

[\[back to Help Center \]](#)

The Basics of Creating and Refining Queries

- [Overview](#)
- [Entering Queries](#)
- [Complex Query](#)
- [Boolean Operators](#)
- [<in>](#)
- [Searching for Phrases](#)
- [Date Range Searching](#)
- [Exact Words vs. Stemming](#)
- [Weighting Search Terms](#)
- [<accrue>](#)
- [Nesting and Order of Operation](#)
- [Order of Precedence](#)
- [Relevancy Scoring](#)
- [<near> <near/n> and <order>](#)
- [Thesaurus Operator](#)
- [Wildcards](#)
- [Relevancy Scoring, Weighting, and <accrue> Compared](#)
- [Alternative Languages](#)

See [Field Help: Definitions and Searching](#) in the Delphion Help Center for field-specific definitions and [abbreviations](#).

See [Using Corporate Tree](#) in the Delphion Help Center for information on using Corporate Tree for Assignee searches.

Overview

This information will help you craft your queries to produce more relevant and manageable result sets. Queries that are carefully and purposely created let you spend less time looking for information and more time acting on it.

[\[back to top\]](#)

Entering Queries on the Delphion Website

Use the [Advanced](#) search page to enter your own queries on the Delphion website.

The Advanced Text Search section of the page (at the top) allows you to specify which collections you want searched and how many queries should be displayed per page in your result set. There are also drop down boxes that allow you to specify a date range.

When you receive your result set, the top of the result set page will show you how Delphion has reformatted your query to make it flow quickly through the search engine. If your query does not return the expected results, you may want to study the reformatted query to see if there are clues to what might be the problem.

Also at the top of the result set page, you will find a text entry box in which the reformatted query is repeated. Use this text entry box to change your query — or enter a new one.

Immediately following the text entry box is confirmation of the collections searched to produce this result set.

[\[back to top\]](#)

Complex Queries

In general a complex query may contain any of the following:

- Several Boolean operators
- A search applied against two or more collections
- Over 100 characters
- Utilization of more than one of the syntax modifiers listed after [Boolean Operators](#) such as: proximity, thesaurus, wildcard or weighting operators

If a complex query is not carefully designed, the results of the search could yield:

- An error message (e.g., "0 matches found out of 0 patents searched")
- A search which times out
- A partial result set that is actually a subset of possible matching records for the collections searched (this could be represented by a series of inconsistent results)

Please follow these suggestions to make your searching go smoothly when designing complex queries. If your query is complex, do not use:

- "Left truncation" (i.e., an asterisk, *, as the first character of a search term)
- An asterisk, *, after a single letter (unless the asterisk is followed by at least two alphanumeric characters)
- Multiple question marks, ?????, without three or more preceding alphanumeric characters
- More than five terms containing asterisks applied against more than one patent office collection

If you follow these suggestions and have difficulties, try these additional suggestions:

- Apply the complex query against only one patent office collection at a time. Save the result set for each individual collection as a [Work File](#). After you have executed your entire complex query string in sections and saved the corresponding results sets as Work Files, use the Work File merge functionality to create a new result set.
- Do not submit the same query over and over again within a short time period. If you are not receiving the results you expect from a complex query, please contact Customer Technical Support for assistance.
- Reduce the number of wildcards in your query because too many wildcards can hamper multiple patent office collection searches
- Break the complex query string into individual sections and execute a search for each section. Save the result set for each query string section as a [Work File](#). After you have executed your entire complex query string in sections and saved the corresponding result sets as Work Files, use the Work File merge functionality to create a new result set.

For additional assistance with complex queries, please contact your [Customer Technical Support](#) representative.

[\[back to top\]](#)

Boolean Operators

Use Boolean operators to indicate which keywords you want included in, or excluded from, your search result set.

- Use **AND** to join two terms that must both be present
- Use **OR** to join two terms if one term or the other must be present
- Use **NOT** to specify a term which should be excluded

Operators are not case sensitive, so AND, OR, or NOT do not need to be capitalized. Many people do so, however, to make it clear to themselves which words are query terms and which words are operators.

Example 1, the AND operator:

To search for all patents with *Hewlett-Packard* as the Patent Applicant/Assignee (PA) and both the words *printer* and *scanner* in the Title (TI), use the following query:

(Hewlett-Packard <in> PA) AND (printer AND scanner) <in> TI

Note that *printer* and *scanner* as well as the Boolean operator *AND* are enclosed by parentheses. The parentheses indicate that both the words *printer* and *scanner* should be in the TI field. If the parentheses were omitted, the result set would include patents with the word *printer* shown in any field in the patent record and with *scanner* in the TI field.

Example 2, the OR operator:

To search for all patents with *Hewlett-Packard* as the Patent Applicant/Assignee (PA) and with

either the word *printer* or the word *scanner* in the Title (TI), use the following query:

(Hewlett-Packard <in> PA) AND (printer OR scanner) <in> TI

Note that, like the example above, the words *printer* and *scanner* as well as the Boolean operator are enclosed by parentheses. The parentheses indicate that both the words *printer* and *scanner* should be considered when the TI field is searched. In this case, the Boolean operator is OR, so the patent qualifies if either (or both) of those words is found in the TI field.

Example 3, the NOT operator:

To search for all patents with *Hewlett-Packard* as the Applicant/Assignee (PA) and with the word *printer* but not the word *scanner* in the Title (TI), use the following query:

(Hewlett-Packard <in> PA) AND (printer NOT scanner) <in> TI

Like the other examples, the words *printer* and *scanner* as well as the Boolean operator are enclosed by parentheses so that both of the words are considered when the TI field is searched. In this case, the Boolean operator is NOT, so the patent will qualify for the result set only if the word *printer* is present and the word *scanner* is not present.

Example 4, stringing NOTs:

You can *string* or use multiple occurrences of the NOT operator to exclude several terms from your query. To include the word *printer* but exclude these common variations of the phrase *ink jet* from your results, use the following query:

(Hewlett-Packard <in> PA) AND (printer NOT "ink-jet" NOT "inkjet" NOT "ink jet") <in> TI

Stringing AND NOTs produces exactly the same results but requires more key strokes. However, if you prefer, you could construct your query like this:

(Hewlett-Packard <in> PA) AND (printer AND NOT "ink-jet" AND NOT "inkjet" AND NOT "ink jet") <in> TI

[\[back to top\]](#)

Using the <in> Operator

<in> is a proximity operator that helps you select documents which contain your keywords in specified fields in the patent record. Proximity operators are always enclosed in angle brackets (less than and greater than symbols). They are not case sensitive so you may use upper or lower case.

When using the <in> operator, you need to specify the field you want to have searched. You can use the accepted full field name or an accepted abbreviation for the field name. The following two queries return exactly the same result set:

Query with full field name:

monoclonal <in> title

Query with field name abbreviation:

monoclonal <in> TI

Field name abbreviations are usually shown all upper case but they are not case sensitive either. The following returns the same result set as the two preceding queries:

Query with field name abbreviation in lower case:

monoclonal <in> ti

Using natural language for the field name will not work. The following query will return a message advising that your query is improperly formed:

Field name improperly formed:

monoclonal <in> the patent title

NOTE: The <in> operator does not work with date fields.

[\[back to top\]](#)

Searching for Phrases

By default, the search engine performs *phrase searching*. This means two or more words entered without commas or other operators will be treated as a unit — a phrase — when the search is performed. So, a search for *optical fiber* will return a result set that includes optical fiber, optical fibers, and optic fiber. The words will always be adjacent and always be in the specified order. Variations of the words will appear because stemming is also, by default, turned on.

If you want your search to include the words *optical* and *fiber* in any order and not necessarily next to each other, then construct your query in a way that turns phrase searching off.

Search syntax...	Phrase searching is...	Result set includes...
optical fiber	on	<ul style="list-style-type: none"> ● optical fiber ● optical fibers ● optic fiber
(optical AND fiber)	off	<ul style="list-style-type: none"> ● optical fiber ● fiber optic ● optical transmission fiber ● optical glass fiber
relational database	on	<ul style="list-style-type: none"> ● relational database ● relational databases ● related databases
(relational AND database)	off	<ul style="list-style-type: none"> ● relational databases ● method for querying multiple, distributed databases by selective sharing of local relative... ● method relating to databases ● database relational extenders

See also: [Searching for Exact Terms vs. Stemming](#)

[\[back to top\]](#)

Date Range Searching

Date Range Searching allows you to define or restrict the dates of your search.

Use the name of a date field followed by one or more of the mathematical symbols for *less than*, *equal to*, or *greater than*. Then specify the dates you want searched. For the best results, always use the ISO date format: YYYY-MM-DD.

For dates that are...	Use this...	Which means...
prior to a specific date	<	less than
equal to a specific date	=	equal to
after a specific date	>	greater than
prior to or equal to a specific date	<=	less than or equal to
after or equal to a specific date	>=	greater than or equal to

between two dates, including the start and stop dates

\geq and \leq

greater than or equal to *and* less than or equal to

NOTE: The less than, equal to, and greater than symbols only work with date fields, they cannot be used with text fields.

Example 1, dates earlier than:

To search for all patents with Publication Date (PD) *before* (less than) 1991-01-01 with *Canon* as the Patent Applicant/Assignee (PA) with the word *image* in the Title (TI), use the following query:

(Canon <in> PA) AND (image <in> TI) AND (PD<1991-01-01)

Notice that each of the three main elements of this query are enclosed by parentheses to ensure that each element is considered independently when patent fields are searched.

In this case, the Boolean operator AND is used between each of the query elements to indicate that positive results in all three elements of the query are required for a patent to appear in your result set.

Example 2, dates earlier than or equal to:

To use the same basic query as the one shown in Example 1, but this time include patents actually issued on 1991-01-01, use both the less than and equal to symbols as shown in this query:

(Canon <in> PA) AND (image <in> TI) AND (PD<=1991-01-01)

The result set for this query will contain all the patents shown in the result set from the Example 1 query and will also contain the patents with Publication Date (PD) actually equal to 1991-01-01.

Example 3, dates between two dates:

To search for patents with Publication Dates (PD) between 1988-12-31 and 1991-01-01, construct your query like this:

(Canon <in> PA) AND (image <in> TI) AND (PD>1988-12-31 AND PD<1991-01-01)

1989-01-01 is the first date that is greater than 1988-12-31 and 1990-12-31 is the first date less than 1991-01-01. So this result set will include patents issued on 1989-01-01 and patents issued on 1990-12-31 — as well as all the patents issued in between (as long as they meet the rest of the specified criteria).

Example 4, dates between two dates, including the start and stop dates:

To search for patents with Publication Dates (PD) between 1988-12-31 and 1991-01-01, but including those two dates, construct your query like this:

(Canon <in> PA) AND (image <in> TI) AND (PD>=1988-12-31 AND PD<=1991-01-01)

1988-12-31 will be the first date included in your result set. 1991-01-01 will be the last date included in your result set. You will also get all of the dates in between.

NOTE: When searching large or multiple collections (which return more than 500 records for your query) you will not be able to see all of the patents that fall into your desired date range. If this is a problem, you should rewrite and target your query to return a result set that is under 500 records and therefore can be viewed easily and manipulated accurately — and using date ranges is usually the easiest and quickest way to do this.

[\[back to top\]](#)

Searching for Exact Words vs. Stemming

The Delphion search engine automatically performs *stemming*. This means when you enter a word such as *prime*, your result set will include words that share a root, or stem, with the word you searched. So, for *prime*, your result set will include words like primed, priming, primaries and primates.

If you do not want stemming used for your search, you need to specify in your query that only the exact keyword should be searched. Do this by enclosing your keyword(s) in double quotes.

Search syntax...	Stemming is...	Result set includes...
prime	on	prime, priming, primed, primates, primaries, and other words with the same stem
"prime"	off	prime
carbon	on	carbon, carbons, carbonate, carbonates, carbonated, and other words with the same stem
"carbon"	off	carbon

Stemming is a linguistic process and your results will include *linguistic* expansions of the stem word. Use wildcards for a result set that includes all (right-hand) expansions of a stem or word.

See also: [Searching for Phrases](#) and [Wildcards](#)

[\[back to top\]](#)

Nesting and Order of Operation

Parentheses are used to create nests which define the order of operation. Nesting directs the search engine to process your query in an exact order, avoiding misunderstandings. The two queries below will retrieve radically different result sets. As in algebra, what appears inside parentheses is processed first.

Search syntax...	Result set
(driving OR protection) AND helmet	under 1000 records
driving OR (protection AND helmet)	over 150,000 records

[\[back to top\]](#)

Order of Precedence

Query expressions are read using specific *precedence* rules. This means that certain operators are processed before others. While query expressions are read from left to right, some are processed before others and this can impact the way the search engine interprets your query.

The following shows the order or precedence in which operators are processed:

- <yesno>
- <thesaurus>
- NOT
- <near> *and* <near/n> *and* <order><near> are all treated equally
- <in>
- AND
- OR

The following examples show how the precedence rules can impact the manner in which the search engine processes your query.

Example 1, AND before OR:

If you want to search for patents about *feline disease* or *ferret disease*, and you enter

ferret OR feline AND disease

because AND is treated before OR, the search engine will interpret your query to mean this

ferret OR (feline AND disease)

and your result set will include records with *feline* and *disease* or records with *ferret* that may or may not include the term *disease*.

So here is a better way to construct the query

(ferret OR feline) AND disease

Now all records in your result set will contain the word *disease* and either the word *feline* or the word *ferret*.

Example 2, <in> before AND:

If you want to search for patents with the words *feline* and *disease* in the Title, and you enter

feline AND disease <in> title

because <in> is treated before AND, the search engine will interpret your query to mean this

feline AND (disease <in> title)

and your result set will include records with *disease* in the Title and *feline* in any searchable field.

So here is a better way to construct the query

(feline AND disease) <in> title

Now all records in your result set will contain both the words *feline* and *disease* in the Title.

[\[back to top\]](#)

Relevancy Scoring

Each record in your Result Set has a score (shown on the far right of the Result Set page). The score is an indication of that record's relevancy to your query.

The score is based on algorithms built into our search engine which factor in the density of the search terms found in the document. Density is calculated in proportion to overall document text — because of this, a longer document that contains more occurrences of a word can score lower than a shorter document that contains fewer occurrences. And, when proximity operators are used, the nearness of terms within the document is also a factor. The exact details of the scoring algorithm are proprietary. This is the default scoring mechanism.

The relevancy score is most useful when searching for key words in text. It's less useful when you're searching for one instance of a term in specific fields, such as in IPC searching or assignee searching.

If you prefer not to use the relevancy score, use the "Use Relevancy Score?" toggle on the Current Results tab of your Result Set to turn it off. You can also easily turn it off by inserting the <yesno> operator before your query like this:

<yesno> (carbon <in> ti)

When relevancy scoring is on, the Result Set is sorted in relevancy order. When scoring is off, the Result Set is returned in date order. You can easily re-sort your Result Set by clicking the desired column header on the Result Set page.

[\[back to top\]](#)

<near> <near/n> and <order> Operators

The <near> and <near/n> proximity operators allow you to retrieve documents which are

relevance-ranked based on the closeness (proximity) of the specified words.

<near> selects documents containing your specified search terms. Delphion's search engine scores a 0 for documents where the search terms are not within 1024 words of each other. The closer the search terms are within a document, the higher the document's relevancy score.

<near/*n*> selects documents containing two or more search terms within a specified number of words of each other. *n* can be an integer up to 1024. Once again, the closer the search terms are within a document, the higher the document's score.

Neither the <near> or the <near/*n*> operators specify the order of the terms. To specify the order of the terms, use <order> preceding the <near> or <near/*n*> operators.

Note: You cannot use a NOT command in conjunction with a proximity operator.

Example 1, the <near> operator:

The following queries demonstrate the syntax for the <near> operator:

coffee <near> filter (This Result Set will include all records with the word *coffee* near the word *filter*.)

(coffee <near> filter) <in> description (This Result Set will include all records with the word *coffee* near the word *filter* in the Description field.)

(coffee <near> filter) <in> (abstract,claims) (This Result Set will include all records with the word *coffee* near the word *filter* in the Abstract or Claims fields.)

((dog OR cat) <near> carrier) <in> description (This Result Set will include all records with the word *dog* or the word *cat* near the word *carrier* in the Description field.)

It is important to remember that the closer the search terms are to each other, the higher the relevancy score of the document.

Example 2, the <near/*n*> operator:

The following queries demonstrate the syntax for the <near/*n*> operator as well as the difference in Result Sets when the variable is changed:

(inorganic <near/10> solvents) <in> claims (over 1,000 records found for collection searched)

(inorganic <near/100> solvents) <in> claims (over 4,000 records found for collection searched)

(inorganic <near/300> solvents) <in> claims (over 5,000 records found for collection searched)

Remember that the closer the search terms are to each other, the higher the relevancy score of the document.

Example 3, the <order> operator used in conjunction with <near> and <near/*n*>:

The following queries demonstrate the syntax for using the <order> operator to specify the order in which the target words should appear to meet the criteria for your query when using the <near> and <near/*n*> operators.

((dog OR cat OR pet) <order><near> (cage OR carrier)) <in> description

For this query, records with the words *dog*, *cat* or *pet* near the words *cage* or *carrier* will be found — but only those instances in which *dog*, *cat* or *pet* precede the word *cage* or *carrier*. If the <order> operator had not been used, then the Result Set would also include instances in which *cage* or *carrier* appeared before *dog*, *cat* or *pet*.

((wire OR mesh OR screen) <order><near> (dog OR cat OR pet)) <order><near> (cage OR carrier) <in> abstract

The Result Set for this query contains records with the word *wire*, *mesh* or *screen* near but preceding the word *dog*, *cat* or *pet* which is near but precedes the word *cage* or *carrier*. Following is an abstract from a patent in the Result Set for this query:

A collapsible pet enclosure that is fashioned to fit in standard windows and has removable material coverings and **screen** panels on all sides. Domestic animals can use the enclosure to enjoy the out doors with out the owners fearing for their safety or health, as is the case in letting pets out of doors on the ground level. There are detachable wheels as part of the unit that convert the enclosure to a mobile **pet carrier**, that one may roll instead of carry.

The `<near/n>` operator works in much the same way — but allows you to specify the maximum distance between your target words and still have the words appear in a preferred order. Following is an example of using `<order>` with `<near/n>`:

`((inventory OR component OR parts) <order><near/10> (management OR tracking OR location) <order><near/15> ("bar code" OR hand-held OR wireless)) <in> abstract`

This Result Set for this query will include all records with *management* or *tracking* or *location* within 10 words of *inventory* or *component* or *parts* and then, within 15 words of that word combination, the word(s) "*bar code*" or *hand-held* or *wireless* must appear. When this search is done with the `<near/n>` operator, a Result Set shows 17 patents — done with just the `<near>` operator, the Result Set had over 100 patents. So you can see how the `<near/n>` operator can help refine your search. Following is an abstract from a patent in the Result Set for this query:

A method of **inventory management** is described. Upon activation of a button on a **wireless** device, the wireless device having a light source and a transceiver with a unique media address corresponding to a unique product, the device broadcasts a first signal including an order command and the unique media address by the transceiver via a wireless medium. A central controller then receives the first signal, identifies the unique media address included in the first signal, and using a database, identifies the unique product associated with the unique media address.

[\[back to top\]](#)

Thesaurus Operator

The THESAURUS operator searches an index of terms and finds synonyms for the key term in your query.

The following query asks the search engine to look for the word *bow* in the Title (TI) field.

`bow <in> TI`

The Result Set includes patents for an archery bow, bow tuning equipment, a gift wrapping bow, a bow assembly for a ski lift, a key bow, a ladder for a boat bow, a bow rake, and a violin bow cover. All patent records have the word *bow* in the Title. For the collection searched, the Result Set was over 1,000.

This query uses the `<THESAURUS>` operator to ask for results that include synonyms for the word *bow* in the Title.

`<THESAURUS> bow <in> TI`

This Result Set includes patents with the word *bow* and patents with synonyms for the word *bow* in the Title. Synonyms found include: bend, bending, turn, turnable, yield, yielding, curved, curving, round, and crook. For the collection searched, the Result Set was over 10,000.

NOTE: The thesaurus includes mostly common words and may not be helpful for technical or scientific terms.

[\[back to top\]](#)

Wildcards

A *wildcard* is a symbol that stands for one or more unspecified number of characters in a query.

Delphion supports the use of wildcards in the right-hand position, after a specific character string (known as *right-hand truncation*), in queries for all collections. This means that you can use both the asterisk and the question mark wildcards to the right-hand side of a search term. This allows you to retrieve words that begin with a specific character string but have one or more unknown characters at the end.

For left-hand wildcards (known as *left-hand truncation*), using both the asterisk and the question mark are supported for both the German Applications and German Granted collections. For all other collections, only the question mark is supported for left-hand truncation. This allows you to retrieve words that have one or more unknown characters at the beginning, but end with a specific character string.

The following wildcard symbols are valid:

Wildcard Symbol	Explanation
?	<p>The question mark wildcard represents one character. Use one or more question marks to stand for a specific number of characters in your search term.</p> <p>The question mark can be used within a word.</p>
*	<p>The asterisk wildcard represents zero or an unlimited number of characters.</p> <p>The asterisk can also be used within a word.</p> <p>When using an asterisk as a right-hand wildcard, the base word must have more than one character (e.g., you cannot use <i>x*</i>; you can use <i>ox*</i>).</p>

Following are examples of queries using wildcards:

Search syntax...	Results include...
carbo?	<p>Words with only one character following the specified search string of <i>carbo</i>, including:</p> <ul style="list-style-type: none"> • carbon • carbox • carboy
carbo??	<p>Words with exactly two characters following the specified search string of <i>carbo</i>, including:</p> <ul style="list-style-type: none"> • carbons • carboxy • carbody • carbony
car?on	<p>Words with one character between <i>car</i> and the suffix <i>on</i>, including:</p> <ul style="list-style-type: none"> • carbon • carton

carbo*	<p>Words with zero or an unlimited number of characters following the specified search string of <i>carbo</i>, including:</p> <ul style="list-style-type: none"> ● carbon ● carbonate ● carbonated ● carbohydrate ● carbonyl ● carboxylic ● carborane-(siloxane or silane)-acetylene ● carbonless ● carboxamide ● carboxylate ● carbonaceous ● carboxypeptidases ● carboxamido
carbo*ate	<p>Words with an unlimited number of characters between <i>carbo</i> and the suffix <i>ate</i>, including:</p> <ul style="list-style-type: none"> ● carbonate ● carbohydrate ● carboxylate-sulfate ● carboxydiketonate ● carbocysteinatate ● carboxysulfonate ● carbothiolate ● carboxylic-acid-amidothiolcarbamate
???oxide	<p>Words with three characters before the suffix <i>oxide</i>, including:</p> <ul style="list-style-type: none"> ● monoxide ● peroxide ● hexoxide ● alkoxide ● suboxide ● trioxide ● ethoxide
????oxide	<p>Words with four characters before the suffix <i>oxide</i>, including:</p> <ul style="list-style-type: none"> ● hydroxide ● nioboxide
*oxide	<p>Words with zero or an unlimited number of characters before the suffix <i>oxide</i>, including:</p> <ul style="list-style-type: none"> ● monoxide ● peroxide ● hexoxide ● alkoxide ● suboxide ● trioxide ● ethoxide ● titanoxide ● hydroxide ● nioboxide ● tetrahydrothiopyran-phenyloxazolidinon-s-oxide ● lithium-mangan-mischoxide <p>NOTE: The asterisk wildcard in the left-hand position</p>

(left-hand truncation) is valid only for the German Applications and German Granted collections.

[\[back to top\]](#)

Weighting Search Terms

You can assign a weight to each search term in a query to indicate the term's relative importance to your search. The weight assignment is shown as a number between 01 and 100, where 01 represents the lowest importance rating and 100 represents the highest. You would never use weighting with just one search term because you are weighting (or comparing) one term against another.

For example, you are working on a toothbrush holding device and want to check for related patents. You are interested in patents on toothbrush holders but also patents on holders of any sort. On a relative scale, patents on toothbrush holders are twice as important to you as patents on other kinds of holders, so you weight them accordingly in your query. To construct this kind of query, enclose the weight in brackets and join the terms with OR, like this:

[100](toothbrush AND holder) OR [50]holder)

When sorted by score, the Result Set for this query will give patents with both the terms *toothbrush* and *holder* a higher score than those with just the term *holder*. Restricting your query in terms of time range (e.g., to a single year) will help highlight the scoring differences.

[\[back to top\]](#)

<accrue> Operator

The <accrue> operator selects documents containing at least one of the search terms you specify (it is somewhat like OR in that manner). <accrue> however ranks documents according to the number of times the search elements appear in a given document — the more occurrences, the higher the score.

The <accrue> operator *accrues*, or adds up, the number of occurrences of all of the terms combined for scoring. Because of this, <accrue> is not valid with just one search term.

Construct your query like one of these:

toothbrush <accrue> holder
 or
toothbrush <accrue> holder <accrue> soap
 or
<accrue> (toothbrush, holder)
 or
<accrue> (toothbrush, holder, soap)

[\[back to top\]](#)

Relevancy Scoring, Weighting, and <accrue> Compared

Weighting, relevancy scoring, and <accrue> are three search options that are somewhat related and often confused. If you ask how documents are scored, the brief answer is that they are scored by frequency of key words. There are however, different aspects of that scoring that you should be aware of as well as different ways in which you can influence the scoring.

Relevancy score

[Relevancy scoring](#) is based on the density of search terms in the retrieved documents. This is the default scoring mechanism.

Weighting

[Weighting](#) is something you request when you form your query. You tell the search engine how you want the qualifying documents weighted against each other.

<accrue>

The [<accrue>](#) operator selects documents that include at least one of the search elements you specify. The more search elements that are present, the higher the score will be.

Results

The following demonstrates how a changing the way a query is formed can impact relevancy scoring. While these queries are not completely parallel, they show how one patent can receive different scores in similar queries.

Default relevancy scoring decided the scores for these two queries:

(toothbrush OR holder) <in> AB = 97%

toothbrush <in> AB = 94%

Weighting of terms when the query was constructed produced this score:

(([100](toothbrush AND holder) OR [50]holder) <in> AB) = 98%

The [<accrue>](#) operator was used to produce this score:

<accrue> (toothbrush, holder) <in> AB = 70%

Actual occurrences of the search terms in the patent abstract:

- toothbrush = 13
- holder = 7
- toothbrush holder = 7

[\[back to top\]](#)

Alternative Languages (EP and WO collections only)

For EP and WO patents, the Title, Abstract, Claims, Description, and Text fields can be searched in German, French, and Spanish.

Use the following search terms in your query:

Field	Language & Search term
Title	German: titlede
	French: titlefr
	Spanish: titlees
Abstract	German: abstractde
	French: abstractfr
	Spanish: abstractes
Claims	German: claimsde
	French: claimsfr
	Spanish: claimses
Description	German: descriptionde
	French: descriptionfr
	Spanish: descriptiones
Text (searches all text fields, including Title, Abstract, Claims, and Description)	German: textde
	French: textfr
	Spanish: textes

Format your query in the following manner:

aufzug <in> titlede

or

ascenseur <in> descriptionfr

NOTE 1: These queries search only those records that have data available in the specified language, within the specified collection.

NOTE 2: English is the primary or default language for Delphion and for the Delphion Integrated View. This means that, when there is an English language version of field information, the English will display on the Integrated View.

For example, your Result Set for the search `aufzug titlede` could display titles without the word "aufzug" but, instead, the English equivalent which is "elevator." Those will be patents that have an English version of the title available (the English displays first because it is the default). When you display an Integrated View for one of these patents, click the language toggle to display the title (or other fields) in German.

NOTE 3: When searching the German collection, you do not need to use `titlede`, `abstractde`, `claimsde`, `descriptionde`, or `textde` because the primary language of the collection is German.

[\[back to top\]](#)

