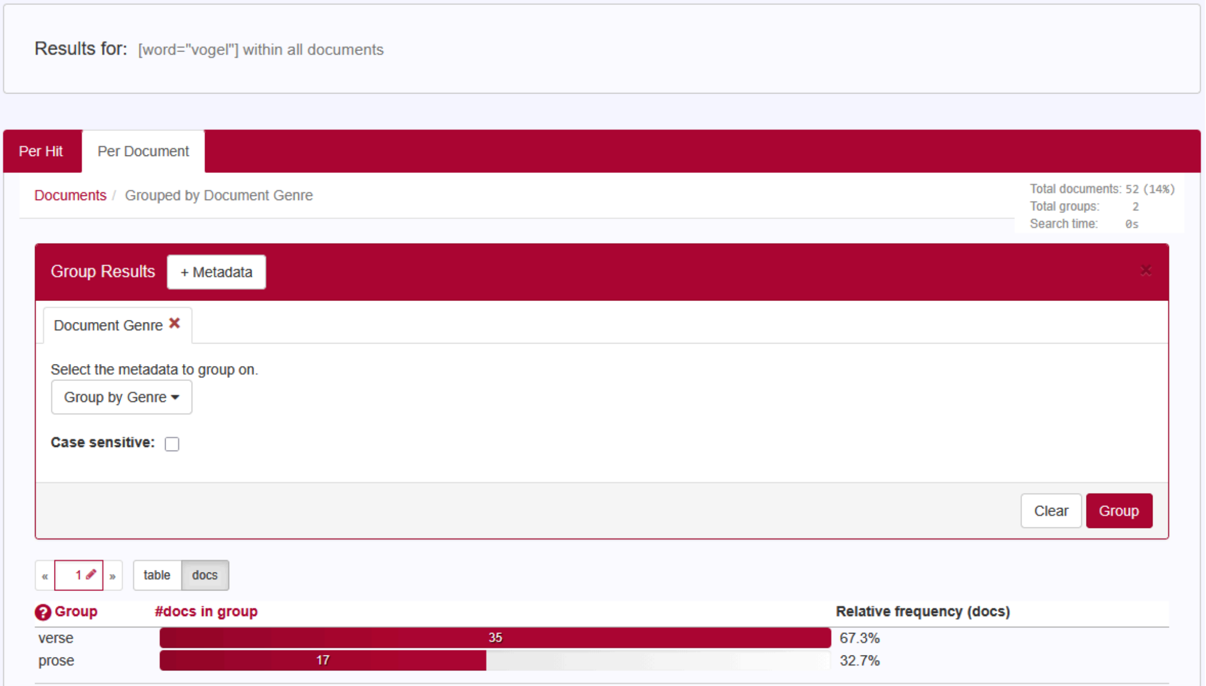
Click on any of the column headings to sort the documents by Document (name), Witness Year or Hits within that column, clicking again inverts the sorting.

You can also sort the results by means of the drop-down menu at the bottom of the page (Sort by…), which offers you the possibility to sort by various attributes such as Hit (Documents), Genre (Text type) and Title (Title and author).



Grouping results

Results Per Document can be grouped by metadata assigned to the document (Date, Text type and Title and author). The example below shows all documents in which the Word *vogel* occurs grouped by genre.



Results for: [word="vogel"] within all documents

Total documents: 52 (14%)
Total groups: 2
Search time: 0s

Select the metadata to group on.

Case sensitive: □

Group	#docs in group	Relative frequency (docs)
verse	35	67.3%
prose	17	32.7%

Exporting results

The search results - both Per hit as Per document - can be exported by using the Export or the Export for Excel button at the bottom right of the page. The first button transfers the search results - including all metadata - to a Comma-Separated Values-file. These CSV-files consist only of text data, which makes it easy to implement (read and/or write) them into a spreadsheet or database program. The second button offers the possibility to export the results - including all metadata - to a CSV-file for use with Excel.

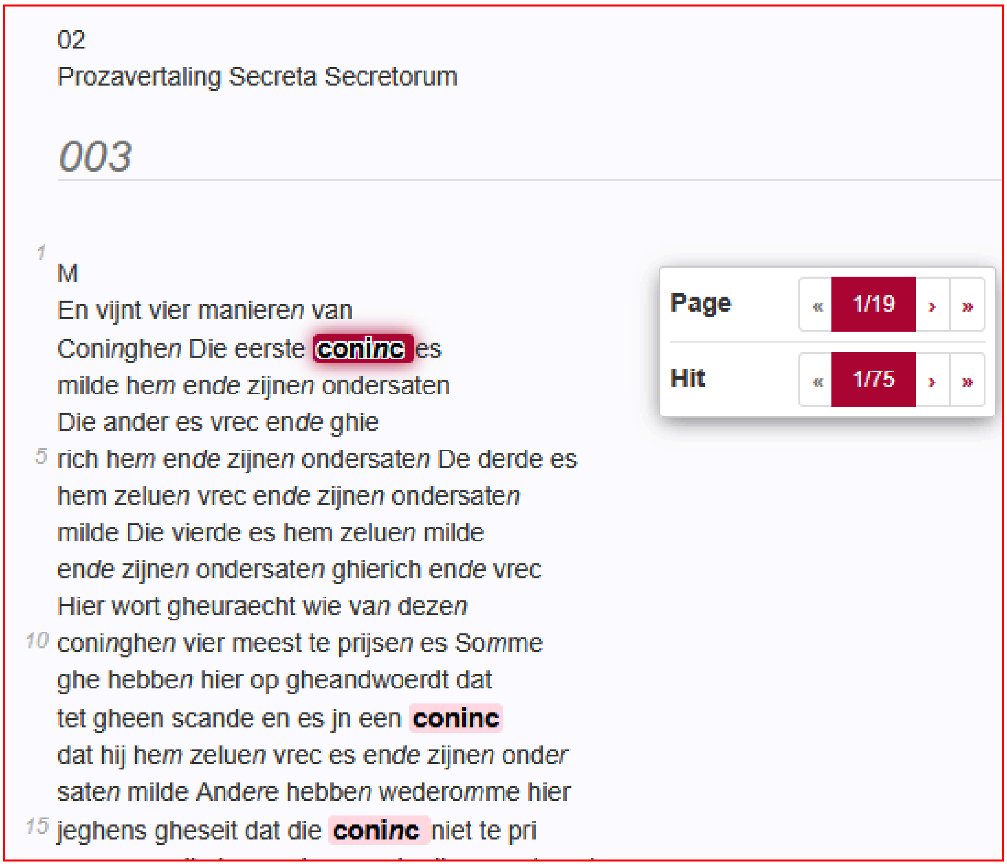
Grouped results can be exported in the same way. However, if you would like to have the metadata with each concordance of a group, you must first click on the red bar of a specific group and then on View detailed concordances. The results you then see can be exported by the use of the Export buttons. This operation must be carried out for each individual group you wish to export.

Information about a document

Click on a document title or the chain icon in the per hit view to open this document in a new window: the Content window.

Content

Hits from the current query will be highlighted in bold in the opened document. In the case of several hits only the current hit will also appear in shadow (such as *coninc* in the example below). You can navigate from one hit to another by using the arrows at the Hits button (this button can be dragged around), and you can also browse through the pages if the document consists of more than one page.



02
Prozavertaling Secreta Secretorum

003

M
En vijnt vier manieren van
Coninghen Die eerste **coninc** es
milde hem ende zijnen ondersaten
Die ander es vrec ende ghie
rich hem ende zijnen ondersaten De derde es
hem zeluen vrec ende zijnen ondersaten
milde Die vierde es hem zeluen milde
ende zijnen ondersaten ghierich ende vrec
Hier wort gheuraecht wie van dezen
coninghen vier meest te prijsen es Somme
ghe hebben hier op gheandwoerdt dat
tet gheen scande en es jn een **coninc**
dat hij hem zeluen vrec es ende zijnen onder
saten milde Andere hebben wederomme hier
jeghens gheseit dat die **coninc** niet te pri

Metadata

In the Metadata tab all metadata properties of the document are displayed.

Statistics

The Statistics tab shows several document statistics: the number of Tokens, Types (unique word forms) and the Type/Token ratio. It is possible to print or to download these statistics via the menu symbol right of the title Vocabulary Growth.

Exploring the corpus

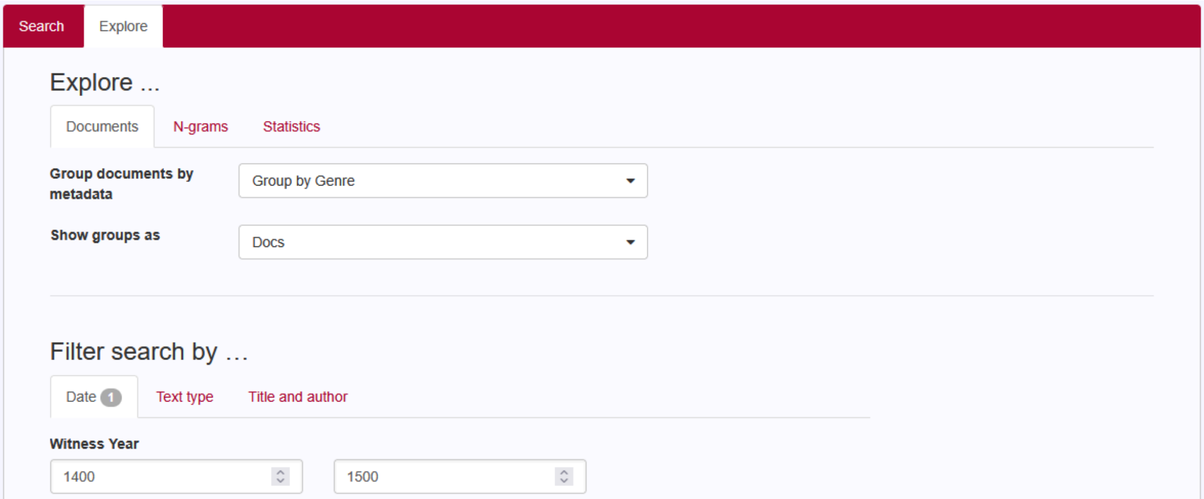
The Explore tab has three subdivisions: Documents, N-grams and Statistics.

Documents

This subtab allows you to investigate the documents. It consists of two drop-down menus to specify the grouping of the metadata and to specify the way the groups are to be shown.

A simple example: suppose we want to obtain information about the Genre of the documents from 1400 till 1500 within the *Corpus Middelnederlands*.

- In the Group documents by metadata drop-down menu, choose Group by Genre (Text type)
- In Show groups as, select *Docs*
- In the metadata search form (Filter search by), fill in at Witness Year *1400* and *1500*
- Press Search



Explore ...

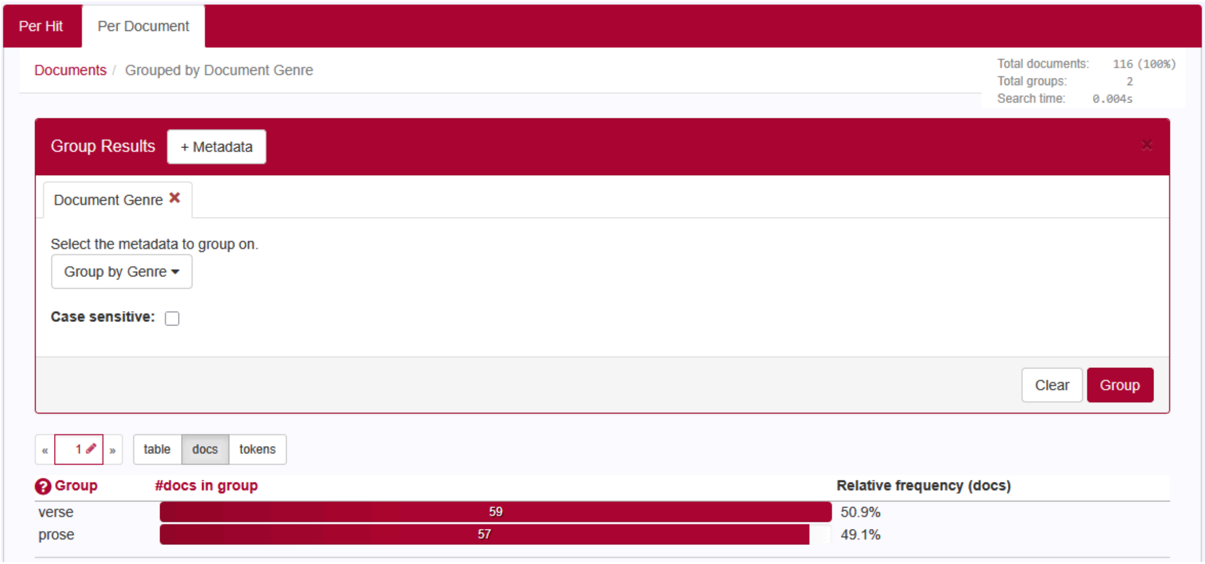
Group documents by metadata: Group by Genre

Show groups as: Docs

Filter search by ...

Witness Year: 1400 – 1500

This will be the result:



Documents / Grouped by Document Genre

Total documents: 116 (100%)
Total groups: 2
Search time: 0.004s

Select the metadata to group on.

Case sensitive: □

Group	#docs in group	Relative frequency (docs)
verse	59	50.9%
prose	57	49.1%

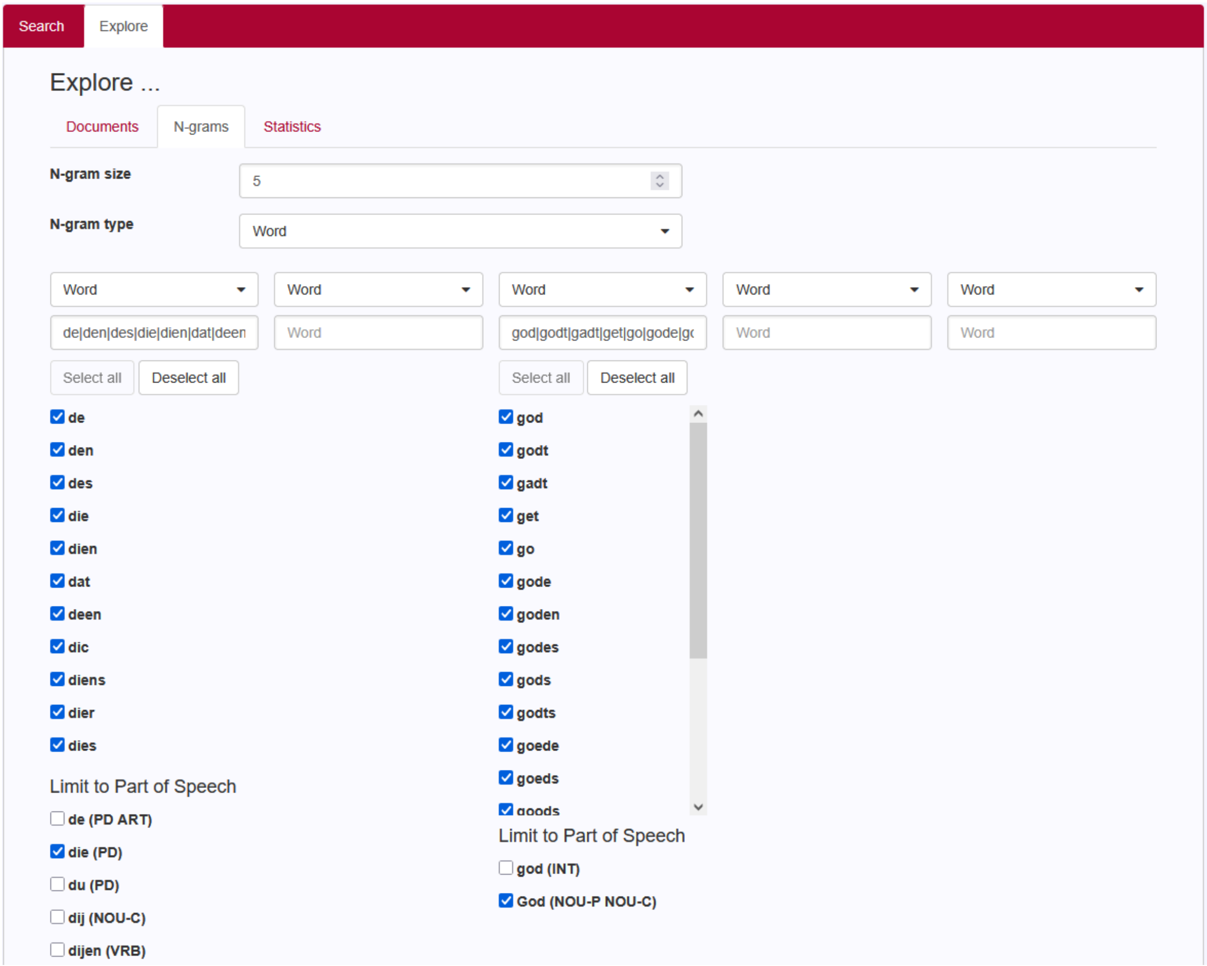
N-grams

An *N-gram* is a sequence of *N* items. This option will list the frequency of different N-grams in a (sub-)corpus.

Options

- *N-gram size*: the length of the sequence (a number from 1 to 5; default setting is 5).
- *N-gram type*: the attribute to search for. You can choose: Word (i.e. word form). If you do not specify the search term a series of arbitrary words equal to the n-gram size will be searched for.
- It is also possible to restrict to, for instance, n-grams with some slots already specified, as is shown in the following example. After entering a search term, a spinner briefly appears on the right side of the search bar. Based on the keyed in word, suggestions are given of possible variants of spelling and/or form from the GiGaNT-lexicon and of parts of speech. By clicking on ‘Select all’ all forms belonging to a GiGaNT lemma are added.
- By using the Filter search by... you can create a subcorpus within the *Corpus Middelnederlands* for specific metadata.

Example



Explore ...

N-gram size: 5

N-gram type: Word

Word	Word	Word	Word	Word
de|den|des|die|dien|dat|deen	Word	god|godt|gadt|get|go|gode|gc	Word	Word

☑ de
☑ den
☑ des
☑ die
☑ dien
☑ dat
☑ deen
☑ dic
☑ diens
☑ dier
☑ dies

Limit to Part of Speech

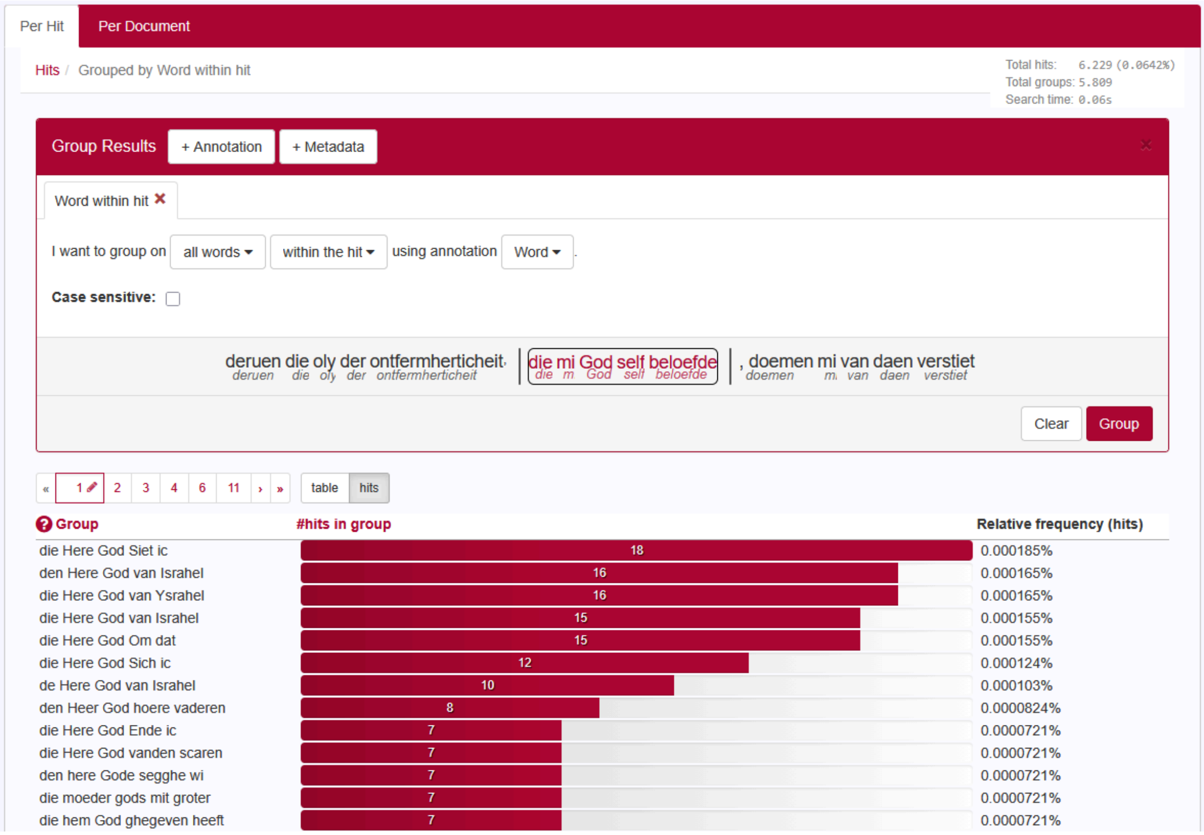
□ de (PD ART)
☑ die (PD)
□ du (PD)
□ dij (NOU-C)
□ dijen (VRB)

☑ god
☑ godt
☑ gadt
☑ get
☑ go
☑ gode
☑ goden
☑ godes
☑ gods
☑ godts
☑ goede
☑ goeds
☑ goods

Limit to Part of Speech

□ god (INT)
☑ God (NOU-P NOU-C)

Within all the documents of the *Corpus Middelnederlands*, you will find 6229 occurrences of this so-called 5-gram (choose the option ‘Select all’ for both words and limit your search to Part of Speech *die* (PD) respectively *God* (NOU-C NOU-P)).



Total hits: 6.229 (0.0642%)
Total groups: 5.809
Search time: 0.06s

I want to group on all words within the hit using annotation Word.

Case sensitive: □

deruen die oly der ontfermherticheit. | die mi God self beloefde | , doemen mi van daen verstiet

Group	#hits in group	Relative frequency (hits)
die Here God Siet ic	18	0.000185%
den Here God van Israhel	16	0.000165%
die Here God van Ysrahel	16	0.000165%
die Here God van Israhel	15	0.000155%
die Here God Om dat	15	0.000155%
die Here God Sich ic	12	0.000124%
de Here God van Israhel	10	0.000103%
den Heer God hoere vaderen	8	0.0000824%
die Here God Ende ic	7	0.0000721%
die Here God vanden scaren	7	0.0000721%
den here Gode segghe wi	7	0.0000721%
die moeder gods mit groter	7	0.0000721%
die hem God ghegeven heeft	7	0.0000721%

Statistics (frequency lists)

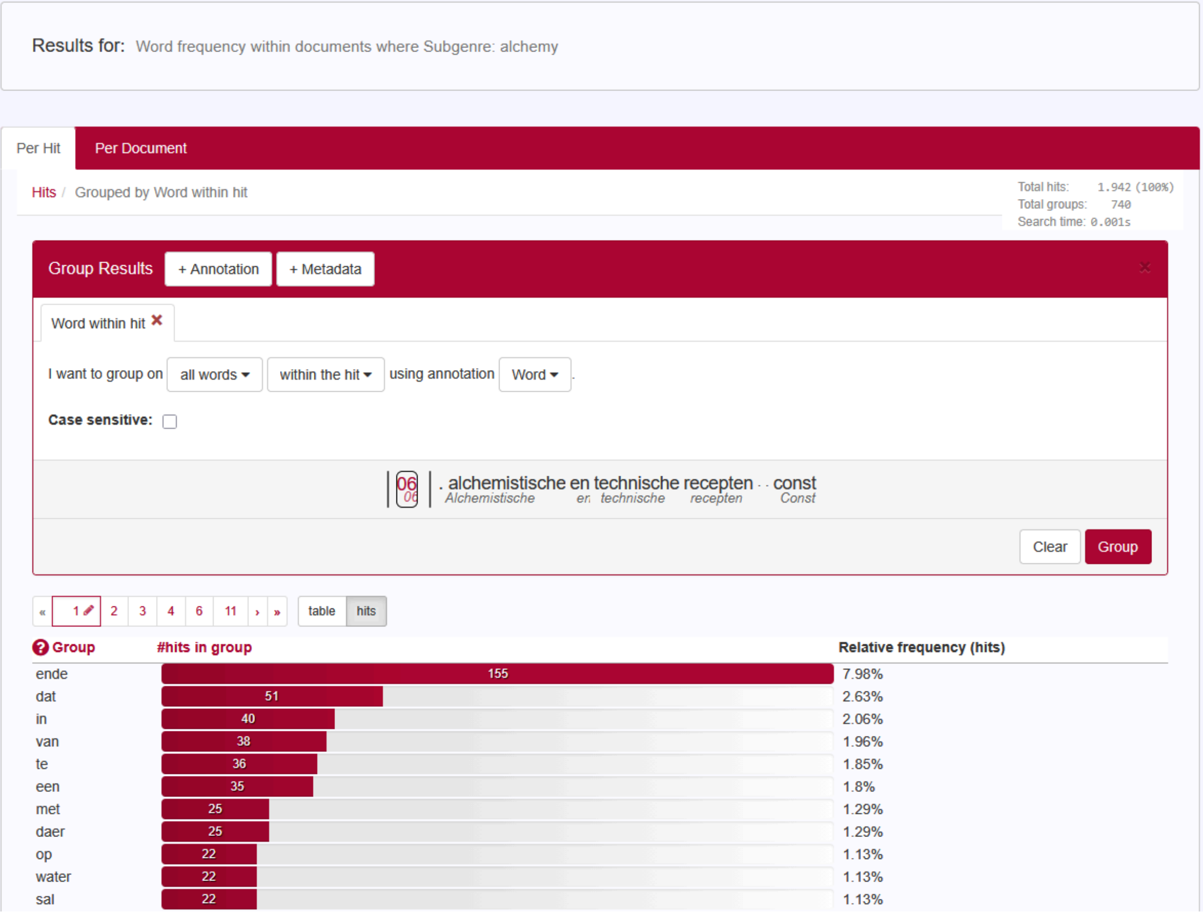
Here, you can produce frequency lists for a subcorpus. It is rather similar to the previous option, but restricted to 1-grams.

Options

- *Frequency list type:* in this corpus, it is only possible to create frequency lists of Words (i.e. word forms).
- By using the Filter search by... you can create a subcorpus within the *Corpus Middelnederlands* for specific metadata.

Example

It is possible to determine the use of the most frequently used Middle Dutch words in texts about alchemy by searching for Frequency list type Word and by filtering search by Subgenre: *alchemy*. This results in:



Results for: Word frequency within documents where Subgenre: alchemy

Total hits: 1.942 (100%)
Total groups: 740
Search time: 0.001s

I want to group on all words ▾ within the hit ▾ using annotation Word ▾.

Case sensitive: □

06 . alchemistische en technische recepten · · const
06 Alchemistische en technische recepten Const

Group	#hits in group	Relative frequency (hits)
ende	155	7.98%
dat	51	2.63%
in	40	2.06%
van	38	1.96%
te	36	1.85%
een	35	1.8%
met	25	1.29%
daer	25	1.29%
op	22	1.13%
water	22	1.13%
sal	22	1.13%

Appendix: Corpus Query Language

BlackLab supports Corpus Query Language, a full-featured query language introduced by the IMS Corpus WorkBench (CWB) and also supported by the Lexicom Sketch Engine. It is a standard and powerful way of searching corpus.

The basics of Corpus Query Language is the same in all three projects, but there are a few minor differences in some of the more advanced features, as well as some features that are exclusive to some projects. For most queries however, this will not be an issue.

This page will introduce the query language and show all features that BlackLab supports. If you want to learn even more about CQL, see CWB CQP Query Language Tutorial and Sketch Engine Corpus Query Language.

CQL support

For those who already know CQL, here's a quick overview of the extent of BlackLab's support for this query language. If there is a feature we don't support, yet is important to you, please let us know. If it's quick to add, we may be able to help you out.

Supported features

BlackLab currently supports (arguably) most of the important features of Corpus Query Language:

- Matching on token annotations (also called properties or attributes), using regular expressions and =, !=, !. Example: [word="bank"] (or just "bank")
- Case/accent-sensitive matching. Note that, unlike in CWB, case-INsensitive matching is currently the default. To explicitly match case/accent-insensitivity, use "(?i)...". Example: "(?-i)Mr\." "(?-i)Banks"
- Combining criteria using &, | and !. Parentheses can also be used for grouping. Example: [lemma="bank" & pos="V"]
- Match-all pattern [] matches any token. Example: "a" [] "day"
- Regular expression operators +, *, ?, {n}, {n,m} at the token level. Example: [pos="AA"]+
- Sequences of token constraints. Example: [pos="AA"] "cow"
- Operators |, & and parentheses can be used to build complex sequence queries. Example: "happy" "dog" | "sad" cat"
- Querying with tag positions using e.g. <s> (start of sentence), </s> (end of sentence), <s/> (whole sentence) or <s> ... </s> (equivalent to <s/> containing ...). Example: <s> "The" . XML attribute values may be used as well, e.g. <ne type="PERS"/> ("named entities that are persons").
- Using within and containing operators to find hits inside another set of hits. Example: "you" "are" within <s/>
- Using an anchor to capture a token position. Example: "big" A:[]. Captured matches can be used in global constraints (see next item) or processed separately later (using the Java interface; capture information is not yet returned by BlackLab Server). Note that BlackLab can actually capture entire groups of tokens as well, similarly to regular expression engines.

- Global constraints on captured tokens, such as requiring them to contain the same word. Example: "big" A:[] "or" "small" B:[] :: A.word = B.word

See below for features not in this list that may be added soon, and let us know if you want a particular feature to be added.

Differences from CWB

BlackLab's CQL syntax and behaviour differs in a few small ways from CWBs. In future, we'll aim towards greater compliance with CWB's de-facto standard (with some extra features and conveniences).

For now, here's what you should know:

- Case-insensitive search is currently the default in BlackLab, although you can change this if you wish. CWB and Sketch Engine use case-sensitive search as the default. We may change our default in a future major version.
  If you want to switch case-/diacritics-sensitivity, use "(?-i).." (case-sensitive) or "(?i).." (case-insensitive, usually the default). CWBs %cd flags for setting case/diacritics-sensitivity are not (yet) supported, but will be added.
- If you want to match a string literally, not as a regular expression, use backslash escaping: "e\.g\.". %l for literal matching is not yet supported, but will be added.
- BlackLab supports result set manipulation such as: sorting (including on specific context words), grouping/frequency distribution, subsets, sampling, setting context size, etc. However, these are supported through the REST and Java APIs, not through a command interface like in CWB. See BlackLab Server overview).
- Querying XML elements and attributes looks natural in BlackLab: <s/> means "sentences", <s> means "starts of sentences", <s type='A'> means "sentence tags with a type attribute with value A". This natural syntax differs from CWBs in some places, however, particularly when matching XML attributes. While we believe our syntax is the superior one, we may add support for the CWB syntax as an alternative.
  We only support literal matching of XML attributes at the moment, but this will be expanded to full regex matching.
- In global constraints (expressions occurring after ::), only literal matching (no regex matching) is currently supported. Regex matching will be added soon. For now, instead of A:[] "dog" :: A.word = "happy|sad", use "happy|sad" "dog".
- To expand your query to return whole sentences, use <s/> containing (...). We don't yet support CWBs expand to, expand left to, etc., but may add this in the future.
- The implication operator -> is currently only supported in global constraints (expressions after the :: operator), not in regular token constraints. We may add this if there's demand for it.
- We don't support the @ anchor and corresponding target label; use a named anchor instead. If someone makes a good case for it, we will consider adding this feature.
- backreferences to anchors only work in global constraints, so this doesn't work: A:[] [] [word = A.word]. Instead, use something like: A:[] [] B:[] :: A.word = B.word. We hope to add support for these in the near future, but our matching approach may not allow full support for this in all cases.

(Currently) unsupported features

The following features are not (yet) supported:

- intersection, union and difference operators. These three operators will be added in the future. For now, the first two can be achieved using & and | at the sequence level, e.g. "double" [] & [] "trouble" to match the intersection of these queries, i.e. "double trouble" and "happy" "dog" | "sad "cat" to match the union of "happy dog" and "sad cat".
- _ meaning "the current token" in token constraints. We will add this soon.
- lbound, rbound functions to get the edge of a region. We will probably add these.
- distance, distabs functions and match, matchend anchor points (sometimes used in global constraints). We will see about adding these.
- using an XML element name to mean 'token is contained within', like [(pos = "N") & !np] meaning "noun NOT inside in an tag". We will see about adding these.
- a number of less well-known features. If people ask, we will consider adding them.

Using Corpus Query Language

Matching tokens

Corpus Query Language is a way to specify a "pattern" of tokens (i.e. words) you're looking for. A simple pattern is this one:

```
[word="man"]
```

This simply searches for all occurrences of the word "man". If your corpus includes the per-word properties lemma (i.e. headword) and pos (part-of-speech, i.e. noun, verb, etc.), you can query those as well. For example, to find a form of word "search" used as a noun, use this query:

```
[lemma="search" & pos="NOU-C"]
```

This query would match "search" and "searches" where used as a noun. (Of course, your data may contain slightly different part-of-speech tags.)

The first query could be written even simpler without brackets, because "word" is the default property:

```
"man"
```

You can use the "does not equal" operator (!=) to search for all words except nouns:

```
[pos != "NOU-C"]
```

The strings between quotes can also contain wildcards, of sorts. To be precise, they are regular expressions, which provide a flexible way of matching strings of text. For example, to find "man" or "woman", use:

```
"(wo)?man"
```

And to find lemmata starting with "under", use:

```
[lemma="under.\*"]
```

Explaining regular expression syntax is beyond the scope of this document, but for a complete overview, see here.

Sequences

Corpus Query Language allows you to search for sequences of words as well (i.e. phrase searches, but with many more possibilities). To search for the phrase "the tall man", use this query:

```
"the" "tall" "man"
```

It might seem a bit clunky to separately quote each word, but this allows us the flexibility to specify exactly what kinds of words we're looking for. For example, if you want to know all single adjectives used with man (not just "tall"), use this:

```
"an?|the" [pos="AA"] "man"
```

This would also match "a wise man", "an important man", "the foolish man", etc.

Regular expression operators on tokens

Corpus Query Language really starts to shine when you use the regular expression operators on whole tokens as well. If we want to see not just single adjectives applied to "man", but multiple as well:

```
"an?|the" [pos="AA"]+ "man"
```

This query matches "a little green man", for example. The plus sign after [pos="AA"] says that the preceding part should occur one or more times (similarly, * means "zero or more times", and ? means "zero or one time").

If you only want matches with two or three adjectives, you can specify that too:

```
"an?|the" [pos="AA"]{2,3} "man"
```

Or, for two or more adjectives:

```
"an?|the" [pos="AA"]{2,} "man"
```

You can group sequences of tokens with parentheses and apply operators to the whole group as well. To search for a sequence of nouns, each optionally preceded by an article:

```
("an?|the"? [pos="NOU-C"])+
```

This would, for example, match the well-known palindrome "a man, a plan, a canal: Panama!"

Punctuation

In BlackLab, punctuation tends to not be indexed as a separate token, but as a property of a word token - CWB and Sketch Engine on the other hand tend to index punctuation as a separate token instead. You certainly could choose to index punctuation as a separate token in BlackLab, by the way -- it's just not commonly done. Both approaches have their advantages and disadvantages, and of course the choice affects how you write your queries.

It is possible to search for punctuation marks. E.g. to find occurrences of the word "want" preceded by a comma use the following query:

```
[punctBefore="," & word="want"]
```

To find occurrences of the lemma "krant" that are followed by an exclamation mark, use:

```
[lemma="krant" & punctAfter="!"]
```

Some punctuation marks have a special function in regular expressions and therefore must be preceded by a backslash (\) when used in queries. For instance, to search for a period (`.`) after the word **"geweest"**, use:

```
[word="sentence" & punctAfter="\."]
```

Case- and diacritics-sensitivity

CWB and Sketch Engine both default to (case- and diacritics-)sensitive search. That is, they exactly match upper- and lowercase letters in your query, plus any accented letters in the query as well. BlackLab, on the contrary, defaults to *IN*sensitive search (although this default can be changed if you like). To match a pattern sensitively, prefix it with "(?-i)":

```
"(?-i)Panama"
```

If you've changed the default search to sensitive, but you wish to match a pattern in your query insensitively, prefix it with "(?i)":

```
[pos="(?i)NOU-C"]
```

Although BlackLab is capable of setting case- and diacritics-sensitivity separately, it is not yet possible from Corpus Query Language. We may add this capability if requested.

Matching XML elements

Corpus Query Language allows you to find text in relation to XML elements that occur in it. For example, if your data contains sentence tags, you could look for sentences starting with "the":

```
<s>"the"
```

Similarly, to find sentences ending in "that", you would use:

```
"that"</s>
```

You can also search for words occurring inside a specific element. Say you've run named entity recognition on your data and all person names are surrounded with <person>...</person> tags. To find the word "baker" as part of a person's name, use:

```
"baker" within <person/>
```

Note the forward slash at the end of the tag. This way of referring to the element means "the whole element". Compare this to <person>, which means "the element's open tag", and </person>, which means "the element's close tag".

The above query will just match the word "baker" as part of a person's name. But you're likely more interested in the entire name that contains the word "baker". So, to find those full names, use:

```
<person/> containing "baker"
```

Or, if you simply want to find all persons, use:

```
<person/>
```

As you can see, the XML element reference is just another query that yields a number of matches. So as you might have guessed, you can use "within" and "containing" with any other query as well. For example:

```
([pos="AA"]+ containing "tall") "man"
```

will find adjectives applied to man, where one of those adjectives is "tall".

Labeling tokens, capturing groups

Just like in regular expressions, it is possible to "capture" part of the match for your query in a "group".

CWB and Sketch Engine offer similar functionality, but instead of capturing part of the query, they label a single token. BlackLab's functionality is very similar but can capture a number of tokens as well. For example:

```
"an?|the" Adjectives:[pos="AA"]+ "man"
```

This will capture the adjectives found for each match in a captured group named "Adjectives". BlackLab also supports numbered groups:

```
"an?|the" 1:[pos="AA"]+ "man"
```

Global constraints

If you tag certain tokens with labels, you can also apply "global constraints" on these tokens. This is a way of relating different tokens to one another, for example requiring that they correspond to the same word:

```
A:[] "by" B:[] :: A.word = B.word
```

This would match "day by day", "step by step", etc.