

IBM StoredIQ Insights

User Guide



Note

Before using this information and the product it supports, read the information in [Notices](#).

This edition applies to Version 7.6.0.17 of product number 5724M86 and to all subsequent releases and modifications until otherwise indicated in new editions.

© **Copyright International Business Machines Corporation 2018.**

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

About this publication.....	iv
IBM StoredIQ product library.....	iv
Contacting IBM StoredIQ customer support.....	iv
IBM StoredIQ Insights.....	1
Working with IBM StoredIQ Insights.....	2
Searching and evaluating results.....	3
Filter condition reference and expression language.....	5
Full-text syntax.....	5
Full-text macros.....	8
Indexed annotations.....	9
Regular expressions.....	9
Extended ASCII characters.....	10
Attribute condition reference.....	12
Specification of attribute conditions.....	12
Type of object to match.....	17
Multiple matching values for operators.....	18
At-least expressions.....	19
Parent and child conditions.....	19
No-inheritance conditions.....	20
Notices.....	21
Privacy policy considerations	22
Terms and conditions for product documentation.....	23
Trademarks.....	23
Index.....	25

About this publication

IBM StoredIQ Insights User Guide describes how to search for data of interest and evaluate the results.

IBM StoredIQ product library

The following documents are available in the IBM® StoredIQ® product library.

- *IBM StoredIQ Overview Guide*
- *IBM StoredIQ Deployment and Configuration Guide*
- *IBM StoredIQ Data Server Administration Guide*
- *IBM StoredIQ Administrator Administration Guide*
- *IBM StoredIQ Data Workbench User Guide*
- *IBM StoredIQ Policy Manager User Guide*
- *IBM StoredIQ Insights User Guide*
- *IBM StoredIQ Integration Guide*

Contacting IBM StoredIQ customer support

For IBM StoredIQ technical support or to learn about available service options, contact IBM StoredIQ customer support at this phone number:

- 1-866-227-2068

Or, see the Contact IBM web site at <http://www.ibm.com/contact/us/>.

IBM Knowledge Center

The IBM StoredIQ documentation is available in [IBM Knowledge Center](#).

Contacting IBM

For general inquiries, call 800-IBM-4YOU (800-426-4968). To contact IBM customer service in the United States or Canada, call 1-800-IBM-SERV (1-800-426-7378).

For more information about how to contact IBM, including TTY service, see the Contact IBM website at <http://www.ibm.com/contact/us/>.

IBM StoredIQ Insights

IBM StoredIQ Insights provides dynamic and interactive filtering for your data with easy access to all metadata and instant plain-text preview of document content for full-text indexed volumes.

Faceted search lets you drill down to refine your search results as needed. In addition, you can apply any valid IBM StoredIQ filter query.

With IBM StoredIQ Insights, you can search data that is managed and indexed by a data server of the type DataServer - Distributed. In mixed deployments that have classic and distributed data servers, only the content from distributed data servers will be searchable.

IBM StoredIQ Insights

ES2_hq12f1n1p1p1-v2 - System Index

Facets

Category

Word processing (248)

Email (76)

Preparation (77)

Spreadsheets (76)

Container files and al... (52)

Computer programs... (5)

Stencils (5)

File name extension

txt (72)

pdf (77)

rtf (76)

doc (74)

docx (84)

htm (58)

Show more

Object size

less than 500 KB (376)

500 KB - 1 MB (84)

1 MB - 50 MB (5)

50 MB - 100 MB (5)

more than one of every dem.pdf

Created date

over 3 years ago (423)

last 30 days (2)

Modified date

over 3 years ago (423)

last 30 days (2)

Volume

ES2_hq12f1n1p1p1-v2 (423)

Path

423 results (click down for more)

Object name	Size	Volume	Path	Created	Last modified
Delivered: Almost every attribute email.pdf	5.40 KB	ES2_hq12f1n1p1p1-v2	pdf	2008-01-30 2:16 AM	2008-01-30 2:16 AM
Delivered: Almost every attribute email.pdf	7.12 KB	ES2_hq12f1n1p1p1-v2	pdf	2008-01-30 2:16 AM	2008-01-30 2:16 AM
Asunt_Tut_A_Mental_Health_in_Africa_A_Bonifacio.zip	666.20 KB	ES2_hq12f1n1p1p1-v2	zip	2007-10-29 11:34 PM	2007-10-29 11:34 PM
..._DS_Store	4.00 KB	ES2_hq12f1n1p1p1-v2	pdf	2018-09-26 2:56 PM	2018-09-26 2:56 PM
image001.jpg	3.22 KB	ES2_hq12f1n1p1p1-v2	pdf	2008-01-24 12:41 AM	2008-01-24 12:41 AM
smokeless2PDF.zip	14.10 KB	ES2_hq12f1n1p1p1-v2	zip	2008-01-01 12:36 PM	2008-01-01 12:36 PM
Asunt_Tut_A_Mental_Health_in_Africa_A_Bonifacio.zip	666.20 KB	ES2_hq12f1n1p1p1-v2	zip	2007-10-29 11:34 PM	2007-10-29 11:34 PM
smoke.pdf	10.63 MB	ES2_hq12f1n1p1p1-v2	pdf	2013-09-10 1:46 AM	2013-09-10 1:46 AM
image001.jpg	2 bytes	ES2_hq12f1n1p1p1-v2	pdf	2008-01-30 2:17 AM	2008-01-30 2:17 AM
Asunt_Tut_A_Mental_Health_in_Africa_A_Bonifacio.zip	666.23 KB	ES2_hq12f1n1p1p1-v2	zip	2007-12-20 5:51 PM	2007-12-20 5:51 PM
more than one of every dem.pdf	205.00 KB	ES2_hq12f1n1p1p1-v2	pdf	2014-09-07 2:02 AM	2014-09-07 2:02 AM
no body, Berlin.pdf	4.91 KB	ES2_hq12f1n1p1p1-v2	pdf	2008-01-18 11:24 PM	2008-01-18 11:24 PM
Asunt_Tut_A_Mental_Health_in_Africa_A_Bonifacio.zip	666.23 KB	ES2_hq12f1n1p1p1-v2	zip	2007-12-20 5:51 PM	2007-12-20 5:51 PM
smokeless2PDF.rar	35.95 KB	ES2_hq12f1n1p1p1-v2	rar	2008-01-01 8:36 PM	2008-01-01 8:36 PM
immodulated.JPG	241.22 KB	ES2_hq12f1n1p1p1-v2	pdf	2008-01-11 12:14 AM	2008-01-11 12:14 AM
L.PNG	116.04 KB	ES2_hq12f1n1p1p1-v2	pdf	2008-01-11 1:04 AM	2008-01-11 1:04 AM
draft.mai	2.04 KB	ES2_hq12f1n1p1p1-v2	pdf	2008-02-14 2:06 AM	2008-02-14 2:06 AM
xamxv001.pdf	2.44 MB	ES2_hq12f1n1p1p1-v2	pdf	2014-09-07 1:02 AM	2014-09-07 1:02 AM

© Copyright IBM Corp. 2018

1

Working with IBM StoredIQ Insights

Use IBM StoredIQ Insights to quickly discover relevant data through queries that you can refine as you go.

Before you can work with IBM StoredIQ Insights, you must meet several prerequisites:

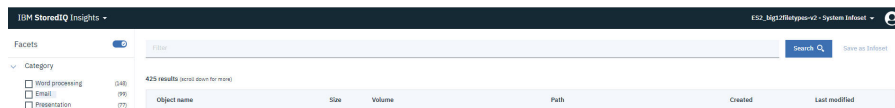
- Your deployment must include at least one Elasticsearch cluster managed by a data server of the type DataServer - Distributed.
- At least a metadata index must exist. However, some queries return meaningful results only for full-text indexed data or aren't even available if no full-text index exists.
- Access to IBM StoredIQ Insights requires the **Insights User** role.

Users with just this role have direct access to IBM StoredIQ Insights when logging in to IBM StoredIQ. Users with multiple roles can switch applications by selecting **Insights** from the list of available applications. This list can be expanded by clicking the arrow next to the application name in the global navigation bar.

What are my search options?

As an IBM StoredIQ Insights user, you can search in multiple ways:

- Select filters from the **Facets** list.
- Enter search terms in the filter field.



Facets provide high-level search criteria and group items and attributes, so called *facet filters*, to target meaningful results. The set of facet filters is dynamic based on the content of the info set. Each facet filter shows a *facet count*. The facet count indicates the number or, when you hover over a facet filter, the percentage of data objects in the info set matching the facet filter.

The set of facets to choose from basically consists of the following ones:

Indexed annotation

Select from all indexed annotations that were found in the info set. This facet is available only for info sets on which a cartridge was run.

Category

Select from the categories in the info set.

File name extension

Select from all file name extensions that exist within the info set.

Object size

Select from several size ranges.

Created date

Select from several date ranges.

Modified date

Select from several date ranges.

Volume

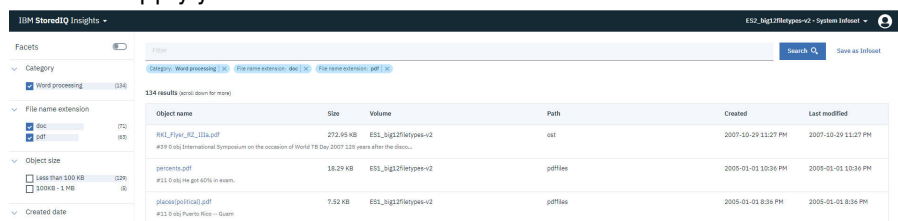
Select from the source volumes that contribute to the current info set.

Path

Select from the file paths found in the info set.

For each selected facet filter, a tag is added to the breadcrumb trail beneath the filter box, so that you can see your search criteria at a glance. By default, the filter mode is set to *automatic*, which means that search results are instantly displayed when you select a facet filter. If you want to select multiple facet

filters for a single search, you can use the toggle switch in the **Facets** pane to set the filter mode to *manual*. Alternatively, you can use the standard key-click combinations for multi-select for your operating system, which also sets the filter mode to manual. In manual filter mode, you must click the **Search** button to apply your filter criteria.



You can further refine the results by either adding or removing facet filters.

Instead of selecting facet filters, you can create your own queries. Also, you can combine a manual query with one or more facet filters. Queries that you build manually must follow the IBM StoredIQ filter syntax. For more information, see “Filter condition reference and expression language” on page 5. A query might fail due to syntax errors or if your data is spread across several Elasticsearch clusters. In this case, the error message `Query failed` is displayed.

What do the results look like?

Objects that match the specified criteria are displayed in the main content area in real time. Highlighting of search hits and the option to preview document content and to view document metadata directly help you evaluate the results.



Clicking an object name lets you access the object's content preview (if available) and its metadata. You can switch between an HTML preview of the content and the object's metadata.

You can save any result list as an infoset, which then becomes also available in IBM StoredIQ Data Workbench as a regular user infoset.

Searching and evaluating results

Search your data with IBM StoredIQ Insights for fast access to relevant information.

Remember that only data managed by data servers of the type `DataSet - Distributed` is searchable by IBM StoredIQ Insights. In an environment with both types of data server, you might therefore see mismatching object counts for mixed infosets.

1. Log in to IBM StoredIQ Insights as a user with the **Insights User** role.

- a) In a browser, enter the URL of the application stack:

```
https://appstack-host
```

- b) Log in with your IBM StoredIQ credentials.

You can use the email address or your user name.

What you see after you're logged in depends on the roles that you have. If you have no other role than the **Insights User** role, you have direct access to IBM StoredIQ Insights. If you have multiple roles, you might see one of the other IBM StoredIQ applications. In this case, you can switch to IBM StoredIQ Insights by selecting **Insights** from the list of available applications. This list can be expanded by clicking the arrow next to the application name in the global navigation bar.

2. Select an infoset.

The title bar shows your current selection. You can switch infosets by selecting a different one from the list.

You might notice that the number of data objects that is shown for the selected infoset is different from the number shown in the overall list. This is due to the fact that the infoset contains data from several volumes some of which cannot be searched by IBM StoredIQ Insights (where the content is not managed by a distributed data server).

3. Enter your search criteria and narrow the search results to find particular documents of interest:

- Drill down by selecting facet filters one by one, where each facet filter is applied to the results of the preceding search.
- Apply several facet filters at once. Set the filter mode to manual and select facet filters as required. Alternatively, use the standard key-click combination for your operating system. Then, click **Search**.
- Enter a query in the filter field and click **Search**.

The query must follow the IBM StoredIQ filter syntax. For more information, see [“Filter condition reference and expression language”](#) on page 5.

- Combine facet filters and manual queries.

To remove a facet filter, either clear the respective check box in the **Facets** list or click the **X** on the respective tag in the breadcrumb trail.

4. Check the search results.

Depending on your query, search terms are highlighted. To access a document's content preview and metadata information, click the object name in the result list. The object type might not allow for a content preview. In this case, you'll only see the metadata information.

5. Refine the results by repeating steps [“3”](#) on page 4 and [“4”](#) on page 4 until the results meet your needs.

Filter condition reference and expression language

The following topics provide in-depth information about full-text and attribute conditions and how to use the expression language.

Full-text syntax

The full-text editor supports full-text syntax.

The full-text feature supports a wide range of full-text syntax to broaden or narrow filters. A one-word term like `September` returns results from all harvested data objects that include the term in their content. Word matching is case-insensitive. A list of words matches any of those words. For example, `September OR October` match all objects that contain either September or October.

Term expansion

When lemmas are turned off (by default), a simple term such as `went` returns only those documents that contain the exact same character sequence. For more information, see the section on configuring full-text index settings in the IBM StoredIQ Data Server Administration documentation.

When lemmas are turned on, the simple term `went` is automatically expanded to return documents that contain words with the same meaning as `went`, such as `go`, `goes`, `going`, and `gone`.

Automatic term expansion applies to any terms. Term expansion can be disabled by placing the term within single quotation marks. Term expansion does not occur when a wildcard is present.

Note: To use term expansion for the anticipated results, lemmas must be turned on before the documents are indexed.

Grouping

When you are examining text, IBM StoredIQ Data Workbench evaluates the requested operators and gives precedence to some over others. These rules are described in the following list.

When parentheses are used to group operators, the expression in parentheses is given the highest precedence. If no parentheses are used in the expression, associativity rules are used instead. Right-associative means that the operator is grouped initially with the operand to its right. Left-associative means that the operators and operands to the left are evaluated before the operators and operands to the right.

- The NOT operator is right-associative.

`"October NOT November OR December"` means `"(October AND (NOT November)) OR December"`.

- Proximity operators (`w/` and `pre/`) are left-associative.

`"October w/3 November pre/5 December"` means `"(October w/3 November) pre/5 December"`, where `"October"` and `"November"` must be within three words of each other, but both must be before `"December"` by no more than five words.

`"October w/3 November and December"` means `"(October w/3 November) and December"`.

- The AND operator is left-associative.

`"October AND November OR December"` means `"(October AND November) OR December"`.

- The OR operator is left-associative. If you do not specify an operator, expressions are ORed.

`"October November OR December"` is equivalent to `"(October OR November) OR December"`.

Boolean Operators

This table lists Boolean operator syntax and descriptions.

Table 1: Boolean operator syntax	
Item	Description
AND &&	- When used between two terms, the AND operator ensures that both terms appear in the data object. For example, the filter <code>chairman AND stock</code> returns only data objects where <code>chairman</code> and <code>stock</code> both appear. - The double ampersand (&&) can be used instead of the AND operator. AND can be uppercase or lowercase.
OR 	- When used between two terms, the OR operator returns data objects where either term appears. For example, the filter <code>chairman OR stock</code> returns data objects that have <code>chairman</code>, <code>stock</code>, or both in the data object. - The double pipe () can be used instead of the OR operator. OR can be uppercase or lowercase.
NOT !	- When used before a term, the NOT operator ensures that the following term does not appear in any data objects returned. For example, <code>Apple NOT IBM</code> returns data objects where word <code>Apple</code> appears, but not <code>IBM</code>. You can think of the NOT item as "and not" or "but not". - The exclamation point (!) can be used instead of the NOT operator. NOT can be uppercase or lowercase.

Multiple Terms or Term Phrase

This table lists multiple terms or term phrase syntax items and descriptions.

Table 2: Terms	
Item	Description
" "	- Double quotation marks around a term phrase return data objects that have that phrase in order in a data object. For example, <code>"September 20 1998"</code> returns only data objects with the exact phrase of <code>September 20 1998</code>. Double quotation marks allow for wildcards in terms whereas single quotation marks do not. - Stop words are removed from a double-quoted term phrase. For example, <code>stock and option</code> returns documents with <code>stock option</code>. - Asterisks (*) and question marks (?) are treated as wildcards. For example, <code>stop option*</code> returns both <code>stock option</code> and <code>stock options</code>. - Automatic term expansion is performed if lemmas are turned on. For example, <code>fox jumps</code> returns any of these phrases: <code>fox jumps</code>, <code>fox jumped</code>, <code>foxes jump</code>, and so on.
' '	- Single quotation marks around a term phrase return data objects that have the exact same character sequence in a data object. For example, <code>'february 4 2013'</code> returns exactly <code>february 4 2013</code>. - Asterisks (*) and question marks (?) are treated as regular characters. For example, <code>'*ism'</code> returns exactly <code>*ism</code>. - Stop words are not removed. For example, <code>'to be or not to be'</code> returns exactly <code>to be or not to be</code>. - Term expansion is not performed even if lemmas are turned on. For example, <code>'went'</code> returns exactly <code>went</code>.

Table 2: Terms (continued)	
Item	Description
Multi-term, no Boolean operators	A filter for a multiple word without Boolean operators or quotation marks is an implied OR filter. For example, September 20 1998 returns data objects that have September OR 20 OR 1998.
Grouping	- Grouping with Boolean operators and parentheses allows users to narrow results by using attributes. For example, (\$10 OR billion) AND options returns data objects where \$10 or billion might be in the data object, but options must exist. - Boolean operators (AND, OR, NOT) work in a grouping or in a proximity group. So ("Sahara Desert" AND "Southeast Asia") OR ("Pacific Ocean" AND "Bavaria") returns data objects that have Sahara Desert and Southeast Asia or Pacific Ocean and Bavaria.

Wildcards

This table lists wildcard syntax items and descriptions.

Table 3: Wildcard	
Item	Description
*	- A single asterisk finds multiple characters or letters at beginning of a term. An asterisk that is placed before a partial word finds terms that end with those letters. For example, a filter for *olol returns results for betaxolol, carteolol, levobetaxolol, levobunolol, metipranolol, and timolol. - An asterisk matches any number of characters in the middle of a term. An asterisk that is placed in the middle of a term finds terms that have those characters on either side of the asterisk. For example, te*t returns results for text, test, tent, and tempest. - An asterisk that is placed after a partial word finds terms that begin with those letters at the end of a term. For example, a filter for ap* returns results for Apple, April, applied, and application. - An asterisk alone finds all objects. Note: An asterisk within single quotation marks is treated as a regular character.
?	The question mark matches a single character. For example, compl? returns results for comply but not complies or compliance. A filter for ap? finds ape, but not apple. To filter for more than one character, use a question mark for each letter, for example, ap??e. Note: A question mark within single quotation marks is treated as a regular character.

Proximity

This table lists proximity syntax items and descriptions.

Table 4: Proximity syntax

Item	Description
w/x	- Filtering uses the w/x syntax (x represents the number of terms between the two terms) to use the relationship between terms to get more targeted results. A two-term filter with one term that appears at the beginning of a data object and the other at the end might not be as relevant as when the terms appear in the same paragraph or sentence. For example, in an SEC data object, chairman w/15 options is a more relevant filter than chairman AND options appearing in the same data object that might be several pages long. - w/0 matches terms that are next to each other in any order. - w/1 matches for terms with one word between them. The terms can appear in any order in the document. - Grouping with proximity allows for even more targeted results. For example, ("Sahara Desert" "Southeast Asia") w/5 ("Pacific Ocean" Bavaria) finds either Sahara Desert or Southeast Asia within five terms of Pacific Ocean or Bavaria. - Terms can have wildcards or phrases.
pre/x	- With the pre/x syntax (x represents the number of terms between the two terms), you can further target proximity filters by dictating which term comes before another term in the data object and how far before it is. For example, chairman pre/15 options returns data objects where chairman comes up to 15 terms before options. - pre/0 matches terms that are next to each other and in the same order. - pre/1 matches terms with at most one word between them and in the same order. - Terms can have wildcards or phrases.
w/p	The w/p syntax lets you find terms within the same paragraph. The terms can appear within the paragraph in any order.
pre/p	The pre/p syntax lets you find terms within the same paragraph. The terms must appear in the specified order.
w/s	The w/s syntax lets you find terms within the same sentence. The terms can appear within the sentence in any order.
pre/s	The pre/s syntax lets you find terms within the same sentence. The terms must appear in the specified order.

Full-text macros

The IBM StoredIQ full-text filter supports a set of predefined macros.

A macro is an abbreviated set of instructions that are used in common filter patterns. For example, the filter {SSN} finds any document that contains Social Security numbers.

Full-text macros have the following characteristics:

- A macro is an alphanumeric name that is enclosed in curly braces, for example, {SSN}.
- A macro's name is case insensitive. For example, the macro {ssn} is the same as the macro {SSN}.
- The macro's value is replaced in the macro's full-text query. Then, the filter is parsed and run.
- Any unrecognized macro results in a syntax error.
- Regular expressions, that is named character classes, do not match macros. That is, a regular expression like \p{Lower} (matches any lowercase ASCII character) is not recognized as a macro.

The following macros are defined within the full-text feature of IBM StoredIQ. Each of these macros is a high-confidence macro. It means that the filter `{SSN}` finds any document that contains Social Security numbers with high confidence that the matches really are Social Security numbers. To find anything that might be a Social Security number, use the low-confidence macro `{SSN-1c}`.

Macros can be combined with any other full-text expression, such as `{SSN} w/5 "John Doe"`.

- `{SSN}`: Social Security numbers, which can include dashes and numbers
- `{USPhone}`: US phone numbers, which can include dashes, dots, parentheses, numbers, and letters. This macro can filter for mnemonics like 1-800-flowers.
- `{InternetAddress}`: Legal IP addresses, either as numbers or as names, which can include dots, numbers, and letters.
- `{EmailAddress}`: Legal email addresses, which can include @, dots, numbers, and letters.
- `{CCAmex}`: American Express credit-card numbers, which can include dashes and numbers.
- `{CCDiners}`: Diners Club credit-card numbers (including enRoute credit-card numbers), which can include dashes and numbers.
- `{CCDiscover}`: Discover credit-card numbers, which can include dashes and numbers.
- `{CCJCB}`: JCB credit-card numbers, which can include dashes and numbers.
- `{CCMC}`: MasterCard credit-card numbers, which can include dashes and numbers.
- `{CCVisa}`: Visa credit-card numbers, which can include dashes and numbers.
- `{CreditCard}`: Any credit-card number from the listed providers.

Indexed annotations

To search for indexed annotations (terms indexed when a cartridge is applied), you can create filters with the query format `ia:filter_term`.

You can search by indexed annotations after the system administrator deployed the cartridge and created a Step-up Analytics job with the cartridge, and you ran the respective Step-up Analytics action on an infoSet. Set up an indexed annotation query by creating a full-text filter with a `ia:filter_term` expression.

Contact your system administrator for a list of deployed cartridges, their supported search results, and their associated actions. You can also find out about the supported results by checking the **Action Details** section of a selected Step-up Analytics action.

The IBM provided cartridges that you can download from Fix Central support these indexed annotation terms:

- The cartridge that is based on regular expressions:
`ia:BankAccountNumber, ia:NationalIdNumber, ia:PassportNumber, ia:PhoneNumber, ia:EmailAddress, ia:IPAddress, ia:TaxIdNumber, ia:SocialSecurityNumber`
- The cartridge that is based on natural language processing:
`ia:Date, ia:Person, ia:Address, ia:PhoneNumber, ia:Organization, ia:Location`

The searches are case-sensitive. If the supported language field is populated, only documents in those languages are searched.

Regular expressions

To search for patterns, you can create full-text filters with regular expressions. The format for a regular expression query is `re:"regex"`.

For DataServer - Distributed, `re:` queries basically follow the Lucene regular expression syntax as described here:

The character classes `\d`, `\D`, `\w`, and `\W` are supported, POSIX character classes such as `\p{Alpha}` or `\p{Alpha}` aren't. Complex constructs such as backreferences or lookahead are also not supported.

For DataServer - Classic, `re:` queries basically follow the Java regular expression syntax as described here: [Java regex pattern](#)

Patterns in IBM StoredIQ are always anchored at the beginning. For example, the string `abcde` would be a match for the query `re: "ab.*"` but not for the query `re: "bcd"`.

The IBM StoredIQ full-text index is a tokenized index. It does not include spaces or other special characters such as `/`, `*`, `\`, `@`, `\`, `#`, `"` and others. Therefore, you cannot create regular expressions that try to match spaces as part of a string.

For example, if you wanted to identify eight numbers in groups of four separated by a space like `1234 5678`, you would probably try the following regular expressions:

- `re: "\d{4} \d{4}"`
- `re: "\d{4} \s \d{4}"`
- `re: "\d{4} ( ) \d{4}"`
- `re: "\d{4} [ ] \d{4}"`

However, neither of these regular expression would yield a result. Instead, you can consider the string to be two 4-digit words. You could then use proximity search to join them:

```
re: "\d{4}" w/0 re: "\d{4}"
re: "\d{4}" pre/0 re: "\d{4}"
```

This type of query is called a *low confidence query* because it is more likely to produce false positives. In addition, such queries consume considerable processing time.

In contrast, indexed annotations from a regular expression cartridge can contain spaces and special characters because a cartridge is applied to the source text and not to the tokenized text in the index. So, depending on your use case, you might want to create a regular expression cartridge, run it on your data, and then use `ia:` queries instead of `re:` queries to find such strings.

Extended ASCII characters

Extended ASCII characters are handled several different ways in the full-text filter. Some are special characters while others are treated as spaces. Some are treated as letters that depend on the spaces around the characters.

Special characters or punctuation are not indexed and cannot be searched for in filters. The following table describes what behavior applies to these characters in the filter.

Table 5: Extended ASCII Character syntax

Extended ASCII Character Syntax	Description
Special characters	Some special characters generate syntax errors because they cannot be processed. Either remove the character from the filter or surround the filter with single quotation marks. For example, a filter for <code>hello!</code> without quotations generates an error. However, if the filter is <code>'hello!'</code> (with single quotation marks), the query can be processed. In general, special characters are treated as a space in a filter and they are not indexed. For example, <code>Hello!</code> is indexed as <code>Hello</code>. Searching for <code>'Hello!'</code> returns zero object. To search for <code>'hello'</code>, data object that has the word <code>hello!</code> or <code>hello</code> is returned. However, there are some exceptions, and they are discussed later in this table Most special characters can also be searched for with quotation marks. For example, <code>"12/24/2014"</code> finds <code>12/24/2014</code>. The difference between quotation marks and single quotation marks is that quotation marks also have another function: to find words that are next to each other. Because of this, special characters that are used elsewhere in the syntax, such as <code>*</code> and <code>?</code>, are treated as the full-text operators for wildcards instead of as the characters <code>*</code> and <code>?</code>. These special characters generate errors unless surrounded by quotation marks: <code>@ / & : ! () { } [] ^ ~ , # =</code>
Period (.)	A period with no space on either side is treated like a letter. For example, a filter for <code>172.16.0.46</code> finds the whole number of <code>172.16.0.46</code>, not partial numbers of <code>172</code>, <code>16</code>, <code>0</code>, <code>46</code>. A period with a space on either side is treated as a space and ignored. For example, a filter for <code>"1234 Mahana St."</code> ignores the period and just finds <code>1234 Mahana St</code> only without a period.
At (@)	An <code>@</code> symbol is indexed like a letter when there are no spaces on either side. For example, a filter for <code>jdoe@yourcompany.com</code> finds the entire email address. An <code>@</code> symbol with a space on either side or one side is treated as a space and ignored. For example, <code>@home</code> finds <code>home</code> in the data object.
Hyphen (-)	A hyphen is indexed like a letter when there are no spaces on either side. For example, a filter for <code>012-44-5668</code> finds the entire number, not <code>012</code>, <code>44</code>, <code>5668</code> as separate numbers. A hyphen with a space on either side is treated as a space and ignored.
Slash (/)	A slash is indexed like a letter when there are no spaces on either side. For example, a filter for <code>10/30/1940</code> finds the entire expression <code>10/30/1940</code> (and not <code>10</code> <code>30</code> <code>1940</code>) as separate numbers.
Ampersand (&)	An ampersand is indexed like a letter when there are no spaces on either side. For example, a filter for <code>S&P</code> finds <code>S&P</code> in the data objects. However, a filter for <code>Proctor & Gamble</code> without quotation marks finds either <code>Proctor</code> or <code>Gamble</code>. A filter with quotation marks <code>"Proctor & Gamble"</code> finds the phrase <code>Proctor Gamble</code> only.
Colon (:	A colon is considered a special character. It is treated as a space in filter and is not indexed. For example, <code>21:35:15</code> is indexed as three consecutive words, that is, <code>21</code>, <code>35</code>, and <code>15</code>. Searching for <code>'21:35:15'</code> returns zero object. To search for <code>'21 35 15'</code>, data object that has the word <code>21:35:15</code> or that has three consecutive words as <code>21</code>, <code>35</code>, <code>15</code> is returned. Specifying terms with colons without the single quotation marks results in an invalid syntax error.

Table 5: Extended ASCII Character syntax (continued)	
Extended ASCII Character Syntax	Description
Apostrophe (')	The apostrophe in the term is indexed as a whole word. For example, a filter for patient 's returns results for patient 's but not for patient. A filter for the name O 'Hara returns results for O 'Hara. An apostrophe is treated as a single quotation mark, so it must always be quoted with double quotation marks.
Comma (,)	Commas are left in when surrounded by numerals, and those numerals make a legitimate number. For example, "1,000" is indexed as 1,000. To search this term, specify 1,000 in single quotation marks or quotation marks. Otherwise, it results in an invalid syntax error. For the term "10,00", it is indexed as 10 and 00. To search for this term, specify "10 00".
Number sign (#), dollar sign (\$), percent (%), equal sign (=), underscore (_)	These characters are treated as white space in the filter. For example, a search for \$100 returns results for the number 100 whether 100 refers to dollars, a percentage, or a whole number. The term 'JAVA_HOME' is indexed as 'JAVA HOME'. To search for this term, specify JAVA HOME in quotation marks.

Attribute condition reference

The following section explains how to type in a reference to an attribute condition.

If you always select an attribute from the custom filters or one of the attribute groups, IBM StoredIQ takes care of the attribute reference for you. If you do not select an attribute from the custom filters or one of the attribute groups, then you need to review and use this section.

A typed attribute reference is composed of several parts: `attribute-type path/attribute-name`. The `attribute-type` part gives the specific type of attribute to which `path` and `attribute-name` refer. This prefix is a short prefix of the attribute type. Attributes can have duplicate names, even within one folder.

Table 6: Attribute type prefix	
Attribute-type Prefix	Description
att: or @	This prefix is a system-defined attribute or user-defined entity.

`path` describes the folder path to the attribute. Generally, it is optional, but it is required if the attribute name alone is not unique. You do not need to type the full folder path to an attribute.

If an attribute name or folder path name has spaces or punctuation in it, then you must enclose the complete name in double or single quotation marks, not just a part. However, the forward slash (/) between folders or before the attribute name does not require the use of quotation marks. Do not enclose the attribute type prefix in quotation marks.

Specification of attribute conditions

Attributes, entity, tag, or query can be divided up in several ways.

- What type of values does the attribute have? Attributes can have values of number, string, enumeration, boolean, or date. An enumeration attribute is an attribute that has a limited set of possible values, which are shown in a list.

When an attribute is a compound attribute, each of its subattributes also has a value type: string, enumeration, boolean, or date. The subattributes can also be either sparse or dense. Currently, all

subattributes are single-valued. Compound attributes can also be sparse or dense. (Currently all compound attributes are sparse.)

- Do all objects have a value for this attribute? If the objects all have a value, it is known as a dense attribute. If the objects do not have a value, it is known as a sparse attribute. All objects have an `Object name` value, so `Object name` is a dense attribute. Only some objects have a `CC` attribute value, so `CC` is a sparse attribute.
- Can an attribute have multiple values within one object? Each object has one name, so `Object name` is a single-valued attribute. There might be multiple `To` values on an email, so `To` is a multivalued attribute. Compound attributes can be single-valued or multivalued. A compound attribute never exists where all of its subattribute values are missing.

These condition operators are applicable to each attribute category.

Note: For illustration purposes, some operators are shown in uppercase, and some are shown in lowercase. The parser recognizes operators in either uppercase or lowercase.

Conditions for string attributes

Tags are considered single-valued, sparse string attributes. They are not considered enumeration attributes, even though they have a limited set of possible values.

```
attribute = value or attribute is value
```

```
attribute != value or attribute is not value
```

```
attribute contains value
```

```
attribute does not contain value
```

```
attribute starts with value
```

```
attribute does not start with value
```

```
attribute ends with value
```

```
attribute does not end with value
```

```
attribute matches value
```

```
attribute does not match value
```

The value can be a single word, a double-quoted string, or a single-quoted string. For matches, the value is a regular expression within quotation marks.

Conditions for enumeration attributes

Some attributes have a limited set of predefined values that they can match, which are called enumeration attributes.

In this case, only `'='` and `'!='` operators are supported.

```
attribute = value or attribute is value
```

```
attribute != value or attribute is not value
```

A list of enumeration attributes is as follows:

- Container type
- Is-part-of

- Box reason
- Encryption type
- System processing state
- Content processing state
- Binary processing state
- System error code
- Content error code
- Binary error code
- Message type
- Latest review status
- Object category

Conditions for date attributes

The conditions for date attributes are listed as follows.

`attribute = value` or `attribute is value`

`attribute != value` or `attribute is not value`

`attribute < value`

`attribute <= value`

`attribute > value`

`attribute >= value`

`attribute between value1 and value2`

`attribute not between value1 and value2`

The value can have one of two forms: an explicit date or a relative date that is offset from today.

An explicit date is in the form of YYYY-MM-DD, for example, 2009-03-08.

- Other date formats are supported.
- Time is not supported.
- The dates are all interpreted as midnight of the listed date.

A relative date offset is specified as a positive number, followed by a required date units, and the ago or ahead keyword. The ago keyword is the default and means a date in the past. The ahead keyword means a date in the future. For example, 3 months ago, 3 days ahead, or 3 weeks ahead are legal relative dates. The allowed date units are as follows:

- days
- weeks
- months
- years

This function is also supported:

- -3 days = 3 days ago
- +3 weeks = 3 weeks ahead

Note: Months are considered as 30 days; years are considered as 365 days.

- Normally, relative date offsets are in the past, but some dates can be in the future, such as a `Retention expiration date`. For these references, use the `ahead` keyword instead of `ago`.
- Relative date offsets can also be given as a positive or negative number, such as `-3 days` or `+3 months` without the `ago` or `ahead` keyword. In this case, negative numbers are considered in the past and positive numbers are considered in the future.

Conditions for number attributes

Conditions for number attributes are listed as follows.

```
attribute = value or attribute is value
```

```
attribute != value or attribute is not value
```

```
attribute < value
```

```
attribute <= value
```

```
attribute > value
```

```
attribute >= value
```

```
attribute between value1 and value2
```

```
attribute not between value1 and value2
```

The value is a positive or negative integer. Size attributes (`Size` and `Size (on disk)`) can take an optional size unit after the number. These amounts are the legal size units.

- `b` or `bytes` for single bytes
- `k` or `kb` for kilobytes
- `m` or `mb` for megabytes
- `g` or `gb` for gigabytes
- `k` or `kb` for kilobytes

Note: The default size unit is bytes. Size units are based on multiples of 1024, not 1000.

Event duration can have time units. These time amounts are the legal size units:

- `m` or `minutes` for minutes
- `h` or `hours` for hours
- `d` or `days` for days

Conditions for sparse attributes

The conditions for multivalued and single valued sparse attributes are listed as follows.

```
attribute APPEARS  
attribute NOT APPEARS or attribute DOES NOT APPEAR
```

These conditions specify whether the attribute has any values for matching objects. In this case, it means that one of the subattributes has a value on this object. Tags are considered single-valued, sparse string attributes.

It is not an error to specify these conditions on dense attributes. However, if you do, the parser warns you that the condition is always `true` or `false`. These operators can also apply to compound attributes. In this case, it means that one of the subattributes has a value on this object.

Note: As a backward compatible measure, if `att: attribute appears` is given on a dense, boolean attribute, it is converted into `att: attribute = true`. If `not appears` is given, it is converted into `att: attribute = false`.

Conditions for compound attributes

Conditions for compound attributes are listed as follows.

```
attribute WHERE <sub-attribute-condition>
attribute WHERE (<sub-attribute expression>)
```

A `<sub-attribute-condition>` is just like any of the attribute conditions that are described except it is on a subattribute of the compound attribute. Additionally, a `<sub-attribute-condition>` can be just the subattribute name. It is equivalent to the subattribute **APPEARS**.

A `<sub-attribute-expression>` is any combination of `<sub-attribute-condition>`'s combined with **AND**, **OR**, or **NOT**. This expression must be enclosed in parentheses. The `<sub-attribute-expression>` can also be empty. In this case, it is the same as attribute **APPEARS**.

This example uses **WHERE** expressions:

```
att: "Site owner (SharePoint)" WHERE name = "John Smith"
att: "Site owner (SharePoint)" WHERE (name = "John Smith" AND url APPEARS )
att: "Site owner (SharePoint)" WHERE (name = "John Smith" OR url NOT APPEARS)
att: "Site owner (SharePoint)" WHERE ( )
att: "Collaborator (Box)" WHERE "Collaborator Role" = "editor"
```

Note: This expression is the same as `att: "Site owner (SharePoint)" APPEARS`.

```
att: "Site owner (SharePoint)" WHERE (name)
```

Note: This expression is the same as `att: "Site owner (SharePoint)" WHERE name APPEARS`.

Conditions for multivalued attributes

Conditions for multivalued sparse or dense attributes are listed as follows.

```
attribute APPEARS operator number
```

```
attribute APPEARS UNIQUE operator number
```

These operators can also apply to compound attributes, and the `operator` can be one of the following operators:

- `= >`
- `>=`
- `<`
- `<=`

The `operator number` specifies a condition on the number of values present for the attribute on any matching object.

- If `operator number` is not given, then `> 0` is assumed.
- If **UNIQUE** is given, it specifies the number of unique values present for the attribute on any matching object.

It is not an error to specify these conditions on single-valued attributes. However, if you do, the parser warns you that the condition is always `true` or `false`.

Access mode attribute

Specifies an attribute that can be used in a search expression for creating infosets that are based on certain file attributes that are collected during harvests.

Access mode attribute supports these operators: `APPEARS`, `NOT APPEARS`, and `WHERE`.

The `Subject` subattribute supports these operators: `IS`, `IS NOT`, `=`, `!=`, `any()`, and `all()`.

The `Readable` and `Writable` subattributes both support these operators: `IS`, `IS NOT`, `=`, and `!=`.

For example,

- To find data objects for which you collect or maintain information about the read or write file system flags, enter `att:"Access Mode" appears`. This search returns every system-level object on CIFS or NFS volumes.
- To find data objects that are writable by either of the three types of subjects, enter `att:"Access Mode" where (Subject is any (Owner, Group,Others) and Writable is True)` or `att:"Access Mode" where (Writable is False)`. These two examples are semantically identical. If you think of the `r/w/x` flags as separate groups of flags (that is, `rw` | `rx` | `rw`), where one is for the Owner, one is for the Group that a user might be associated with, and one is for Others in that order, then the `Readable` or `Writable` subattributes are evaluated independently for each three group of flags.

The `Subject` subattribute (if present) then just defines which group out of the three must be considered or ignored. The lack of the `Subject` subattribute indicates that the objects must be considered responsive to the expression if the `Readable` or `Writable` conditions (or any combination of them) evaluate to `True` in ANY of the three groups.

- To find data objects that are read-only by everyone, enter `att:"Access Mode" where (Subject is all (Owner, Group, Others) and Readable is True and Writable is False)`. The subattributes are all ANDed together and the `Subject` subattribute lists every possible option with the `all` qualifier.
- To find data objects that are read-only by at least one type of subject, enter `att:"Access Mode" where (Subject is any (Owner, Group, Others) and Readable is True and Writable is False)` or `att:"Access Mode" where (Readable is True and Writable is False)`. Again, the two examples here produce identical results for the same reason as explained in the previous bullet point.
- To find data objects that are read-only by the owner, enter `att:"Access Mode" where (Subject = Owner and Readable is True and Writable is False)`. You care only about the owner in this case, so you explicitly call `Owner` and only `Owner` out.
- To find data objects that owner can either read or write (not necessarily both), enter `att:"Access Mode" where (Subject = Owner and (Readable is True or Writable is True))`.
- To find data objects that can be written by the owner but is read-only by the group, enter `att:"Access Mode" where ((Subject = Owner and Writable = True) and (Subject = Group and Writable is False))`.

Type of object to match

For any attribute condition, you can specify what object that condition must match.

You can specify `IN object-type` after the expression. For example, in `att: Extension = doc=in system`, the object must be a doc file at the system level. The condition-scope can be any of these object types here.

Table 7: Available object types	
Available Object Types	Description
all	The matching object can be any type of object. This object is the default.
container	The matching object must be a container object that can have other objects nested inside it. File-archive, email-archive, email, calendar, contact, and task objects are included.
simple	The matching object must not be a container object.
system	The matching object must be a system-level object.
contained	The matching object must be nested within some other container object.

Only one condition-scope can apply to any condition. If you give multiple condition-scopes, the parser issues a warning.

Multiple matching values for operators

You can give multiple matching values that use the following operators.

- =
- !=
- contains
- not contains
- matches
- not matches
- starts with
- not starts with
- ends with
- not ends with

To specify multiple matching values for these operators, give the keyword `any` or `all` after the operator, then a list of values in parentheses. The `any` keyword means that any of the values are a match. The `all` keyword means that all the values must be present. The `all` keyword is only meaningful for multivalued attributes. For example:

Table 8: Keyword examples	
Example	Result
<code>att:"System metadata/Owner" = any (joe, bob, sally)</code>	Owner can be joe, bob, or sally.
<code>att:Size = any (1mb, 2mb, 1gb)</code>	Size can be 1mb, 2mb, or 1gb.
<code>att: "Modified date" = any (1 week ago, 2 months ago)</code>	
<code>att: "Accessed date" = any (2009-09-19, 2009-09-20, 2009-09-21)</code>	
<code>att:To = all (joe, bob)</code>	"To" must contain both joe and bob

For `any`, the operator must be true for at least one of the values. It is as if the user spelled out `attribute oper val1 OR attribute oper val2 OR ...`

- For all, the operator must be true for all the values. It is as if the user spelled out `attribute oper val1 AND attribute oper val2 AND...`
- For the NOT various operators, this syntax is the same as using NOT around the entire expression and the corresponding positive operator. For example, `att:Size != any( 1mb, 2mb, 2gb )` is the same as `NOT (att:Size = any( 1mb, 2mb, 1gb ))`. The NOT matches all documents that do not match the positive operator.
- It is not an error to specify all on single-valued attributes. However, if you do, the parser warns you that the condition is always false.

At-least expressions

On occasion, it is helpful to specify that some number of conditions must be true before an object is considered to match. These expressions are called `at-least` expressions because at least X subexpressions must match an object before the object matches the overall expression.

The syntax for an `at-least` expression is to give the keywords `"at least"` followed by a number, and then by a list of expressions in parentheses. For example:

```
at least 2 (att: size=1mb, att: "system metadata/owner"=joe, att:
"object name"=foo.doc, att: "modified date">2009-03-28 )
```

This example specifies that at least two of the four conditions must be true before any object is considered to match the overall expression.

Each subexpression in the list can be arbitrarily complex, but normally they are one condition. Each subexpression in the list must be separated from the others with a comma.

Parent and child conditions

This parent and child conditions represent advanced function that must be undertaken only by experienced administrators.

You might want to filter for objects whose parents or children match certain conditions. For example, find all emails to Joe in a `saved-emails.pst` file or find emails to Joe that have an attachment that is named `big-contract.pdf`. To specify these kinds of queries, give the conditions for the matching objects as normal and add a prefix `parent` or `child` to the conditions that apply to the parent or child objects. Here are the expressions for the two previous examples:

```
att:"any address"=Joe and parent att: "object name"="saved-
emails.pst"
```

```
att:To=Joe and child att:"object name"="big-contract.pdf"
```

If there is more than one expression that must match the parent or child objects, you must put all the conditions inside parentheses, for example:

```
att:"any address"=Joe and parent(att: "object name"="saved-
emails1.pst" or att:"object name"="saved-emails2.pst")
```

```
att:To=Joe and child (att:"object name"="big-contract.pdf" and
att:"modified date">2009-09-07)
```

It is also possible to specify conditions on the object level of parent or child objects. An object level is the nesting level of an object. System-level objects are at level one, immediately nested objects are at level two, and so on. For example, if you want to find the non-nested emails to Joe in `saved-emails.pst`, you can do:

```
att:"any address"=Joe and att:"object level" = 2
and parent
att:"object name"="saved-emails.pst"
```

It specifies that the emails to Joe must be at object level 2 and be inside saved-email.pst.

It is also possible to specify the object level of the parent or child object relative to the other. For example, to find the parent object of any file that is named nested-file.doc, you can specify it this way:

```
child att: "object name" = "nested-file.doc" parent-level = child-level - 1
```

It means that nested-file.doc can itself be at any level, and the query results are the direct parent of those files.

The general form of relative parent-child level conditions is listed here.

```
parent-level operator child-level +|- number  
child-level operator parent-level +|- number
```

```
parent-level operator child-level +|- number  
child-level operator parent-level +|- number
```

- Plus (+) in the +|- indicates that the parent-child must be more deeply nested than the other objects.
- Minus (-) means the parent-child must be less deeply nested than the other objects.

The operator can be any of these comparisons:

Numeric Comparisons for Operators					
=	!=	<	<=	>	>=

No-inheritance conditions

The no-inheritance conditions represent advanced function that must be undertaken only by experienced administrators.

Normally, the search engine allows some objects to inherit the values of their parent objects. Most of the time this behavior is intuitive, but there are times when this behavior must be turned off. When attribute inheritance is not wanted, prefix the expression with `noinheritance`, and attribute inheritance is turned off for that expression. For example, this expression finds emails to Joe containing stock option in the body or any attachment:

```
att: To=joe and "stock option"
```

However, if you want to find emails to Joe with stock option in the email body itself, then you must turn off inheritance for that part of the expression:

```
noinheritance ( att: To=joe and "stock option" )
```

Notices

This information was developed for products and services offered in the U.S.A. This material may be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
J46A/G4
555 Bailey Avenue

San Jose, CA 95141-1003
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows: © (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. 2004, 2010. All rights reserved.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Privacy policy considerations

IBM Software products, including software as a service solutions, ("Software Offerings") may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering's use of cookies is set forth below.

[Depending upon the configurations deployed, this Software Offering may use [session] [and] [persistent] cookies that collect each user's [name,] [user name,] [password,] [profile name,] [position,] or [other personally identifiable information] for purposes of [session management,] [authentication,] [enhanced user usability,] [single sign-on configuration] [or other usage tracking or functional purposes.] These

cookies [cannot be disabled] [can be disabled, but disabling them will also [likely] eliminate the functionality they enable].

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, See IBM's Privacy Policy at <http://www.ibm.com/privacy> and IBM's Online Privacy Statement at <http://www.ibm.com/privacy/details> the section entitled "Cookies, Web Beacons and Other Technologies" and the "IBM Software Products and Software-as-a-Service Privacy Statement" at <http://www.ibm.com/software/info/product-privacy>.

Terms and conditions for product documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

Applicability

These terms and conditions are in addition to any terms of use for the IBM website.

Personal use

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM.

Commercial use

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

Trademarks

IBM, the IBM logo, and [ibm.com](http://www.ibm.com) are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" <http://www.ibm.com/legal/copytrade.shtml>.

Adobe and PostScript are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

UNIX is a registered trademark of The Open Group in the United States and other countries.

VMware, VMware vCenter Server, and VMware vSphere are registered trademarks or trademarks of VMware, Inc. or its subsidiaries in the United States and/or other jurisdictions.

Index

A

- access mode attribute [17](#)
- ASCII characters [10](#)
- at-least expressions [19](#)
- attribute [17](#)
- attribute condition [12](#)
- attribute conditions [12](#)

C

- compound attribute [12](#)
- condition operators [12](#)
- conditions
 - compound attributes [16](#)
 - date attributes [14](#)
 - enumeration attributes [13](#)
 - multivalued sparse or dense attributes [16](#)
 - no-inheritance [20](#)
 - number attributes [15](#)
 - parent and child conditions [19](#)
 - sparse attributes [15](#)

D

- dense attribute [12](#)

E

- expression language [5](#)

F

- filter
 - full-text syntax [5](#)
- filter condition reference [5](#)
- full-text filter [8](#)
- full-text macro [8](#)

I

- Insights
 - concepts [2](#)
 - evaluating [3](#)
 - facets [2](#)
 - filter mode [2](#)
 - results [2](#)
 - search [2](#)
 - searching [3](#)

L

- legal
 - notices [21](#)
 - trademarks [23](#)

M

- macro [8](#)
- multiple matching values [18](#)
- multivalued attribute [12](#)

N

- no-inheritance conditions [20](#)
- notices
 - legal [21](#)

O

- operators [18](#)

P

- parent and child conditions [19](#)

Q

- queries
 - regex [9](#)

R

- re: queries [9](#)
- reference [12](#)
- regex [9](#)
- regular expressions [9](#)

S

- single-valued attribute [12](#)
- sparse attribute [12](#)
- specification of attribute conditions [12](#)
- string attributes [13](#)

T

- trademarks [23](#)

