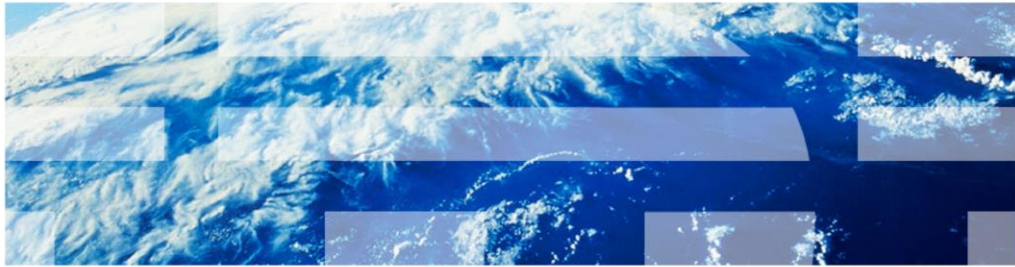


# DB2 Text Search

## Search features



This presentation describes various search features available in DB2® Text Search.

## Overview

- Built-in advanced search functionality combined with DB2 optimizer (SQL, SQL/XML and XQuery support)
  - SQL functionality
    - CONTAINS() and SCORE()
  - XPath-like syntax to search XML documents
  - XQuery db2-fn:xmlcolumn-contains function
- Search in plain text, HTML, XML, and rich text/proprietary formats
  - Use wildcards, boolean expressions, phrase search, optional terms, synonyms, and so on
  - Get scores
    - How well does document match search criteria?
  - Set query language, result limits
- Linguistic processing for 20+ languages/locales

DB2 Text Search allows you to issue SQL and XQuery statements to search the data stored in your DB2 database. The built-in SQL functions, CONTAINS() and SCORE(), are used to search the text indexes and are seamlessly integrated with DB2 optimizer to automatically select the best optimization plan according to the expected search results.

Using the XPath syntax subset and the built in search function db2-fn:xmlcolumn-contains(), it is possible to perform a structural full text search in XML documents. It allows you to search specific portions of an XML document.

Supported document formats include Text, HTML, XML and, with the DB2 Accessories Suite, rich text documents are searchable.

The search scope can either be extended or reduced using wild cards, Boolean expressions, phrase search, optional terms, synonyms and so on. Using the SCORE() function, you can find out how well a document meets the search criteria compared to other documents in the same text index. Other search features include setting the query language in the search query to explicitly specify the locale to DB2 Text Search, result limit can be specified to limit the returned result set.

DB2 Text search also supports linguistic processing for 20+ languages to retrieve only the relevant search results.

The next slides provide more details and examples about the search functionality.

## CONTAINS()

- Used in SQL query to perform word or phrase search on text index
- Syntax
  - CONTAINS (text-index-column-name, search-argument,<optional-search-argument-options>)
  - text-index-column-name : column name on which text index is created
  - search-argument : An expression containing terms to be searched
  - search-argument-options is optional
    - QUERYLANGUAGE locale
    - RESULTLIMIT value
    - SYNONYM OFF | ON
- Example

```
SELECT author,title FROM books WHERE CONTAINS(title,'database')=1
AUTHOR          TITLE
-----
Samantha Smitt  The Database Compendium
John Doe        The Database Book
2 record(s) selected.
```

The SQL function CONTAINS() searches the text search index using the criteria specified in the search query. It returns an integer value of one if the document contains the text, otherwise it returns zero. This is the most common way of performing a search with DB2 Text Search. Use this where standard SQL is used and combine it with other conditions using the WHERE clause.

The CONTAINS() function takes two mandatory parameters; the column name on which the text index is created and the actual search criteria that should be applied. There is a third optional parameter used to specify the options for search criteria.

There is an example provided on this slide to show how to search the 'title' column in the 'books' table for a specific search term 'database' using the CONTAINS() SQL function.

## SCORE()

- Used in SQL query to obtain relevance of a document
- Syntax
  - SCORE (text-index-column-name, search-argument,<optional-search-argument-options>)
    - text-index-column-name : the column name on which text index is created
    - search-argument : An expression containing the terms to be searched
    - search-argument-options is optional
      - QUERYLANGUAGE locale
      - RESULTLIMIT value
      - SYNONYM OFF | ON

- Example

```
SELECT title FROM books WHERE CONTAINS(title,'database')=1 ORDER BY
SCORE(title,'database') DESC
TITLE
-----
The Database Compendium
The Database Book
2 record(s) selected.
```

The SQL function SCORE() can be used to obtain the relevancy of a found text document. It returns a double precision floating-point number between zero and one.

The SCORE() function takes two mandatory parameters; the column name on which the text index is created and the actual search criteria that should be applied. There is a third optional parameter used to specify the options for search criteria.

There is an example displayed on this slide to show how to use the SCORE() function to find out which documents qualifies the search criteria better compared to the others.

## Xmlcolumn-contains()

- Used in XQuery to perform word or phrase search on XML column types
- Syntax
  - `xmlcolumn-contains(string-literal, search-argument, <optional-search-literal-options>)`
    - `string-literal` : the XML column name on which text index is created
    - `search-argument` : An expression containing the terms to be searched
    - `optional-search-literal-options` is optional
      - `QUERYLANGUAGE locale`
      - `RESULTLIMIT value`
      - `SYNONYM OFF | ON`
- Example

```
xquery db2-fn:xmlcolumn-contains('BOOKS.BOOKINFO', 'Database')/bookinfo/title
1
-----
<title>The Database Compendium</title>
<title>The Database Book</title>
2 record(s) selected.
```

5

DB2 Text Search - Search features

© 2013 IBM Corporation

The XML function `xmlcolumn-contains()` can be used in XQuery statement and operates on XML columns only. It returns XML documents containing the search term or phrase.

This function takes two mandatory parameters; the column name on which the text index is created and the actual search criteria that should be applied. There is a third optional parameter used to specify the options for search criteria.

There is an example displayed on this slide to show how to use the `db2-fn:xmlcolumn-contains` function in an XQuery to find the XML documents stored in the column 'books.bookinfo' that contain the search term 'Database'.

## Search XML documents

- Example

```
SELECT author FROM books WHERE  
contains(bookinfo, '@xpath: '/bookinfo/story [.contains("Database")]''')=1
```

```
AUTHOR
```

```
-----  
Samantha Smitt  
John Doe
```

```
2 record(s) selected.
```

- Example

```
xquery db2-fn:xmlcolumn-contains  
( 'BOOKS.BOOKINFO', '@xpath: '/bookinfo/story [.contains("Database")]''')/bookinfo/title  
1
```

```
-----  
<title>The Database Compendium</title>  
<title>The Database Book</title>
```

```
2 record(s) selected.
```

The examples displayed on this slide show two different ways of searching XML documents using the XPath expression in CONTAINS() SQL function and db2-fn:xmlcolumn-contains() XML function in XQuery context.

## Search features (1 of 2)

- Boolean operators
  - For conjunction (AND), disjunction (OR), and exclusion (NOT) of search term
    - Example
      - SELECT title FROM books WHERE CONTAINS (story, '(nature OR moon) NOT forest')=1
- Fuzzy search
  - Search for documents that contain similar terms
    - Example
      - SELECT author, story FROM books WHERE CONTAINS (story, 'analytics~0.7') = 1
- Proximity search
  - Search for documents that contain terms in specified distance
    - Example
      - SELECT author, story FROM books WHERE CONTAINS (story, '"analytics DB2"~5') = 1
- Search for special characters
  - Example
    - SELECT author, story FROM books WHERE CONTAINS (story, 'AS/400') = 1

7

DB2 Text Search - Search features

© 2012 IBM Corporation

Now that you have some understanding about the basic search functionality, the next few slides discuss other search features offered by DB2 Text Search.

You can use the boolean operators – AND and OR to combine several search terms. You can use the boolean operator "NOT" to exclude any particular text document from the search. The example displayed on this slide searches for documents that contain the terms 'nature' or 'moon' but not 'forest'.

You can use the Fuzzy search to find the documents that contain words which are spelled in a similar way to the search term. Use the tilde symbol at the end of a term to do a Fuzzy search. The example displayed on this slide finds the documents that include the terms analytics, analysis, analyze, and so on. The fuzzy factor is 0.7 in this example, increase it further to include more results.

You can use the Proximity search to find the documents that contain terms in a distance of at most, x words. Use the tilde (~) symbol at the end of a phrase and specify the distance in words as a valid integer number. The example displayed on this slide finds the documents with terms 'analytics' and 'DB2' located within five words from each other.

The special characters can be searched in the same way as the other query terms. Some of them should be escaped in order to be searched.

## Search features (2 of 2)

- More information on searching for special characters
  - <http://pic.dhe.ibm.com/infocenter/db2luw/v10r5/topic/com.ibm.db2.luw.admin.ts.doc/doc/c0059608.html>
- Wildcard characters
  - Question mark (?) and asterisk (\*)
- Optional search
  - Using percentage (%)
- Exact search
  - Double quotation marks (")
- Weight or boosting of score values for search terms
  - Caret (^) character

For more information on searching for special characters, see the link displayed on this slide.

Increase the search scope by using wildcard characters. Use a question mark (?) to specify that a single character can be added to your search term. Use an asterisk (\*) to specify that any number of characters can be added to your search term.

You can use a percentage sign (%) to specify that a term or phrase is optional while executing the search.

For an exact search, use quotation marks (") around your search term or phrase to perform an exact search.

To give weight to or to boost score values for search terms, use the caret (^) character.



## Search features: Morphological and ngram for CJK

- N-gram
  - Index overlapping character sequences
    - Example

```
create index test.bookidx for text on test.books(story) index configuration (segmentation ngram)
select bookname from books where contains (story, '書十') = 1
BOOKNAME
-----
book1
1 record(s) selected
```
- Morphological
  - Index based on meaningful words
    - Example

```
create index test.bookidx for text on test.books(story) index configuration (segmentation
morphological)
select bookname from books where contains (story, '書十') = 1
BOOKNAME
-----
0 record(s) selected.
```

© 2013 IBM Corporation

DB2 Text Search provides dictionary packs to support the linguistic processing of documents and queries. In addition, n-gram segmentation is supported for languages such as Chinese, Japanese, and Korean.

There are two processing options for Chinese, Japanese, and Korean languages. First, is the n-gram segmentation option which is the default and indexes overlapping pairs of characters. The second is the morphological segmentation option. It uses a dictionary specific to the language for identifying word boundaries.

The example displayed on this slide shows the difference between morphological and n-gram segmentation with a search for a meaningless Chinese word.

## Optional search arguments

- QUERYLANGUAGE
  - Example
    - `SELECT author, story FROM books WHERE CONTAINS (story, 'cat', 'QUERYLANGUAGE=en_US') = 1`
- RESULTLIMIT
  - Example
    - `SELECT author, story FROM books WHERE CONTAINS (story, 'cat', 'RESULTLIMIT=10') = 1`
- SYNONYM
  - Example
    - `SELECT author, story FROM books WHERE CONTAINS (story, 'cat', 'SYNONYM=ON') = 1`

This slide displays the optional arguments that can be used in the search query.

QUERYLANGUAGE is used to explicitly specify the locale that DB2 Text Search uses to perform a search. If not provided, then the default is the locale of the text search index. In the example displayed on this slide, “en\_US” locale is used.

RESULTLIMIT is used to limit the search results where a high count is anticipated. In the example displayed on this slide, the result is limited to 10 records.

SYNONYM is used to enable or disable the usage of a synonym dictionary related to the text search index. The default is OFF. Set it to ‘ON’ to search for synonyms of the query terms, thus, improving the result.

## Linguistic processing

- Search term form, variants, inflections due to case, gender, number, tense and so on
- Search term relevance
- Computer, computers
- Mouse, mice
- Good, better, best, well
- Drive, drives, drove, driven

– Example

```
SELECT pk,col FROM test WHERE CONTAINS(col,'driven')=1
PK      CO
```

-----

```
1 This bike is driven by the waiter
2 I have been driving this car for the last four hours
3 John drove to New York yesterday.
   3 record(s) selected.
```

DB2 Text Search uses a dictionary based approach to perform linguistic processing to return only a relevant result set. Hence, depending on the language, linguistic variations of query terms can be returned.

For example, a search for good, as per linguistic processing, returns the documents that match good or better or best or well.

The example shows that the search for the term 'driven' returns the documents that contain driven, driving or drove.

For more details on linguistic processing, visit the Information Center.

## Performance considerations during search

- Disable Statement concentrator (STMT\_CONC ) on DB2 Text Search V9.7 and higher
  - <http://www-01.ibm.com/support/docview.wss?uid=swg21607350>
- Set MALLOCOPTIONS environmental variable only on AIX® for DB2 Text Search V9.7 and V10.1
  - <http://www-01.ibm.com/support/docview.wss?uid=swg21647558>
- More information on enhancing performance for full-text queries
  - [http://pic.dhe.ibm.com/infocenter/db2luw/v10r5/topic/com.ibm.db2.luw.admin.ts.doc/doc/t\\_performanceconsiderations.html](http://pic.dhe.ibm.com/infocenter/db2luw/v10r5/topic/com.ibm.db2.luw.admin.ts.doc/doc/t_performanceconsiderations.html)

It is expected that enabling the statement concentrator results in the generation of a sub-optimal plan for Text Search queries and hence, performance degradation is noticed. This parameter should be disabled for DB2 Text Search queries running on V9.7 or higher releases. For more details, go to the technote located at the link provided on this slide.

Another contributing factor for performance degradation can be due to a single memory heap allocation. This can be avoided by setting MALLOCOPTIONS environmental variable. It is applicable only on the AIX platform for V9.7 and V10.1. This variable setting is not required in V10.5 since DB2 automatically sets it during instance startup. For more details, go to the technote located at the link provided on this slide.

## Additional resources

- IBM Education Assistant modules
  - DB2 Text Search - [Overview](#)
  - DB2 Text Search - [Installation, configuration and upgrade](#)
  - DB2 Text Search - [Indexing](#)
  - DB2 Text Search - [Administration commands](#)
  - DB2 Text Search - [Command line tools](#)
- IBM Information Center
  - <http://pic.dhe.ibm.com/infocenter/db2luw/v10r5/topic/com.ibm.db2.luw.admin.ts.doc/doc/c0051296.html>

This slide displays additional IBM Education Assistant modules related to DB2 Text Search and it provides the link to the IBM Information Center.

## Trademarks, disclaimer, and copyright information

IBM, the IBM logo, ibm.com, AIX, AS/400, and DB2 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 other IBM trademarks is available on the web at "[Copyright and trademark information](http://www.ibm.com/legal/copytrade.shtml)" at <http://www.ibm.com/legal/copytrade.shtml>

Other company, product, or service names may be trademarks or service marks of others.

THE INFORMATION CONTAINED IN THIS PRESENTATION IS PROVIDED FOR INFORMATIONAL PURPOSES ONLY. WHILE EFFORTS WERE MADE TO VERIFY THE COMPLETENESS AND ACCURACY OF THE INFORMATION CONTAINED IN THIS PRESENTATION, IT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. IN ADDITION, THIS INFORMATION IS BASED ON IBM'S CURRENT PRODUCT PLANS AND STRATEGY, WHICH ARE SUBJECT TO CHANGE BY IBM WITHOUT NOTICE. IBM SHALL NOT BE RESPONSIBLE FOR ANY DAMAGES ARISING OUT OF THE USE OF, OR OTHERWISE RELATED TO, THIS PRESENTATION OR ANY OTHER DOCUMENTATION. NOTHING CONTAINED IN THIS PRESENTATION IS INTENDED TO, NOR SHALL HAVE THE EFFECT OF, CREATING ANY WARRANTIES OR REPRESENTATIONS FROM IBM (OR ITS SUPPLIERS OR LICENSORS), OR ALTERING THE TERMS AND CONDITIONS OF ANY AGREEMENT OR LICENSE GOVERNING THE USE OF IBM PRODUCTS OR SOFTWARE.

© Copyright International Business Machines Corporation 2013. All rights reserved.