SimSage
Search less; find more.

Purpose

This document explains the Super Search Syntax used by SimSage to find items across documents and their metadata and exposes the true power of the SimSage engine. First we look at some examples. Then we look at the SimSage Text to SuperSearch processor to see how ordinary user text is transformed, invisible to the user, into Super Search Syntax. Finally, we look at the syntax view of the query syntax.

10 Super Search examples

1. The simplest query
(word(test, body)) (equivalent to (word(test)) since the default metadata marker is “body”)

meaning
Search for the word “test” and all its possible relationships in the body text of all documents ingested by SimSage, to which the user has access.

2. A more exact search
(word(test, body, exact)) (equivalent to (word(test, exact)) for the same reason as above.)

meaning
Search for the exact word “test” and none of its relationships, this will include stems of test, like “testing, tested, tests, since you did not specify

3. Even more exact
(word(test,n,exact))

meaning
Search for the exact word “test” that is a noun. The system will only search for “test” and “tests” in the context of nouns in the body of all SimSage documents accessible to this user.

4. Words joined
(word(market) and word(researchers) and word(James))

meaning
Search for three keywords and all their relationships within the document contents of all SimSage documents accessible to this user within the maximum default distance of 40 tokens (words and punctuation markers) between all words.

NB. some concepts can be defined as being more than one word. For instance, it could be that “market researcher” is a single concept with its own relationships. In that case, the word is not split and is considered a single word in a search. Use two words if you’re not sure, SimSage will deal with that case too.

5. Mixed metadata query
(word(market) and word(rock,title))

meaning
This query searches for “market” and its relationships in the body of all documents, AND must also have the word “rock” (or any of its relationships) in the title of these documents.

6. Searches within searches
((word(market) and word(research)) sub (word(john)))

meaning
Search for “market and research” in the body of all documents in SimSage, and then perform a second search within the set of documents found (a search within a search, or a SUB search) for the word “john”. This is fundamentally different from *(word(market) and word(research) and word(john))*. SimSage throws away its distance constraints when doing a SUB search. The second search is scored on its proximity to the first results but does not have to be near the result set to succeed.

NB. SUB searches can only occur at the top level of a search. You cannot mix AND with SUB (i.e. an AND must always reside inside a set of SUB searches).

NB. it is always preferable to use extra brackets ( ) when in doubt. Such brackets can avoid any ambiguity in constructing / translating the query from text to an internal super-search expression.

7. Entity queries
(entity(person))

meaning
Find all entities (semantically marked words in SimSage) of type “person” in the body of all documents indexed. SimSage has a build in set of 100,000 people’s names, and 2M+ cities (entity(city)) and will mark items as such when seen in text. Other entities out of the box include are shown in a table in this document.

8. And NOT
(word(rock) and not word(mark))

meaning
Look for “rock” and all its relationships in the body of documents, but filter out any document that has the word “mark” or any of its relationships.

9. Source filtering
(word(market) and word(forces) and source(1,2,3))

meaning

Look for the words “market” and “forces” and all their relationships in close proximity in the bodies of all documents but only in the sources identified by IDs 1, 2 and/or 3. All sources in SimSage have a unique ID number, starting at 1 and incrementing thereafter. These numbers can be found through the admin UX.

10. Query
(word(market) and meta(categorization, top-secret))

meaning
Perform a search for “market” and its relationship in the body of all documents, and filter all documents on their “category” metadata field (documents that do not have this field are ignored / not included). This “category” field must have the value “top secret” to be included.

NB. Only document body items, and metadata document titles use relationships. This can be configured, but is the default of SimSage. Other metadata items, no matter what or where, do not get relationship expansions.

SimSage
Search less; find more.

Text to SuperSearch

Text to super-search is a subsystem of SimSage that translates ordinary user text typed in the search box into complex super-search expressions.

This sub-system is entirely configurable, and we show the default values shipped with SimSage as an example here.

User words	Super Search equivalent	AND or SUB search?
most recent, recently, newest, latest, sort by date, sort by time, sort by most recent	sort()	AND
credit card, credit-card, credit card number	entity(credit-card)	SUB
who, what person, which person, what people, a person	entity(person)	SUB
when, what date, the date	entity(date)	SUB
how much, the cost, the price	entity(money)	SUB
how many, it weigh, how heavy	entity(number)	SUB
national insurance number, national insurance, nin, nino	entity(nin)	SUB
law, law firm, law-firm, firm	entity(law-firm)	SUB
where, what town, what city	entity(city)	SUB
what country, which country	entity(country)	SUB
email, email address	entity(email)	SUB
url, uri, web address, http address, https address	entity(url)	SUB
similar, identical, group, group by similar, identical group	group()	AND
web documents, html file, web doc, web file	meta(document-type,html)	SUB
pdf file, pdf, adobe acrobat file, adobe file, portable document format, pdf document	meta(document-type,pdf)	SUB
office document, office file	meta(document-type,docx) or meta(document-type,doc) or meta(document-type,xls) or meta(document-type,xlsx) or meta(document-type,ppt) or	AND

	meta(document-type,pptx)	
word file, word document	meta(document-type,docx) or meta(document-type,doc)	AND
excel file, excel document, spreadsheet	meta(document-type,xls) or meta(document-type,xlsx)	AND
powerpoint file, powerpoint file, presentation	meta(document-type,ppt) or meta(document-type,pptx)	AND
image, image file	meta(document-type,jpg) or meta(document-type,jpeg) or meta(document-type,png)	AND
jpeg, jpg, jpeg file, jpg file	meta(document-type,jpg) or meta(document-type,jpeg)	AND

Example 1
An example to explain the above table. The user types “most recent PDF files”

“most recent” is associated with “sort()”, and “PDF files” is associated with “meta(document-type,pdf)”. The PDF files is of type SUB, and the “most recent” is of type “AND”.

Reading from left to right we can build the following expression:

(AND sort() SUB meta(document-type,pdf))

The first AND is not used and is “thrown away” by the parser’s logic, this leaving

(sort() SUB meta(document-type,pdf))

meaning
The user wants to see all documents of type PDF, ordered by newest first.

Example 2
The user types “latest similar market forces”

The query parser detects “latest” being associated with “sort()” AND, and “similar” with AND “group()”. The words “market forces” match nothing and are assumed to be keywords for the default “body” metadata.

((word(market) and word(forces)) and sort() and group())

meaning
Do a search for “market” and “forces” (and all their relationships) near each other in all document contents, group any documents that are similar to each other, and sort the results by newest documents first.

SimSage
Search less; find more.

List of predefined semantics

Here is a list of predefined semantics in SimSage that can be searched on using the “entity(…)” super-search query.

Entity	Description	Example
nin	National Insurance Number, the British equivalent of Social Security Number	SX123456G
credit-card	A credit card number	4545-4545-4545-4545
ip-address	An IP v4 or IP v6 address	130.216.1.1 ::1:1
mac-address	A MAC address	001B638445E6 00:1b:63:84:45:e6 00-1B-63-84-45-E6
person	Name of a person, of a 100,000 predefined set of names (common first/surnames)	John, Amy, Smith, Claus
money	A monetary amount, euros, dollars or Brithish pounds	$4.55, $ 12
date	A date object, SimSage understands the main formats, US, ISO, and UK	22/05/2020, 2 January 1990
time	A time object	11:23, 11:23:30
number	A number of some sort	12, 12,323, 145.10
email	An email address	test@test.com
city	Name of a city, SimSage has a set of 2 million cities predefined	New York, Paris
continent	One of the seven continents	Australia, America
country	One of the 250 something countries in the world	Albania, Belgium
phone	A UK compatible phone number	08931 489 381 (44) 8931 489 381
capital	A capital of the countries in the world	Paris, Washington
law-firm	One of a set of predefined law firms.	Ashursts, Blake Morgan
secret	29 secret OAUTH/key patterns. Google keys, Slack keys, RSA	AIza01234567890\13_A\ 189a1234123789_233

	keys, PGP keys, Facebook keys, Heroku keys, Paypal keys, Stripe keys, Twitter keys, Square keys, AWS keys, Mailgun keys.	amzn.mws.1234abcd-ab12-cd23-44ff-123456abcdef
zip	US zip code	09498-0048
postcode	UK postcode	CB1 4DD
ssn	US Social Security Number	902-10-5000

Super Search Syntax

Here is the BNF (Backus Naur Form) for the Super Search Syntax. Terminals are enclosed in single quotes (e.g. 'exact:'). The pipe symbol | is used as an OR for the grammar. Square brackets denote optional items. *e* is the empty symbol. The non-terminal symbols (in *italic*) are:

query	an expression, the start symbol for any super-search
word_marker	an extension for 'word' with optional markers
word	a free string representing a search word
metadata	a metadata name, without spaces, restricted string
number	an integer number value
e	empty set

BNF

query → 'meta' '(' *metadata* ',' *word* ')' *query* |
'range' '(' *metadata* ',' *number* ',' *number* ')' *query* |
'num' '(' *metadata* ',' *number* ')' *query* |
'source' '(' *word* [',' *number*] ')' *query* |
'syn' '(' *word* ',' *number* ')' *query* |
'dist' '(' *number* ')' *query* |
'word' '(' *word* [',' *word_marker*] ')' *query* |
'entity' '(' *metadata* ')' *query* |
'sort' '(' ')' *query* |
'group' '(' ')' *query* |
'(' *query* ')' *query* |
query 'and' *query* |
query 'sub' *query* |
query 'or' *query* |
query 'and' 'not' *query*

word_marker → 'n' [',' word_marker] |
'v' [',' word_marker] |
'a' [',' word_marker] |
'exact' [',' word_marker] |

SimSage
Search less; find more.

metadata [',' word_marker] |
number [',' word_marker] |
e

The terminal symbols (in single quotes) are:

range	a metadata numerical or monetary range filter
entity	a semantic entity marker
meta	a metadata categorical filter
dist	set the maximum distance allowed between words (default 40)
syn	UX filter, set the syn-set to use for a given word
num	search for an exact number in a metadata set
source	specify a list of source to filter by / use
word	set a single word to search for with additional filter fields
exact	word filter: the exact word must match
n	word filter: the text following must be a noun
v	word filter: the text following must be a verb
a	word filter: the text following must be an adjective
metadata	word filter: denotes a metadata marker, the default marker is "body" and denotes the content of a document. Other well known fields include "url", "title", "author" and any other name that occurs in data.
and	a logical AND between statements
sub	a sub-search between statements
or	a logical OR between statements
and not	a logical AND NOT at the end of a statement
sort	a sort indicator, overwriting search result ordering (default is score based)
group	a grouping of similar document indicator
#	a hashtag metadata item (any word starting with # is a hash-tag)
(…)	bracket precedence marker

Precedence
This grammar has a precedence structure. Brackets ( ) have the highest precedence, followed by SUB. You cannot construct a query that has anything more important than a SUB.

The following query is rejected as it puts "AND" before "SUB":

((word(rock) sub word(mark)) and (word(Sean)))

The correct version:

((word(rock) and word(mark)) sub (word(Sean)))

The use of OR

OR is used for metadata search items like *meta*(…), and *range*(…). **OR is not to be used for body text searches**. The semantic expansions give you a natural logical OR.