# Contents


<table>
<thead>
<tr>
<th>Chapter 1. Overview and concepts</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key concepts</td>
<td>3</td>
</tr>
<tr>
<td>Using an SQL scalar search function</td>
<td>6</td>
</tr>
<tr>
<td>Using a stored procedure search</td>
<td>7</td>
</tr>
<tr>
<td>Using an SQL table-valued function for searching</td>
<td>8</td>
</tr>
<tr>
<td>Additional concepts</td>
<td>9</td>
</tr>
<tr>
<td>Column transformation function</td>
<td>9</td>
</tr>
<tr>
<td>Instance services</td>
<td>9</td>
</tr>
<tr>
<td>Externally stored data</td>
<td>9</td>
</tr>
<tr>
<td>Administration views</td>
<td>10</td>
</tr>
<tr>
<td>Key features</td>
<td>10</td>
</tr>
<tr>
<td>DB2 Net Search Extender in the DB2 client/server environment</td>
<td>11</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Chapter 2. Installation</th>
<th>13</th>
</tr>
</thead>
<tbody>
<tr>
<td>System requirements</td>
<td>13</td>
</tr>
<tr>
<td>Installation on UNIX</td>
<td>14</td>
</tr>
<tr>
<td>Step 1 for UNIX: Install the product components</td>
<td>14</td>
</tr>
<tr>
<td>Step 2 for UNIX: Update the DB2 instance</td>
<td>14</td>
</tr>
<tr>
<td>Windows installation</td>
<td>15</td>
</tr>
<tr>
<td>Directory and file names</td>
<td>15</td>
</tr>
<tr>
<td>Installation verification</td>
<td>15</td>
</tr>
<tr>
<td>Installation verification on Windows</td>
<td>15</td>
</tr>
<tr>
<td>Installation verification on UNIX</td>
<td>16</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Chapter 3. User scenarios</th>
<th>17</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple example with the SQL scalar search function</td>
<td>17</td>
</tr>
<tr>
<td>Simple example with cache usage and stored procedure search</td>
<td>19</td>
</tr>
<tr>
<td>Simple example with the SQL table-valued function</td>
<td>20</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Chapter 4. Planning</th>
<th>23</th>
</tr>
</thead>
<tbody>
<tr>
<td>Directory locations and index storage</td>
<td>23</td>
</tr>
<tr>
<td>Document formats and supported code pages</td>
<td>23</td>
</tr>
<tr>
<td>User roles</td>
<td>24</td>
</tr>
<tr>
<td>Using the command line or DB2 Control Center interface for indexing</td>
<td>25</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Chapter 5. Net Search Extender instance services</th>
<th>27</th>
</tr>
</thead>
<tbody>
<tr>
<td>Starting and stopping DB2 Net Search Extender</td>
<td>27</td>
</tr>
<tr>
<td>Locking services</td>
<td>27</td>
</tr>
<tr>
<td>Using the locking services</td>
<td>27</td>
</tr>
<tr>
<td>Viewing a lock snapshot</td>
<td>29</td>
</tr>
<tr>
<td>Update services</td>
<td>30</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Chapter 6. Creating and maintaining a text index</th>
<th>31</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introducing the db2text commands</td>
<td>31</td>
</tr>
<tr>
<td>Enabling a database</td>
<td>32</td>
</tr>
<tr>
<td>Disabling a database</td>
<td>33</td>
</tr>
<tr>
<td>Creating a text index</td>
<td>34</td>
</tr>
<tr>
<td>Creating a text index on binary data types</td>
<td>36</td>
</tr>
<tr>
<td>Creating a text index on a nonsupported data type</td>
<td>36</td>
</tr>
<tr>
<td>Creating a text index for DATALINK data types</td>
<td>36</td>
</tr>
<tr>
<td>Installing the Data Links jar file</td>
<td>37</td>
</tr>
<tr>
<td>Creating a text index which the stored procedure search can use</td>
<td>38</td>
</tr>
<tr>
<td>Text indexes on views</td>
<td>42</td>
</tr>
<tr>
<td>Performance considerations</td>
<td>44</td>
</tr>
<tr>
<td>Maintaining text indexes</td>
<td>44</td>
</tr>
<tr>
<td>Updating and reorganizing a text index</td>
<td>45</td>
</tr>
<tr>
<td>Altering a text index</td>
<td>46</td>
</tr>
<tr>
<td>Clearing index events</td>
<td>47</td>
</tr>
<tr>
<td>Dropping a text index</td>
<td>48</td>
</tr>
<tr>
<td>Viewing text index status</td>
<td>48</td>
</tr>
<tr>
<td>Backing up and restoring indexes</td>
<td>49</td>
</tr>
</tbody>
</table>

| Chapter 7. Using DB2 Control Center            | 51 |
About this book

This book describes how to use the IBM DB2 Universal Database™ Net Search Extender to prepare and maintain a DB2® database for retrieving text data. It also describes how you can use the SQL functions provided to access and manipulate these types of data. By incorporating DB2 Net Search Extender’s functions in your program’s SQL statements, you can create powerful and versatile text-retrieval programs.

References in this book to "DB2" refer to DB2 UDB.

Who should use this book

This book is intended for DB2 database administrators who are familiar with DB2 administration concepts, tools, and techniques.

This book is also intended for DB2 application programmers who are familiar with SQL and with one or more programming languages that can be used for DB2 application programs.

How to use this book

This book is structured as follows:

This part gives an overview of DB2 Net Search Extender, describes how to install and set it up, and discusses planning considerations. It also describes how to prepare and maintain a DB2 database so that you can search for text.

Read this part if you are new to DB2 Net Search Extender and want to learn how to use the DB2 Net Search Extender functions to search for text.

“Part 2. Reference”
This part presents reference information for DB2 Net Search Extender functions and commands.

Read this part if you are familiar with DB2 Net Search Extender concepts and tasks, but need information about a specific DB2 Net Search Extender function or command.

“Part 3. Appendixes”
This part provides additional reference information for DB2 Net Search Extender. It contains information on migration, memory usage, views, document models, messages, and codes.
About this book

Read this part if you are familiar with DB2 Net Search Extender concepts and tasks, but need specific information about certain DB2 Net Search Extender concepts.

How to read the syntax diagrams

Throughout this book, syntax is described using the structure defined as follows:

- Read the syntax diagrams from left to right and top to bottom, following the path of the line. The symbol indicates the beginning of a statement.

The symbol indicates that the statement syntax is continued on the next line.

The symbol indicates that a statement is continued from the previous line.

The symbol indicates the end of a statement.

- Required items appear on the horizontal line (the main path).

- Optional items appear below the main path.

- If you can choose from two or more items, they appear in a stack. If you must choose one of the items, one item of the stack appears on the main path.

If choosing none of the items is an option, the entire stack appears below the main path.

A repeat arrow above a stack indicates that you can make more than one choice from the stacked items.
About this book

- Keywords appear in uppercase; they must be spelled exactly as shown. Variables appear in lowercase (for example, srcpath). They represent user-supplied names or values in the syntax.
- If punctuation marks, parentheses, arithmetic operators, or other such symbols are shown, you must enter them as part of the syntax.

Related information

**DB2 Universal Database Version 8**
- *IBM DB2 Universal Database Quick Beginnings Version 8* for DB2 Servers (GC09-4836), for DB2 Clients (GC09-4832), for DB2 Connect Personal Edition (GC09-4834), for DB2 Personal Edition (GC09-4838), and IBM Data Links Manager (GC09-4829-00). These books describe how to plan for, install, configure, and migrate to DB2 Universal Database on the appropriate platform.
- *IBM DB2 Universal Database Administration Guide Version 8* Planning (SC09-4822), Performance (SC09-4821), and Implementation (SC09-4820). These books describe how to design and implement a DB2 database.
- *IBM DB2 Universal Database Call Level Interface Guide and Reference, Volume 1 Version 8* (SC09-4849). This book describes how to develop applications that access DB2 databases using the DB2 Call Level Interface, a callable SQL interface that is compatible with the Microsoft ODBC specification.
- *IBM DB2 Universal Database Call Level Interface Guide and Reference, Volume 2 Version 8* (SC09-4850). This book describes how to develop applications that access DB2 databases using the DB2 Call Level Interface, a callable SQL interface that is compatible with the Microsoft ODBC specification.
- *IBM DB2 Universal Database Command Reference Version 8* (SC09-4828). This book describes how to use the DB2 command line processor and gives reference information about DB2 commands.

**IBM DB2 Universal Database Enterprise-Extended Edition, Version 8**
About this book

How to send your comments

Your feedback helps IBM to provide quality information. Please send any comments that you have about this book or other DB2 Extenders documentation. You can use the following method to provide comments:

• Send your comments by e-mail to swsdid@de.ibm.com. Be sure to include the name of the book, the part number of the book, the version of the product, and, if applicable, the specific location of the text you are commenting on (for example, a page number or table number).

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contacting IBM

If you have a technical problem, please review and carry out the actions suggested by the Troubleshooting Guide before contacting DB2 Customer Support. This guide suggests information that you can gather to help DB2 Customer Support to serve you better.

For information or to order any of the DB2 Universal Database products contact an IBM representative at a local branch office or contact any authorized IBM software remarketer.

If you live in the United States, then you can call one of the following numbers:

• 1-800-237-5511 for customer support
• 1-888-426-4343 to learn about available service options

Product information

If you live in the United States, then you can call one of the following numbers:

• 1-800-IBM-CALL (1-800-426-2255) to order products or get general information
• 1-800-879-2755 to order publications

http://www.ibm.com/software/data/db2/
The DB2 World Wide Web pages provide current DB2 information about news, product descriptions, education schedules, and more.

http://www.ibm.com/software/data/support/
The DB2 support Web pages provide access to frequently asked questions, fixes, books, and up-to-date DB2 technical information.

Note: This information may be in English only.
About this book

http://www.ibm.com/software/data/db2/extenders/
The DB2 Extenders Web pages provide information on all the currently available DB2 Extenders. These include DB2 XML Extender, DB2 Spatial Extender, and DB2 AIV Extender.

http://www.ibm.com/software/data/db2/extenders/support/
The DB2 Extenders support Web pages provide access to frequently asked questions, hints and tips, fixes, and documentation.

The Publications Center provides information on how to order or download publications.

The Professional Certification Program from the IBM Web site provides certification test information for a variety of IBM products, including DB2.

On Compuserve: GO IBMDB2
Enter this command to access the IBM DB2 Family forums. All DB2 products are supported through these forums.

For information on how to contact IBM outside of the United States, refer to Appendix A of the *IBM Software Support Handbook*. To access this document, go to the following Web page:
http://techsupport.services.ibm.com/guides/contacts.html

Note: In some countries, IBM-authorized dealers should contact their dealer support structure instead of the IBM Support Center.
About this book
Chapter 1. Overview and concepts

DB2 Net Search Extender Version 8.1 is one of a family of DB2 Extenders™.

It replaces DB2 Text InformationExtender Version 7.2, and Net Search Extender Version 7.2 and offers users and application programmers a fast, versatile, and intelligent method of searching full-text documents stored in DB2, other databases, and file systems using SQL queries.

Key concepts

To fully understand the capabilities of DB2 Net Search Extender, it is necessary to understand key terms, which are found bold in this section, and the various options available. It is also necessary to have a basic understanding of DB2 Universal Database concepts and terms.

Basically, DB2 Net Search Extender searches text documents held in the column of a database table.

The text documents must be uniquely identifiable. Net Search Extender uses the primary key of the table for this purpose.

The documents can be in various formats, such as HTML or XML.

Rather than sequentially searching through the text documents that would take a considerable amount of time, Net Search Extender creates a text index in order to make documents searchable.

A text index consists of significant terms extracted from the text documents.
Text index creation is the process of defining and declaring the properties of the index, such as the location of the index. After creation, the text index contains no data. Index update is the process of adding data to the text index. The first index update adds all text documents from the text column to the index. It is known as the initial update.

By using a text index for searching, there are synchronization issues between the table and the text index that must be taken into account, as any follow-up changes to the table, such as additions, deletions, and updates to the text documents must be reflected in the text index.

Synchronization in Net Search Extender is based on triggers that automatically store information about new, changed, and deleted documents in a log table. There is one log table for each text index. Applying the contents of the log table to its corresponding text index is called incremental update.
You can update the text index using a manual or automatic option. The automatic option uses an update schedule to set days and times.

Note that neither of these options synchronizes the text index within the scope of a transaction that updates, deletes, and inserts text documents. Net Search Extender’s asynchronous text indexing improves performance and concurrency. The update is applied within a separate transaction to a copy of a very small part of the index. The index is locked for read access only during a very short period of time when the copy is put in place of the original. This is invisible to search operations, see Chapter 5, “Net Search Extender instance services”, on page 27 for information.

A text index has certain properties, such as index file location and automatic update properties. If necessary, some properties can be changed. This is called altering the index.

One such property is whether the text index should be presorted according to a certain ORDER BY phrase on the table columns. In such a case, the initial update will index the text document in the order specified and search results will be returned in this order.

For example, specifying presorted book abstracts according to the book price. When looking for the least expensive books about relational database systems, you can restrict your text search to return only the first couple of books as these will be the cheapest. However, without presorted indexes, you would have to search for all books and join these with the cheapest books, which would be a more costly operation.

Net Search Extender allows several presorted indexes per text column. For example, one index for presorting books according to the date of publication and a second presorting books according to the price.

Usually the first update after creating the text index is an initial update, and the following updates are incremental. However, when working with presorted indexes you want to keep the order in case of updates. This is
addressed by the **Recreate Index on Update** option, which totally rebuilds the index each time an update is performed.

After the text index is updated, you can search using one of the following options:

- An SQL scalar search function
- A stored procedure search
- An SQL table-valued function

As the search options have different operating characteristics, they are explained in the following sections.

**Using an SQL scalar search function**

Net Search Extender offers three scalar text search functions (CONTAINS, NOOFMATCHES, and SCORE) that are seamlessly integrated within SQL. They can be used within SQL queries in the same places where standard SQL expressions would be used. Typical queries are:

```sql
SELECT * FROM books WHERE CONTAINS (abstract, '"relational databases"') = 1 AND PRICE <10
```

```sql
SELECT ISBN, SCORE (abstract, '"relational databases"') as SCORE
from BOOKS
where NUMBEROFMATCHES (abstract, '"relational databases"') >5 AND PRICE <10
order by SCORE
```

The SQL scalar functions return an indicator to how well the text documents matched a given text search condition. Then the SELECT phase of the SQL query determines the information returned to the end user.
Key concepts

Use the SQL scalar search function as the default search method which should be suitable for a majority of situations, especially when the text search expression is combined with other, different conditions.

Note that the DB2 Optimizer is aware of how many text documents can be expected to match a CONTAINS predicate and how costly different access plan alternatives will be. The optimizer will choose the cheapest access plan.

Using a stored procedure search

The stored procedure search works differently from the SQL scalar search functions. At text index creation, you have to specify which columns out of the table/view should be returned to the end user. This data is stored in a cache in main memory. This enables the stored procedure search to return search results extremely quickly. The cache needs to be activated before it can be used and there is a corresponding deactivate command.

Call TextSearch stored procedure search

The ACTIVATE command loads data into either a temporary cache (which is created from scratch on activation), or a persistent cache, which is maintained on disk.

Using the stored procedure for searching requires memory calculations, such as how much memory is required and how much free memory should be left for index updates.

The stored procedure can work on text indexes created on views. However, as triggers cannot be created on views, any changes are not automatically recognized. You can manually add the changed information to the log table, or you can work with the RECREATE option.
Key concepts

Use the stored procedure search for high performance/high scalability applications that are interested in text-search-only queries, such as queries that do not need to join text search results with the results of other complex SQL conditions.

The main functional differences to the SQL scalar search functions are:
- The stored procedure search cannot be used in arbitrary SQL queries, but is a query against a predefined cache table.
- The stored procedure search can exploit indexes on views.
- The stored procedure search can exploit multiple presorted text indexes on a column.

Note that for this option, a large amount of main memory must be available. For additional information, see Appendix B, “Using large amounts of memory”, on page 179.

Using an SQL table-valued function for searching

The SQL table-valued function is a compromise between the SQL scalar search functions and stored procedure search.

The main functional differences to the stored procedure search are:
- No cache is necessary (and no cache is exploited).
- The table-valued function can be used in arbitrary SQL statements.
- A large amount of main memory is not necessary.

The main functional differences to the SQL scalar search functions are:
- The SQL table-valued function can exploit indexes on views.
The table-valued function can exploit presorted text indexes.

Use the SQL table-valued function in those cases where you would normally use an SQL scalar function, but you want to exploit text indexes on views or presorted text indexes.

Additional concepts

As well as understanding the key concepts of DB2 Net Search Extender, there are also some additional concepts that must be explained.

For more information on developing Net Search Extender-based applications, see Chapter 4, “Planning”, on page 23.

Column transformation function

You can use your own function to convert a nonsupported format or data type into a supported format or data type. By specifying a User Defined Function (UDF), you can get the original text document as input. The output from the UDF should be a supported format, which can be processed during indexing.

You can also use this feature for indexing documents that are stored on external nonsupported data stores. In this case, the DB2 column contains document references and the function returns the document contents that have the relevant document reference.

See “Creating a text index on a nonsupported data type” on page 36.

Instance services

Net Search Extender Instance Services take care of index-specific locking services and text index update services (both automatic and manual).

See Chapter 5, “Net Search Extender instance services”, on page 27 for more information.

Externally stored data

In a majority of cases, the data on which you create a text index is stored within native DB2 tables, such as in CLOBs or VARCHARS.

However, text documents that are stored externally, such as in files or other databases, are also supported. For documents stored in files, the DB2 Data Links feature is available. For documents stored on other databases, use DB2 nickname tables to create a text index.

See “Related information” on page ix.
Additional concepts

You can also use the column transformation function for data stored in nonsupported external data stores. See "Column transformation function" on page 9.

Administration views

There are several views available in DB2 Net Search Extender. They provide information on the text indexes and their properties.


For information on other concepts and issues, see Chapter 4, “Planning”, on page 23 for more information.

Key features

DB2 Net Search Extender Version 8.1 has the following key features:

• Indexing
  – Fast indexing of very large data volumes
  – Dynamic updating of indexes
  – Storing table columns in main memory at indexing time to avoid expensive physical read operations at search time
  – A choice of command line or interface via the DB2 Control Center for indexing
  – Different text formats, for example HTML and XML
  – Nickname table support
  – DB2 Datalink Manager support
  – Support of presorted text indexes

• Search
  SQL scalar search functions, a stored procedure search, and an SQL table-valued function support the following search functions:
  – Boolean operations
  – Proximity search for words in the same sentence or paragraph
  – “Fuzzy” searches for words having a similar spelling as the search term
  – Wildcard searches, using front, middle, and end masking, for whole words and single characters
  – Free-text searches, for documents containing specific text the search argument is expressed in natural language
  – Thesaurus support
  – Restrict searching to sections within documents
  – Numeric attribute support
Key features

- High-speed searching through a large number of text documents with many concurrent users
- Search results
  - You can specify how the search results are sorted at indexing time
  - You can specify search result subsets when searching large data volumes and large result lists are expected
  - You can set a limit on search terms with a high hit count
  - Built-in SQL functionality combined with the Optimizer automatically selects the best plan according to the expected search results

DB2 Net Search Extender in the DB2 client/server environment

DB2 Net Search Extender search functionality is integrated into SQL and executed at the server. Therefore, you do not need to install Net Search Extender on the client to issue text search queries.

DB2 Net Search Extender supports administration calls to the server from the client side. Either install DB2 Net Search Extender on the client and server sides, or alternatively, use the DB2 Control Center to administrate DB2 Net Search Extender from the client side.
Chapter 2. Installation

This chapter describes how to install DB2 Net Search Extender in UNIX® and Windows® systems.

After installation, run the DB2 Net Search Extender installation verification script.

System requirements

The following versions of software are required to run DB2 Net Search Extender:

- DB2 Version 8.1
- Java Runtime Environment (JRE) Version 1.3.1

DB2 Net Search Extender is available as a 32-bit application on the following operating systems:

Solaris Operating Environment:
- Solaris 7
- Solaris 8
- Solaris 9

AIX®:
- AIX Version 4.3.3 The following file set is also required: xlC.aix43.rte 5.0.2.x.
- AIX Version 5.1.0 The following file set is also required: xlC.aix50.rte 5.0.2.x.

Note that AIX file sets can be downloaded from: http://techsupport.services.ibm.com/server/fixes

Windows:
- Windows NT® Version 4
- Windows 2000
- Windows XP

HP:
- HP-UX 11i
Installation

Linux:

- DB2 Net Search Extender is only supported on Intel machines. The validation status for new Linux kernels and distributions is frequently updated. To obtain the latest information for supported Linux software levels, refer to: http://www.ibm.com/software/data/db2/linux/validate

DB2 Net Search Extender has the same minimum software and hardware requirements as DB2 Universal Database Version 8.1. For these requirements, as well as specific operating system patches, refer to the corresponding section in the IBM DB2 Universal Database Quick Beginnings Version 8 documentation.

Any additional hardware requirements depend on the size and type of text index selected. For DB2 documentation, see "Related information" on page ix.

Installation on UNIX

To install on UNIX, follow these steps:

1. Install the product.
2. Update the DB2 instance.

Step 1 for UNIX: Install the product components

To install on UNIX, follow these steps:

1. Log on at the target machine as the root user.
2. Change to the correct directory for your platform:
   - cd /<cdrom> where <cdrom> is your CD-ROM driver path.
   - cd /<platform>
3. Call ./nsetsetup.sh and follow the instructions displayed on the screen.
   Note: Ignore any of the 'Exited with' messages.

Step 2 for UNIX: Update the DB2 instance

To update the DB2 instance, follow these steps:

1. Ensure that you are active as the root user.
2. Depending on the platform, use one of the following commands:
   - For AIX: cd /usr/opt/db2_08_01/instance
   - For Solaris, Linux, HP-UX: cd /opt/IBM/db2/V8.1/instance
3. Run db2iupdt using ./db2iupdt <db2instance>, where <db2instance> is an existing DB2 instance user ID that you would like to use with Net Search Extender.
4. Log out.
New DB2 instances are automatically created for the Net Search Extender during *db2icrt*.

**Windows installation**

To install on Windows you must be logged on with a user ID that has administrative rights, and then follow these steps:

1. Use the `<cdrom>:\windows\install\setup.exe` to transfer the files from the package to the target machine. Note that for every DB2 service, you must enter a user ID and password to create the correct DB2 Net Search Extender service.
2. Reboot the system after data transfer.
3. Call `db2text start` to start the DB2 Net Search Extender Instance Services.

For every DB2 instance, a Windows service is created. Make sure that the DB2 instance services run under a user account and not under the systems account.

**Directory and file names**

You must specify the directory and file names in SBCS characters for all Net Search Extender commands. The maximum length of the path names (including the file name) is 256 bytes or less.

**Installation verification**

Net Search Extender installation verification is available on Windows and UNIX platforms.

**Installation verification on Windows**

Complete the following steps to verify that Net Search Extender is correctly installed.

- Follow these steps to call the administration script `nseexample.bat` to set up the text indexes:
  1. Call `db2cmd` to open a DB2 command window.
  2. Change to `<sql1ib>\samples\db2ext`
Installation verification

3. From the DB2 command window, call `nsesample.bat <yourdb>` where `<yourdb>` is the name of a database. Note that the database will be created if it does not exist.

4. Check the generated output file `nsesample.log` in the current directory.
   - Then call the following sample queries to execute in the DB2 command window:
     1. Connect to your database using `db2 connect to <yourdb>`
     2. Execute the sample queries using `db2 -tvf search`
     3. Check the results of the queries contained in the script. Note that every query should return one or more hits.

   If no errors were found in the `nsesample.log` file and all the queries are working, Net Search Extender is successfully installed.

Installation verification on UNIX

Complete the following steps to verify that Net Search Extender is correctly installed.

- Follow these steps to call the administration script `nsesample` to set up the text indexes:
  1. Change to `<instance_owner_home>/sqllib/samples/db2ext`
  2. Call `nsesample <yourdb>` where `<yourdb>` is the name of a database. Note that the database will be created if it does not exist.
  3. Check the generated output file `nsesample.log` in your home directory.

- Then call some sample queries to execute in the same DB2 command window:
  1. Connect to your database using `db2 connect to <yourdb>`
  2. Execute the sample queries using `db2 -tvf search`
  3. Check the results of the queries contained in the script. Note that every query should return one or more hits.

   If no errors were found in the `nsesample.log` file and all queries are working, Net Search Extender has been successfully installed.

---

**Note**

For migration information, see Appendix A, “Migration”, on page 177.
Chapter 3. User scenarios

Use this chapter to learn about Net Search Extender by using the following walk-through examples:

**The SQL scalar search example**
This command line example demonstrates the indexing and search functions available.

**The stored procedure example**
This command line example uses the index command from the example above, but with the addition of a cache demonstrates the different indexing and search functions available.

**The SQL table-valued function example**

For more information on using Net Search Extender, see the following chapters:

- Chapter 4, “Planning”, on page 23
- Chapter 6, “Creating and maintaining a text index”, on page 31
- Chapter 8, “Searching”, on page 73

---

**Note**

Before using the examples, ensure that Net Search Extender is installed successfully by using the installation verification procedure.

---

**Simple example with the SQL scalar search function**

Use the following steps in the DB2 Net Search Extender example:

1. Creating a database
2. Enabling a database for text search
3. Creating a table
4. Creating a full-text index
5. Loading sample data
6. Synchronizing the text index
7. Searching with the text index

You can issue the sample commands on the command line of the operating system by using an existing database. For the following examples, the database name is sample.
User scenarios

Creating a database
You can create a database in DB2 using the following command:

```
db2 "create database sample"
```

Enabling a database for text search
You can issue DB2 Net Search Extender commands in the same way as DB2 commands on the command line of the operating system. For example, use the following command to start Net Search Extender Instance Services:

```
db2text "START"
```

Use the following command to prepare the database for use with DB2 Net Search Extender:

```
db2text "ENABLE DATABASE FOR TEXT CONNECT TO sample"
```

You need to do this step only once for each database.

Creating a table

```
db2 "CREATE TABLE books (isbn VARCHAR(18) not null PRIMARY KEY, author VARCHAR(30), story LONG VARCHAR, year INTEGER)"
```

This DB2 command creates a table called `books`. It contains columns for the `author`, `story`, `isbn` number and the `year` the book was published.

Creating a full-text index

```
db2text "CREATE INDEX db2ext.myTextIndex FOR TEXT ON books (story) CONNECT TO sample"
```

This command creates a full-text index for the column `story`. The name of the text index is `db2ext.myTextIndex`.

Loading sample data

```
db2 "INSERT INTO books VALUES ('0-13-086755-1','John', 'A man was running down the street.',2001)"
```

```
db2 "INSERT INTO books VALUES ('0-13-086755-2','Mike', 'The cat hunts some mice.', 2000)"
```

```
db2 "INSERT INTO books VALUES ('0-13-086755-3','Peter', 'Some men were standing beside the table.',1999)"
```

These commands load the isbn, author, story, and publishing year for these books into the table.

Synchronizing the text index
To update the text index with data from the sample table, use the following command:

```
db2text "UPDATE INDEX db2ext.myTextIndex FOR TEXT CONNECT TO sample"
```
Searching with the text index

To search the text index, use the following CONTAINS scalar search function:

```
db2 "SELECT author, story FROM books WHERE CONTAINS
  (story, '"cat"') = 1 AND YEAR >= 2000"
```

Note

Depending on the operating system shell you are using, you might need a different escape character in front of the double quotes surrounding the text search phrase. In the above example, the `\` is used.

This query searches for all books about the term cat that are greater or equal to the year 2000. The following result table is returned:

<table>
<thead>
<tr>
<th>AUTHOR</th>
<th>STORY</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mike</td>
<td>The cat hunts some mice.</td>
</tr>
</tbody>
</table>

Other functions supported include SCORE and NUMBEROFMATCHES. SCORE returns an indicator on how well a found document is described by the search argument. NUMBEROFMATCHES returns how many matches of the query terms were found in a resulting document.

Simple example with cache usage and stored procedure search

Use the following steps in the DB2 Net Search Extender stored procedure search example:

1. Creating a text index with cache option.
2. Synchronizing the index and activating the cache.
3. Searching with the TEXTSEARCH Stored Procedure.

Note

The stored procedure example assumes that the steps from the previous example are complete and that the database is still enabled.

Creating a text index with cache option

As the database is already enabled, use the following command to create a full-text index:

```
db2text "CREATE INDEX db2ext.mySTPTextIndex FOR TEXT ON books (story)
  CACHE TABLE (author, story) MAXIMUM CACHE SIZE 1
  CONNECT TO sample"
```
User scenarios

In this example, the full-text index is for the column story and it specifies a cache table on the columns author and story. The text index is called mySTPTextIndex.

Synchronizing the index and activating the cache
To update the index according to the data inserted into the table, use the following command:

```
db2text "UPDATE INDEX db2ext.mySTPTextIndex FOR TEXT CONNECT TO sample"
```

To activate the cache, use the following command:

```
db2text "ACTIVATE CACHE FOR INDEX db2ext.mySTPTextIndex FOR TEXT CONNECT TO sample"
```

Searching with the TEXTSEARCH Stored Procedure
You can use the DB2 Net Search Extender TEXTSEARCH stored procedure only in certain cases. For details, see “Using a stored procedure search” on page 7.

```
db2 "call db2ext.textSearch ('"cat"','DB2EXT','MYSTPTEXTINDEX',0,2,0,0,?,?)"
```

This query searches for all books about a cat but only returns the first two results. In this case, the following result table is returned:

<table>
<thead>
<tr>
<th>Value of output parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter Name</td>
</tr>
<tr>
<td>-----------------</td>
</tr>
<tr>
<td>SEARCHTERMCOUNTS</td>
</tr>
<tr>
<td>TOTALNUMBEROFRESULTS</td>
</tr>
</tbody>
</table>

```
AUTHOR  | STORY
Mike    | The cat hunts some mice.
```

Return Status = 0

For more samples about the search syntax check the following file:
sqllib\sample\db2ext\search

Simple example with the SQL table-valued function
You can use the SQL table-valued function on the text indexes created in the previous examples.

The SQL table-valued function query corresponds to the previously used CONTAINS query. See “Synchronizing the text index” on page 18 for information.

```
db2 "SELECT author, story FROM books b, table (db2ext.textsearch ('"cat"','DB2EXT','MYTEXTINDEX',0,2,CAST (NULL AS VARCHAR(18)))) T where T.primKey = b.isbn"
```
For details on using the table-valued function, see "Using an SQL table-valued function for searching" on page 8 for more information.

**Note**

The cast (NULL AS VARCHAR(18)) is to call the corresponding table-valued function to the primary key for table books.
User scenarios
Chapter 4. Planning

To use DB2 Net Search Extender in the most effective way, it is essential that some planning occurs prior to development. This development should involve several groups including those in database administration, interface and system designers, system architects, and developers.

The following sections provide a guide to the areas that should be considered.

For more information on developing DB2 Net Search Extender based applications, see the following chapters:

- Chapter 5, “Net Search Extender instance services”, on page 27
- Chapter 6, “Creating and maintaining a text index”, on page 31
- Chapter 8, “Searching”, on page 73

Directory locations and index storage

The disk space you need for an index depends on the size and type of data to be indexed. As a guideline for an index, reserve disk space for about 0.7 times the size of the documents being indexed for single-byte documents. For double-byte documents, reserve the same size as the documents being indexed.

The amount of space needed for the temporary files in the work directory is 1.0 to 4.0 times the amount of space needed for the final index file in the index directory.

If you have several large indexes, you should store them on separate disk devices, especially if you have concurrent access to the indexes during index update or search.

You must also specify the directory where the text index is to be stored. Ensure that there is enough disk space and that the DB2 instance owner has write access to the directory.

Document formats and supported code pages

DB2 Net Search Extender needs to know the format (or type) of text documents that you intend to search. This information is needed when indexing text documents.

The following document formats are supported:
Planning

<table>
<thead>
<tr>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TEXT</td>
<td>Plain text (for example, flat ASCII)</td>
</tr>
<tr>
<td>HTML</td>
<td>Hypertext Markup Language</td>
</tr>
<tr>
<td>XML</td>
<td>Extended Markup Language</td>
</tr>
<tr>
<td>GPP</td>
<td>General Purpose Format (flat text with user-defined tags)</td>
</tr>
</tbody>
</table>

For the document formats HTML, XML, and GPP, searching can be restricted to specific parts of a document. Chapter 9, “Working with structured documents”, on page 83 explains how to define and work with document models.

For nonsupported document formats, you can write a User Defined Function (UDF), which must be specified at index creation time and converts the data from the nonsupported format to a supported format.

See “CREATE INDEX” on page 114 for more information.

Documents can be indexed if they are in one of the supported Coded Character Set Identifiers (CCSIDs). These are also known as code pages. See Appendix D, “Supported CCSIDs”, on page 189 for a list of these code pages.

To check the database code page, use the following DB2 command:

db2 GET DB CFG for <dbname>

For consistency, DB2 normally converts the code page of a document to the code page of the database. However, when you store data in a DB2 database in a column having a binary data type, such as BLOB, FOR BIT DATA, or a datalink value, DB2 does not convert the data, and the documents retain their original CCSIDs.

Note that having two different code pages might cause problems when creating a text index or searching. See “Creating a text index on binary data types” on page 36 for further information.

User roles

These are the different roles and authorizations for users of Net Search Extender:

**DB2 instance owner**

The DB2 instance owner user can start and stop the instance services for DB2 Net Search Extender and control the locking services. In addition, the DB2 instance user becomes DBADM for each enabled database. This enables a central point of control for all database changes driven by DB2 Net Search Extender.
User roles

Required DB2 authorizations
DBADM is granted on ENABLE DATABASE.

Required file system authorizations
Read/write for all text index directories and read access to model files.

Commands for the instance owner
DB2TEXT START, DB2TEXT STOP, and DB2TEXT CONTROL.

The commands are only allowed on the server, which in a distributed DB2 environment can be any of the servers, and each command checks if the user running the command is the DB2 instance owner. Note that using a separate fenced user ID on UNIX systems does not influence Net Search Extender processing in terms of authorization or performance.

Database administrators

Database administrators can enable and disable databases for use with DB2 Net Search Extender.

Required DB2 authorizations
DBADM (SYSADM for ENABLE DATABASE).

Commands for the database administrator
DB2TEXT ENABLE DATABASE and DB2TEXT DISABLE DATABASE.

Text table owners

The text table owner can create, drop, and change indexes. Note that they must be able to control the location of indexes and updates to the full-text indexes.

Required DB2 authorizations/privileges
Owner of text table.

Commands for the text table owner:
DB2TEXT CREATE INDEX, DB2TEXT DROP INDEX, DB2TEXT ALTER INDEX, DB2TEXT ACTIVATE CACHE, DB2TEXT DEACTIVATE CACHE, DB2TEXT UPDATE INDEX, DB2TEXT CLEAR EVENTS, and DB2EXTTH.

Note that the command implementation is partially run under the user ID of the DB2 instance owner. Therefore, the necessary file system access has to be granted to the instance owner before creating or altering the text indexes.

Using the command line or DB2 Control Center interface for indexing

For indexing, you can either use the command line option, or the DB2 Control Center interface.
Chapter 5. Net Search Extender instance services

DB2 Net Search Extender Instance Services consist of the following services:

- Locking services
- Update services

This chapter explains how to start and stop the DB2 Net Search Extender Instance Services. It also discusses Locking Services and Update Services in detail.

Starting and stopping DB2 Net Search Extender

Before you can create a text index and search your documents, you have to start the DB2 Net Search Extender Instance Services.

To start the Instance Services, log on to the DB2 instance owner user ID (UNIX systems only) and enter the following command:

db2text start

To stop the Instance Services, enter the following command:

db2text stop

Note that there must be one Net Search Extender Instance Service per DB2 instance. The locking service maintains the locks for multiple databases.

Locking services

When you start DB2 Net Search Extender, locking services are automatically started. The locking services are needed to synchronize concurrent access to text indexes in Net Search Extender.

The locking services ensure that no two processes attempt to change a text index simultaneously, or that no process reads text index data while another process is making changes to the same text index data. Therefore, most processes request a lock on a text index before starting and release it again when processing has completed.

Note that the locking services for Net Search Extender text indexes must not be confused with DB2 locks that control access to DB2 tables.

Using the locking services

In Net Search Extender, there are different types of locks that control concurrent access to an index depending on whether the text index is only
Using the locking services

being read, as in the case of a search request, or if changes to the text index need to be computed and subsequently written to files.

During db2text start, the locking services are automatically started. There are the following types of locks on a text index:

**S-lock** For shared read-only access. For example, search requests.

**U-lock** For read/write access while computing changes to an index (update) with concurrent read access. For example, search requests.

**X-lock** For exclusive read/write access for a short period during which changes are actually written to the index.

**IX-lock** For intended exclusive read/write access preventing any new S-locks while the update process is waiting for an X-lock.

There is one Net Search Extender locking service per DB2 instance. The locking service maintains the locks for multiple databases.

The locking services configuration file is called db2extlm.cfg. It is stored on <DB2_INSTOWNERHOMEDIR>/sqllib/db2ext for UNIX systems and on <sqlib><DB2INSTANCE>\db2ext for Windows.

Changes to the configuration file only take effect when Net Search Extender Instance Services are started during db2text start. See "CONTROL" on page 94 for further information. The following values can be set by the user:

- The maximum number of databases
- The maximum number of indexes per database
- The maximum number of allowed locks (concurrent users) per index
- Waiting times and the number of attempts to obtain a lock

The default values of the configuration file are as follows:

```xml
<default
    maxDbs     = " 8"
    maxIdxPerDb = " 50"
    maxLocksPerIdx = "100"
    sWait = " 50"
    uWait = " 500"
    xWait = " 500"
    sAttempt = "50"
    uAttempt = "10"
    xAttempt = "60"
```
Using the locking services

latchTimeout = "80"

The syntax is <default attribute=value.../> and the attributes have the following meanings:

maxDbs
The number of databases the locking services can handle (integer >1).

maxIdxPerDb
The number of indexes per database that can be locked (integer >1). This value is the same for all databases.

maxLocksPerIdx
The number of locks that can simultaneously exist on an index (integer >1). This value is the same for all indexes.

sWait/sAttempt
When requesting an S-lock, sAttempt is the number of attempts that are made if the lock is not granted immediately. sWait is the waiting time between these attempts (integer >1). These parameters also apply to IX-locks.

uWait/uAttempt
When requesting a U-lock, uAttempt is the number of attempts that are made if the lock is not granted immediately. uWait is the waiting time between these attempts (integer >1).

xWait/xAttempt
When requesting an X-lock, xAttempt is the number of attempts that are made if the lock is not granted immediately. xWait is the waiting time between these attempts (integer >1).

latchTimeout
This is additional waiting time for interval locking services. To determine the total waiting time for a lock, use the following calculation:

waiting time = # attempts * (# waits + (2 * # latchTimeout))

The waiting time is calculated in milliseconds. Note that with each attempt, the latchTimeout value is doubled when added to the overall waiting time.

Viewing a lock snapshot
You can view a lock snapshot by using one of the following commands:

• For a single text index:
  db2text CONTROL LIST ALL LOCKS FOR DATABASE mydatabase INDEX myindex
For all locked text indexes of a database (only indexes that are actually locked are listed):

    db2text CONTROL LIST ALL LOCKS FOR DATABASE mydatabase

The first time a text index is locked, memory is reserved for the database and the text index in the locking services. If further text indexes are locked, memory is also allocated for these indexes in the locking services. This memory is only freed again when the text index is dropped or the database disabled, or whenever the Net Search Extender services are restarted. This means that a text index or database consumes memory in the locking services, even if there are no locks currently set.

The command "db2text CONTROL CLEAR ALL LOCKS" forces the release of all the locks on a database or index. See "CONTROL" on page 94 for details on how to use this command. Note that this command does not free any memory allocated for the database or indexes. To free memory, you must either drop the index or disable the database, or restart the Net Search Extender services.

**Viewing a lock snapshot**

- For all locked text indexes of a database (only indexes that are actually locked are listed):

  db2text CONTROL LIST ALL LOCKS FOR DATABASE mydatabase

Update services

Update services are started during db2text start. These services are required to update the text index automatically at the specified times. Note that the text index is not immediately synchronized with the user table.

During index creation, you can specify how often Update Services check if an update of the index is required by using the following command:

    db2text create index DB2EXT.TITLE for text on DB2EXT.TEXTTAB (TITLE)
    UPDATE FREQUENCY D(1,3) H(0,12) M(0) update minimum 5

In this example, this means that at 12 p.m. and 12 a.m. every Monday and Wednesday the Update Services wake up and check if there is some work to be completed on index db2ext.title. Note that there need to be at least five changes before the automatic index update can start to synchronize with the database.

See "CREATE INDEX" on page 114 for more details on the parameters.
Chapter 6. Creating and maintaining a text index

This chapter provides information on creating and maintaining a text index and covers the following areas:

- Introducing the db2text commands
- Enabling a database for text search
- Creating a text index for different data types
- Creating a text index which a stored procedure can use
- Maintaining an index

There is also information on avoiding code page problems which might occur, and performance considerations which you might need to take into account.

Before creating a text index, ensure that you have considered the prerequisites found in Chapter 4, “Planning”, on page 23. Other indexing prerequisites include starting DB2 Net Search Extender Instance Services, using the db2text start command.

For examples of creating a text index and making text searchable, see Chapter 3, “User scenarios”, on page 17.

Note
You can also create and maintain a text index by using the DB2 Control Center. See Chapter 7, “Using DB2 Control Center”, on page 51.

Introducing the db2text commands

Here is an example of a DB2 Net Search Extender command:

```
db2text ENABLE DATABASE FOR TEXT
```

Note
The db2text commands, such as db2text ENABLE DATABASE FOR TEXT and db2text CREATE INDEX, are also called commands.
Tip
For every create and index maintenance command, you can specify the database, user, and password.

db2text ... connect TO <database> USER <userID> USING <password>

Note that if you leave out the connect options in the db2text command, the environment variable DB2DBDFT specifies the database.

To display a list of commands, enter the following command:

db2text ?

To display the syntax of an individual command, enter the following command:

db2text ? command

For example, to display the syntax of the CREATE INDEX command, use the following command:

db2text ? CREATE INDEX

Note
The system shell interprets special characters such as ?, (, ), *, !, and ".
Therefore, if the command contains these characters, use quotation marks or an escape character.

Here is an example of a UNIX command that uses special characters:

db2 "SELECT * FROM sample WHERE CONTAINS (DESCRIPTION, '"enable"') = 1"

Enabling a database

Summary

<table>
<thead>
<tr>
<th>When</th>
<th>Once for each database that contains columns of text to be searched in.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command</td>
<td>ENABLE DATABASE FOR TEXT</td>
</tr>
<tr>
<td>Authorization</td>
<td>SYSADM</td>
</tr>
</tbody>
</table>
Enabling a database

This command prepares the connected database for use by DB2 Net Search Extender.

This command also declares DB2 Net Search Extender search functions and procedures that are described in Chapter 15, "SQL scalar search function and the SQL table-valued function", on page 147.

When you enable a database, you also create the following tables and views:

- **db2ext.dbdefaults**
  Shows the database default values for index, text, and processing characteristics.

- **db2ext.textindexformats**
  Shows the list of supported formats and the model files used.

- **db2ext.indexconfiguration**
  Shows the index configuration parameters.

- **db2ext.textindexes**
  A catalog view that keeps track of all text indexes.

- **db2ext.proxyinformation**
  Shows proxy information for accessing files using a proxy server.

For information on all the views, see Appendix C, “Net Search Extender information catalogs”, on page 181.

When a database is enabled, it remains enabled until you disable it.

Disabling a database

<table>
<thead>
<tr>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>When</strong></td>
</tr>
<tr>
<td><strong>Command</strong></td>
</tr>
<tr>
<td><strong>Authorization</strong></td>
</tr>
</tbody>
</table>

When DB2 Net Search Extender prepares the database for use, certain administrative changes are made. This section describes functions that help you to reverse this process.

To disable the connected database, use the following command:
Disabling a database

db2text DISABLE DATABASE FOR TEXT

When you disable a database, you also delete the following objects:
• The DB2 Net Search Extender catalog views that were created when the database was enabled.
• The declaration of DB2 Net Search Extender’s SQL functions (UDFs).

To disable the database and remove all the text indexes, use the following command:
db2text DISABLE DATABASE for text force

Note
If you disable a database without using force, the existing indexes are not removed. Therefore, drop the indexes before disabling the database, or call disable with the force option.

Creating a text index

<table>
<thead>
<tr>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>When</strong></td>
</tr>
<tr>
<td><strong>Command</strong></td>
</tr>
<tr>
<td><strong>Authorization</strong></td>
</tr>
</tbody>
</table>

You can create a text index on supported data types, although there are different requirements for the following data types:
• Binary data types
• Nonsupported data types
• Datalink data types

There are also different requirements for creating a text index for a stored procedure search.

When you create a text index, you also create the following objects:

A log table
This keeps track of all changed rows in the user table. Note that if you select the **Recreate index on Update** option, the log table is not created.
Creating a text index

An event table
This collects information about problems during an update of the text indexes.

Triggers on the user table
These add information to the log table whenever a document in the column is added, deleted, or changed. The information is necessary for index synchronization when time indexing next occurs.

Note that you only create triggers if you create a log table, and the text index is created on a base table and not on views or nickname tables.

To optimize performance and disk space, use the CREATE INDEX command to specify a different tablespace for the tables.

---

Note
Using the DB2 LOAD command to import your documents can cause problems, as triggers do not fire and incremental indexing of the loaded documents is not possible.

Therefore, it is preferable to use the DB2 IMPORT command as this activates the triggers.

---

The following example creates a text index on text column HTMLFILE in table htmltab.

db2text create index DB2EXT.HTMLIDX for text on DB2EXT.HTMLTAB (HTMLFILE) format HTML

A primary key on this table is necessary.

The default values for index creation are from the db2ext.dbdefaults view.

If errors occur during indexing, so-called index update events are added to the event table. For example, when a document queued for indexing could not be found. For additional information, see the "Event view" on page 187.

To reverse the changes made by CREATE INDEX, use the DROP INDEX command. See "Dropping a text index" on page 48 for this information.

To synchronize the text index with the database, use the following command:
db2text update index DB2EXT.HTMLIDX for text

Note that you can only find documents after synchronization.
Creating a text index

Search summary

Depending on the options selected during index creation, different ways of searching are possible:

- The SQL scalar search functions work on all text indexes, except those created on views.
- The stored procedure search function only works on text indexes that are created with a cache.
- The SQL table-valued function works on all text indexes, including those created on views.

Creating a text index on binary data types

When you store data in a column having a binary data type, for example BLOB, FOR BIT DATA, or a datalink value, DB2 does not convert the data. This means that the documents retain their original code pages (CCSIDs), which can cause problems when creating a text index as you might have two different code pages. Therefore, you need to determine whether you are using the code page of the database, or the code page specified in the CREATE INDEX command.

To avoid this problem, specify the code page when creating the text index:

```sql
db2text CREATE INDEX db2ext.comment FOR TEXT ON db2ext.texttab (comment) CCSID 1252
```

If the code page is not specified, check which CCSID has been used to create the index, by calling:

```sql
db2 SELECT ccsid FROM db2ext.textindexes WHERE INDSCHEMA = 'COMMENT' AND INDNAME = 'DB2EXT'
```

Note that having documents in different code pages is not supported within one text index. For information on how DB2 converts document code page settings, go to the DB2 Universal Database Administration Guide. See "Related information" on page ix.

Note that there is no such problem with creating indexes on character data types.

Creating a text index on a non supported data type

To create an index, the text columns must be one of the following data types:

- CHAR
- VARCHAR
- LONG VARCHAR
- CLOB
Creating a text index

- GRAPHIC
- VARGRAPHIC
- LONG VARGRAPHIC
- DBCLOB
- BLOB
- DATALINK

If the documents are in a column of a different type, such as a user-defined type (UDT), you must provide a function that takes the user type as input and provides as an output type one of the above-mentioned types.

Specify the name of this transformation function. See "CREATE INDEX" on page 114 for further information.

Example: You intend to store compressed text in a table.
1. Create a user-defined type (UDT) for the text in an interactive SQL session:
   ```sql
   db2 "CREATE DISTINCT TYPE COMPRESSED_TEXT AS CLOB(1M)"
   ```
2. Create a table and insert the text into it:
   ```sql
   db2 "CREATE TABLE UDTTABLE (author VARCHAR(50) not null, text COMPRESSED_TEXT, primary key (author))"
   db2 "INSERT ...
   ```
3. Create a user-defined function (UDF) called, for example, uncompress. This receives a value of type COMPRESSED_TEXT and returns the corresponding uncompressed text as, for example, a CLOB(10M) value.
4. Create your text index in the following way to specify the uncompress UDF:
   ```sql
   db2text "CREATE INDEX UDTINDEX for text ON UDTTABLE
   (uncompress(text))
   ..."
   ```

Creating a text index for DATALINK data types

DB2 Net Search Extender supports the data type DATALINK.
1. If you are using proxies, add one row to the `db2ext.proxyinformation` table, counting the host name, a timeout value in seconds, and either port 'proxy' or 'socks'.
   ```sql
   db2 INSERT into db2ext.proxyinformation values ('hostname', '80' 'proxy', 10)
   ```
   Note that only one row is allowed in this table. A trigger assures this.
2. See the DB2 Universal Database Version 8 documentation for details on how to set up the Java environment on different platforms, for example in the DB2 Information Center. Basically, you must adjust the database manager configuration.
Creating a text index

The Data Link UDF returns blob (100 KB). To change the return size, use DB2EXTDL and update the database manager configuration parameter, java_heap_sz.

Installing the Data Links jar file

With Net Search Extender, you can index data stored in files that are referenced using the DB2 Data Links feature. For this, you need to install the Data Links jar file ctedludf.jar. In the db2 command line processor, run the following command:

- For UNIX:
  call sqlj.install_jar
  ('file:/<homedir>/sqllib/java/ctedludf.jar','ctedludf_jar')

- For Windows:
  call sqlj.install_jar
  ('file:///<partition>/sqllib/java/ctedludf.jar','ctedludf_jar')

You also need to update your Java heap size using the following command:

db2 update dbm cfg using JAVA_HEAP_SZ 1024

Use the following command to unregister the jar file on all platforms:

call sqlj.remove_jar('ctedludf_jar')

For a list of error messages, see Appendix K, “Data Link messages”, on page 235.

Creating a text index which the stored procedure search can use

To use the stored procedure search, you need to specify cache options during the CREATE INDEX command. This enables high performance, by moving all the specified data into main memory.

However, before the first index update for searching, ensure that your table contains documents to avoid updating an index on a non-populated table. This provides a better indexing performance and a solid estimation of cache memory requirements.

The stored procedure search allows you to quickly return predefined data that is associated with a document. Use the cache table option to define this in the CREATE INDEX command. The ACTIVATE CACHE command then moves the specified data into a memory cache.

Note

The SQL scalar search functions can also use this text index, if the text index is not created on a view.
Creating a cached text index which the stored procedure search can use

When creating a text index for stored procedure search, you must determine and calculate the following parameters:

- The type of cache.
- How to update the index.
- The maximum amount of memory that Net Search Extender can use, the `MAXIMUM CACHE SIZE`.
- The amount of free memory necessary for subsequent document updates, the `PCTFREE`. Note that this is only for incremental updates.

The following types of cache are available:

**A temporary cache**
This is rebuilt with each `DB2TEXT ACTIVATE CACHE` command, and requires loading the data from your DB2 table to memory. This takes longer than activation of a persistent cache, especially for large indexes. However, it might provide slightly better search performance.

**A persistent cache**
This is maintained on disk and can be quickly mapped to memory by means of the operating system on each `DB2TEXT ACTIVATE CACHE` command. In incremental index update scenarios, it must remain activated to allow synchronization between the index and the cache. If this does not occur, the next `DB2TEXT ACTIVATE CACHE` command recreates the cache from scratch.

The following methods of updating a text index are available:

**Without the Recreate index on update option**
Avoid deleting and re-inserting a document in the table as the slot for a deleted document cannot be reused in the cache. As a consequence, the changing of key columns should be avoided on an activated index.

This is also known as incremental update.

**With the Recreate index on update option**
This recreates the index on each update. Use variable data types in the cache column expressions wherever possible. This will save cache space. Use the corresponding cast expressions in the `CACHE TABLE` clause.

Use this option if you expect to insert more than 50% of your documents after the initial index activation.

Net Search Extender provides two SQL functions to help you determine the `CREATE INDEX` memory parameters. These are: `MAXIMUM CACHE SIZE` and `PCTFREE`. 
Creating a cached text index which the stored procedure search can use

For incremental and recreate updates

The following command returns the recommended MAXIMUM CACHE SIZE value in megabytes (MB):

```sql
DB2EXT.MAXIMUM_CACHE_SIZE(maximumNumberDocs INTEGER,
   averageRowLength INTEGER, numberOfCacheColumns INTEGER)
```

The following command returns the average row length parameter from your table:

```sql
SELECT AVG(LENGTH(cache column_1) + ... + LENGTH(cache column_n))
```

Note that the average may change significantly when further documents are inserted into your table. The number of cache columns relates to the number of column expressions used in the CACHE TABLE clause of the DB2TEXT CREATE INDEX command.

For additional information, see Appendix B, “Using large amounts of memory”, on page 179

For incremental updates only

The following command returns the recommended PCTFREE value based on the actual and maximum numbers of documents.

```sql
DB2EXT.PCTFREE(actualNumberDocs INTEGER, maximumNumberDocs INTEGER)
```

The actual numbers of documents are the number of rows in your table at the time of the first ACTIVATE CACHE command, which creates the memory cache.

The maximum number of documents is an estimate of the maximum number of documents in your table before the next DB2TEXT ACTIVATE command (for a temporary cache), or DB2TEXT ACTIVATE CACHE RECREATE command (for a persistent cache) is run.

If you are recreating the index on each update, set the PCTFREE value to 0.
Creating a cached text index which the stored procedure search can use

Examples
Assume that you have 10 000 rows in your table and you do not expect more than 20 000. Use the following call to calculate the PCTFREE value you require:
```
db2 "values DB2EXT.PCTFREE(10000,20000) " 
```
Assume that your maximum row size is 20 000 and that you have 2 rows in your cache with an average size of 76. Use the following call to return the size:
```
db2 " values DB2EXT.MAXIMUM_CACHE_SIZE(20000,76,2) " 
```

After determining suitable parameters, you can create your index and cache table by using the following call:
```
db2text CREATE INDEX db2ext.comment FOR TEXT ON db2ext.texttab (comment) 
   CACHE TABLE (docid) PCTFREE 10 MAXIMUM CACHE SIZE 5
```

In this example, the docid column is built in addition to the index, using main memory for fast result table return. Ten percent of the cache memory is reserved for future documents and the cache is limited to a maximum of 5 MB.

Updating the text index
To search on this index, you need to update and then activate the index. This copies the specified table cache expression from the database into memory.

If during ACTIVATE or UPDATE operations, the MAXIMUM CACHE SIZE or MAXIMUM NUMBER OF DOCUMENTS are exceeded, the following actions are recommended:

**MAXIMUM CACHE SIZE for incremental updates**
Rebuild the cache by using the following sequence of DB2EXT commands: DEACTIVATE CACHE, ALTER INDEX MAXIMUM CACHE SIZE, and ACTIVATE CACHE RECREATE.

**MAXIMUM NUMBER OF DOCUMENTS**
Use the DEACTIVATE CACHE and the ACTIVATE CACHE RECREATE command for incremental updates, or recreate the index.

If you expect frequent updates on documents, consider using fixed-size data types for the cache column expressions in the CACHE TABLE clause. The following example shows how you can use the same cache storage during update operations:
```
CACHE TABLE(cast(C1 as char(20), cast(substr(C2,1,10) as char(10))....
```
Creating a cached text index which the stored procedure search can use

Activating and deactivating the cache for a text index
Before cache activation, perform any pending incremental updates to avoid a poor PCTFREE calculation.

To activate the text index, use the following command:
```
db2text ACTIVATE CACHE FOR INDEX db2ext.comment FOR TEXT
```
This command retrieves the specified cache table data out of the database and stores this in memory. The time taken depends on the size of the table.

---

**Note**
If you call update index when an index is activated, this will also update the cache tables. As deleted documents take slots in cache memory, ensure that you set PCTFREE with a high enough value.

The ACTIVATE CACHE call needs to be redone every time you stop your system. If you use the persistent cache, the new activate will be quicker.

---

**Note**
If an update occurs when the persistent cache is not activated, the persistent cache is dropped and recreated during the activate call.

To save resources, you should also deactivate any indexes that are not currently required.

To check how much memory is left, use the following call:
```
db2text control show cache status for database cte index db2ext.comment
```
This displays whether the index has been activated and how much of the specified cache space is left.

Text indexes on views

When using the stored procedure, you are able to create text indexes on views. However, one major drawback is that you cannot create triggers on views, so any changes in the underlying base tables are not recognized.

So with incremental index updates, the user has to know which document has been added, updated, or deleted in order to synchronize the text index with the database. To do this, you must add all the changes to the log table. This process is shown in the following sample:

1. To create the base table, use the following command:
Text indexes on views

db2 "create table sample (key INTEGER not null PRIMARY KEY, name VARCHAR(50) not null, comment VARCHAR(90))"

2. To add some entries, use the following commands:
   db2 "insert into sample values(1,'Claus','works in room 301')"
   db2 "insert into sample values(2,'Manja','is in the same office as Juergen')"
   db2 "insert into sample values(2,'Juergen','has the longest way to Raiko')"
   db2 "insert into sample values(3,'Raiko','is sitting in the office besides Claus')"

3. To create the view, use the following command:
   db2 "create view sampleview as select key, comment from sample"

4. Use the following commands to create, update, and activate the text index:
   db2text "create index indexview for text on hde.sampleview(comment)
cache table (comment) maximum cache size 1 key columns
for index on view (key)"
   db2text "update index indexview for text"
   db2text "activate cache for index indexview for text"

   Note that you need to specify the cache table to be able to create a text index on a view. To create the correct log table, you must specify the key columns for the index on view.

5. To update the table, use the following commands:
   db2 "insert into sample values(4,'Bernhard','is working in the same floor as Manja, but not as Claus')"
   db2 "insert into sample values(5,'Guenter','shares the office with Raiko')"

6. Then update the log table. To get the name of the log table, use the following command:
   db2 "select INDSCHEMA,INDNAME,LOGVIEWSCHEMA,LOGVIEWNAME from db2ext.textindexes"

   This is the layout of the log table:

<table>
<thead>
<tr>
<th>sqltype</th>
<th>sqlllen</th>
<th>sqlname.data</th>
<th>sqlname.length</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTEGER</td>
<td>4</td>
<td>OPERATION</td>
<td>9</td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td>26</td>
<td>TIME</td>
<td>4</td>
</tr>
<tr>
<td>INTEGER</td>
<td>4</td>
<td>PK01</td>
<td>4</td>
</tr>
</tbody>
</table>

   To add the entries into the log table, use the following commands:
   db2 "insert into sample values(0,CURRENT TIMESTAMP,4)"
   db2 "insert into sample values(0,CURRENT TIMESTAMP,5)"

   The first value describes the operation (0 = insert, 1 = update, 2 = delete).
The second should always be the CURRENT TIMESTAMP and the last value, the key which has been inserted.

7. Use the following command to update the index again:
Text indexes on views

```
db2text "update index indexview for text"
```

You can now search with the stored procedure on the new values.

### Performance considerations

To enhance performance during indexing, consider the following issues:

- Using a `VARCHAR` data type to store the text documents instead of `LONG VARCHAR` or `CLOB`.
- Using different hard disks to store the text index and the database files.
- Using small primary keys.
- Ensuring that your system has enough real memory available for all this data. If there is insufficient memory, the operating system uses paging space instead. This decreases the search performance.

For information on configuring the memory requirements of different platforms, see Appendix B, “Using large amounts of memory”, on page 179.

- The update `commitcount` parameter, used during the automatic or manual updating of the index, slows down the indexing performance during incremental indexing. Note that the parameter is not used during the initial update process.

### Note


### Maintaining text indexes

This chapter describes how to maintain text indexes and get useful information about them. The maintenance tasks are:

1. Updating and reorganizing a text index
2. Altering a text index
3. Deleting index update events
4. Dropping a text index
5. Showing index status

You can run these tasks at any time and in any sequence. The chapter also includes information on how to back up and restore indexes and enabled databases.

For commands that display information on text indexes, directory names, and updates, see Appendix C, “Net Search Extender information catalogs”, on page 181.
Maintaining text indexes

You can also maintain a text index by using the DB2 Control Center. See “Maintaining a text index” on page 67.

Updating and reorganizing a text index

After creating and updating the text index for the first time, you must keep the text index up to date. For example, when you add a text document to a database, or change an existing document in a database, you must index the document to keep the content of the index synchronized with the content of the database. Likewise, when you delete a text document from a database, its terms must be removed from the index.

If the text index was created without the RECREATE INDEX ON UPDATE option, triggers automatically store information about new, changed, or deleted documents in an internal log table. So, the next time an index update takes place, the documents referenced in the log table are indexed. For a text index on views, see “Text indexes on views” on page 42 for additional information.

If you specify the RECREATE option in the CREATE INDEX command, the index is totally rebuilt for each update. This option creates no log table or triggers.

Typically you update an index at intervals. You can change the update frequency for an existing index by using the ALTER INDEX command.

You specify the index update frequency in terms of when the update is to be made, and the minimum number of text changes that must be queued. If there are not enough changes in the log table at the day and time given, the index is not updated.

You should plan periodic indexing carefully; to index text documents is a time- and resource-consuming task. The time taken is dependent on many factors. These include the size of the documents, how many text documents have been added or changed since the previous index update, and how powerful the processor is.

Note

On a DB2 table, rollback and deadlock situations might occur in the following cases:

- High update frequencies
- High frequency change transactions
- Long transactions

The UPDATE INDEX command lets you update an index immediately on request.
Updating and reorganizing a text index

Summary

When When an index must be updated immediately without waiting for periodic indexing to occur.

Command UPDATE INDEX

Authorization CONTROL on the table

The following command updates the index:

```
db2text UPDATE INDEX comment FOR TEXT
```

This command is useful when you have added several text documents to a database and want to search them immediately.

To determine if manual reorganization is necessary, query the `db2ext.textindexes` view by using the following command:

```
db2 "select reorg_suggested from db2ext.textindexes where INDNAME = 'comment'"
```

If you specify MANUAL REORGANIZATION and often update a column, the update process becomes slower. To manually reorganize, use the following command:

```
dbtex UPDATE INDEX comment FOR TEXT reorganize
```

However, if you specify AUTOMATIC REORGANIZE during CREATE INDEX, the index will be automatically reorganized when necessary.

Altering a text index

Summary

When When the update frequency or index and work directories have to be changed.

Command ALTER INDEX

Authorization CONTROL on the table

Use this command to change the index work directory, the update frequency of an index, or the cache characteristics, principally the MAXIMUM CACHE SIZE or PCTFREE. If you do not specify an update frequency, the current settings are
left unchanged. If an index update or search is running, an error message displays. This states that the index is currently locked and no changes can be made.

The following example changes the update frequency for the index.

```
 db2text ALTER INDEX comment FOR TEXT
    UPDATE FREQUENCY d(1,2,3,4,5) h(12,15) m(00) UPDATE MINIMUM 100
```

In this example, the index is to be updated at 12:00 or 15:00, on Monday to Friday, if a minimum of 100 text documents are in the queue.

Use the following command to stop the periodic updating of an index:

```
 db2text ALTER INDEX comment FOR TEXT
    UPDATE FREQUENCY NONE
```

### Clearing index events

<table>
<thead>
<tr>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>When</strong></td>
</tr>
<tr>
<td><strong>Command</strong></td>
</tr>
<tr>
<td><strong>Authorization</strong></td>
</tr>
</tbody>
</table>

Information about indexing events, such as the update start and end times, the number of indexed documents, or document errors that occurred during the update, are stored in the index’s event table. This can help you determine the cause of the problem. When you no longer need these messages, you can delete them.

The following example deletes messages from the specified text index:

```
 db2text CLEAR EVENTS FOR INDEX comment FOR TEXT
```
Dropping a text index

Dropping a text index

Summary

When  When you no longer intend to make text searches in a text column.

Command  DROP INDEX FOR TEXT

Authorization  CONTROL on the table

Example:

db2text DROP INDEX comment FOR TEXT

When dropping a text index, you also drop the following tables and views:

- The log table and view
- The event table and view
- The log table triggers (if present)

Note

Always drop the indexes on the table before dropping the table. If you drop the table first, the indexes still exist.

Viewing text index status

To get information about the current text indexes within the database, use the views. For example, if you want to know about the current database defaults, use the following command:

db2 "select * from db2ext.dbdefaults"

For information about the currently available indexes, their corresponding tables, and the number of indexed documents, use this command:

db2 "select indschema, indname, tabschema, tabname, number_docs from db2ext.textindexes"

Use this command for information about the formats of a specific index:

db2 "select format, modelname from db2ext.textindexformats where indschema = 'DB2EXT' and indname = 'TITLE'"

For further information, see Appendix C, “Net Search Extender information catalogs”, on page 181.
Backing up and restoring indexes

Use the following steps to back up enabled databases and text indexes created by DB2 Net Search Extender:

1. To find out which indexes DB2 Net Search Extender has created and where they are stored, call a select statement on the db2ext.textindexes view:
   
   ```
   db2 "select indschema, indname, indexdirectory from db2ext.textindexes"
   ```

2. Ensure that no index update is running, and then stop DB2 Net Search Extender services with the following command:
   
   ```
   db2text stop
   ```

3. After backing up the database, back up the index directories and subdirectories.

4. Restart DB2 Net Search Extender services with the following command:
   
   ```
   db2text start
   ```

Use the following steps to restore the enabled databases and text indexes created by DB2 Net Search Extender:

1. Stop DB2 Net Search Extender with the following command:
   
   ```
   db2text stop
   ```

2. Restore the backup copies of the index directories to the same path as before.

3. Restart DB2 Net Search Extender with the command:
   
   ```
   db2text start
   ```
Backing up and restoring indexes
Chapter 7. Using DB2 Control Center

Use DB2 Control Center to manage DB2 Net Search Extender administration functions, DB2 instances, databases and database objects such as tables, views, and user groups.

You can invoke the commands on different DB2 Control Center objects, for example:

- Instance objects
- Database objects
- Text index objects

The main elements of DB2 Control Center are the menu bar, toolbar, object tree, and content pane.

Alternatively, you can use the command line. For more information, see the following chapters:

- Chapter 6, “Creating and maintaining a text index”, on page 31
- Chapter 5, “Net Search Extender instance services”, on page 27
Using DB2 Control Centre

Note
To use the examples and the DB2 Control Center for Net Search Extender, a valid Net Search Extender licence must be installed on the database server.

Only DB2 Net Search Extender indexing and administration functions are found in this chapter. For information on using DB2 Control Center, see “Related information” on page ix.

Starting and stopping DB2 Net Search Extender Instance Services

From the object tree, click on a system to display the available instances. Highlight the instance and right-click to display the instance object pop-up menu. Highlight Net Search Extender and select one of the following commands from the pop-up menu:

Start DB2 Net Search Extender Instance Services
This starts the instance services if they are not already started.

Stop DB2 Net Search Extender Instance Services

![Stop Net Search Extender Services dialog](image)

This displays a dialog. Use the check box to stop the instance services and index processes. In the command syntax, this is known as the FORCE option. Click on the OK button.

Instance status
This displays a dialog showing the status of the instance.

See Chapter 11, “Administration commands for the instance owner”, on page 93 for further information.
Enabling a database

In the object tree, click on the instance object to display the available databases. Highlight the database and right-click to display the pop-up menu. Highlight Net Search Extender and select one of the following commands from the extended menu:

Enable the database for text
This displays a dialog if the database is not enabled. Click on the OK button to enable the database. If the database is enabled, a message box displays.

Disable the database for text
This displays a dialog if the database is not disabled. Click on the OK button to disable the database. If the database is already disabled, a message box displays.

Click on the check box if you want to disable the database and drop all the text indexes.

Change the Data Link return size
This displays a dialog showing the current Data Link return size. Enter the new Data Link value in kilobytes (KBs) and click on the OK button.

See Chapter 12, “Administration commands for the database administrator”, on page 99 for further information.

Note that in all the dialogs, the Show Command button displays the command line alternative.

Text index administration

In the object tree, below the database object you can see the text index object. Click on the text index object to view the text indexes in the content pane.
Text index administration

Right-click on the text index object and select one of the following commands from the pop-up menu:

Create  This displays a wizard for creating a text index. See "Creating a text index" on page 55 for more information.

Filter  This displays a dialog where you can select which text index objects display in the control pane view.

Refresh  This refreshes information in the object tree and control pane.

To maintain text indexes, see "Maintaining a text index" on page 67

Note  
To access the instance, database and text index object commands without using the right-click option, click on the Selected menu command and highlight Net Search Extender to access the relevant commands.

Before creating a text index, ensure that you have considered the prerequisites found in Chapter 4, "Planning", on page 23

Other indexing prerequisites include:
- Starting DB2 Net Search Extender Instance services
- Enabling the database
Creating a text index

Select the Create command, and a Create Text Index Wizard pops up. Use the wizard to specify the configuration options for the text index in a number of panels. To move between the panels, enter all the mandatory information and click on the Next button until the Finish button is enabled. Click on the Finish button to create the text index.

To create a text index on views, use the CREATE INDEX command described in "CREATE INDEX" on page 114.

Name panel
This panel allows you to specify the schema and name for the text index. You can also specify a work and index directory for the text index files. Create the administrative tables for the index on the administration tablespace.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Mandatory/Optional</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Figure 10. Create Text Index Wizard: Name panel</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Here is a description of the fields in the panel:

Table 1. Name panel text fields
Creating a text index

Table 1. Name panel text fields (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Mandatory</th>
<th>user ID</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Index schema</td>
<td>Mandatory</td>
<td>user ID</td>
<td>Select a schema name of the text index. This is the DB2 schema name for the index-specific administration tables.</td>
</tr>
<tr>
<td>Index name</td>
<td>Mandatory</td>
<td>N/A</td>
<td>Enter a valid DB2 index name for the text index. With the index schema, this uniquely identifies a full-text index in the database.</td>
</tr>
<tr>
<td>Index directory</td>
<td>Optional</td>
<td>See the path name</td>
<td>Specify the directory path where you will store the text index. The directory must exist with read, write, and run permissions for the DB2 instance owner user ID.</td>
</tr>
<tr>
<td>Work directory</td>
<td>Optional</td>
<td>See the path name</td>
<td>Specify the work directory where you will store temporary files during search and administration operations. The directory must exist with read, write, and run permissions for the DB2 instance owner user ID.</td>
</tr>
<tr>
<td>Administration tablespace</td>
<td>Optional</td>
<td>Use the default tablespace</td>
<td>Select a tablespace name for the text index administration tables. You must define the tablespace on the same node group as the tablespace for the user table.</td>
</tr>
</tbody>
</table>

**Target panel**
This panel allows you to specify the schema and name of the table or nickname table, and the name of the text column containing the data you want to index. You can use a transformation function to modify the content of the text column. In addition to the text column, you can also specify numeric attributes if you want to index content of a table column expression.
Here is a description of the fields in the panel:

**Table 2. Target panel text fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Mandatory/Optional</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table schema (1)</td>
<td>Mandatory</td>
<td>user ID</td>
<td>Select the schema of the table or nickname table on which you are creating a text index.</td>
</tr>
<tr>
<td>Table name (2)</td>
<td>Mandatory</td>
<td>N/A</td>
<td>Select the name of the table or nickname table on which you are creating an index. The table must have a primary key.</td>
</tr>
<tr>
<td>Text column (3)</td>
<td>Mandatory</td>
<td>N/A</td>
<td>Select the name of the column used for creating the text index. The column must be transformed to, or be one of the following types: CHAR (for bit data), VARCHAR (for bit data), LONG VARCHAR (for bit data), CLOB, DBCLOB, BLOB, GRAPHIC, VARGRAPHIC, LONG VARGRAPHIC, and DATALINK.</td>
</tr>
<tr>
<td>Transformation function</td>
<td>Optional</td>
<td>Disabled</td>
<td>Select to use a transformation function.</td>
</tr>
</tbody>
</table>
Creating a text index

Table 2. Target panel text fields (continued)

<table>
<thead>
<tr>
<th>Transformation function: Schema</th>
<th>Mandatory (if function selected)</th>
<th>user ID</th>
<th>Select the schema of the UDF used to access the text documents.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transformation function: Name</td>
<td>As above</td>
<td>N/A</td>
<td>Select the name of a UDF used to access the text documents.</td>
</tr>
</tbody>
</table>

Note that you can only specify the table schema (1), table name (2) and text column (3) in this order.

Figure 12. Numeric Attributes dialog

To view or add attributes, click on the **Numeric Attributes** button. A window displays. To add numeric attributes to the index, click on the **Add** button and a further window displays. Specify the SQL column expression and name for the attribute.

Alternatively, select an attribute and press the appropriate buttons to change, move, or remove an entry.
Explaining Numeric Attributes
Use Numeric Attributes to index column expressions in addition to the text column. For example, if you want to index the column date of type TIMESTAMP in addition to the text column, specify a numeric attribute "cast(julian_day(date) as double)" and specify a name for the attribute.

Specify a numeric attribute if you want to use a numeric expression inside a search query. If you are searching with SQL queries, you can use a combined search instead of using numeric attributes, for example: WHERE numattrib = 123 AND contains('...'). If you are searching using a stored procedure, you must specify numeric attributes if you want to search additional information to the text documents.

Text Properties panel
This panel allows you to specify the language and format of the text documents. If the documents are not the same CCSID as the database and the text column is of binary type, specify the CCSID. Note that the database CCSID is initially selected. If your documents are a GPP, HTML or XML structured format, you can specify a document model.
Creating a text index

Figure 13. Create Text Index Wizard: Text Properties panel

Here is a description of the fields in the panel:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Mandatory/Optional</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Language</td>
<td>Optional</td>
<td>EN_US</td>
<td>Select a language to determine end-of-sentence and end-of-paragraph delimiters when indexing documents.</td>
</tr>
<tr>
<td>CCSID</td>
<td>Optional</td>
<td>CCSID of database</td>
<td>Select the CCSID for indexing text documents.</td>
</tr>
<tr>
<td>Format</td>
<td>Optional</td>
<td>TEXT</td>
<td>Select the text document format: HTML, XML, TEXT, or GPP.</td>
</tr>
<tr>
<td>Default Document Model</td>
<td>Optional</td>
<td>Enabled</td>
<td>Use the default document model.</td>
</tr>
<tr>
<td>User Document Model</td>
<td>Optional</td>
<td>Disabled</td>
<td>Use your document model.</td>
</tr>
</tbody>
</table>
Creating a text index

Table 3. Text Properties panel text fields (continued)

<table>
<thead>
<tr>
<th>Model name</th>
<th>Mandatory (if User Document Model selected)</th>
<th>N/A</th>
<th>Enter the name of the document model. For HTML, XML, and GPP formats, you can specify a document model. Note that the name is only found in the model file.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model file</td>
<td>As above</td>
<td>N/A</td>
<td>Specify the document model file. The file must be readable by DB2 instance owners.</td>
</tr>
<tr>
<td>Model CCSID</td>
<td>As above</td>
<td>Database CCSID</td>
<td>Select the CCSID to interpret the contents of the document model file.</td>
</tr>
<tr>
<td>Treat numbers as words</td>
<td>Optional</td>
<td>Disabled</td>
<td>Select to interpret sequences of digits as separate words, even if they are adjacent to characters.</td>
</tr>
<tr>
<td>Index stopwords</td>
<td>Optional</td>
<td>Enabled</td>
<td>Select to enable language-specific stopword processing. The <code>&lt;language&gt;.tsw</code> in the directory <code>sql1ib/db2ext/resources</code> contains the stopword list.</td>
</tr>
</tbody>
</table>

Update characteristics panel
This panel allows you to specify whether the index updates incrementally or is recreated from scratch. You can specify update settings so that the index automatically updates at the specified time.
Creating a text index

Figure 14. Create Text Index Wizard: Update Characteristics panel

Here is a description of the fields in the panel:

Table 4. Update Characteristics panel text fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Mandatory/Optional</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Incremental update</td>
<td>Optional</td>
<td>Enabled</td>
<td>Select for incremental index updates. If you do not enable the check box, you recreate the index when an update operation is performed.</td>
</tr>
<tr>
<td>Commitcount</td>
<td>Optional or Mandatory</td>
<td>0</td>
<td>Number of changes processed during an update in one transaction.</td>
</tr>
<tr>
<td>Reorg automatic or manual radio button</td>
<td>Optional or Mandatory</td>
<td>Enabled/disabled</td>
<td>Completes index reorganization automatically or manually.</td>
</tr>
<tr>
<td>Minimum number of changes for Update</td>
<td>Optional</td>
<td>1</td>
<td>Specify the minimum number of changes to the text documents before the index incrementally updates at the specified time.</td>
</tr>
</tbody>
</table>
Creating a text index

Table 4. Update Characteristics panel text fields (continued)

<table>
<thead>
<tr>
<th>Update schedule</th>
<th>Optional</th>
<th>Disabled</th>
<th>Select to add automatic update settings.</th>
</tr>
</thead>
</table>

To add index update settings, click on the **Settings** button. Note that this button is only enabled if you select **Automatic Update**. In the dialog, select the days, hours, and minutes for the update time. Note that if you select multiple days, the update occurs at the same time on all the selected days.

**Cache table panel**

This panel allows you to specify a cached table in addition to the index. You can specify the result columns to be cached and you can search the cache using a stored procedure. You can also specify other cache parameters, such as type, maximum size, and the order in which you retrieve the contents of the user table during initial indexing.

Figure 15. Create Text Index Wizard: Cache Table panel
Creating a text index

Here is a description of the fields in the panel:

Table 5. Result Cache panel text fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Mandatory/ Optional</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable table cache</td>
<td>Optional</td>
<td>Disabled</td>
<td>Select to enable the building of a cached table.</td>
</tr>
<tr>
<td>Result column table</td>
<td>Mandatory</td>
<td>N/A</td>
<td>Displays a list of SQL column expressions specifying the search result columns.</td>
</tr>
<tr>
<td>Maximum number of documents</td>
<td>Mandatory</td>
<td>Row count of table</td>
<td>See the following section: Determining cache utilization and cache size.</td>
</tr>
<tr>
<td>Average cache row length</td>
<td>Mandatory</td>
<td>N/A</td>
<td>See the following section: Determining cache utilization and cache size.</td>
</tr>
<tr>
<td>Percentage of initial cache utilization</td>
<td>Optional</td>
<td>50%</td>
<td>Select the percentage of the cache held free for additional documents.</td>
</tr>
<tr>
<td>Maximum cache size</td>
<td>Optional</td>
<td>N/A</td>
<td>Specify a maximum size for the cached table built during index activate. If the number is too small, the activation will fail.</td>
</tr>
<tr>
<td>Initial search result order</td>
<td>Optional</td>
<td>Disabled</td>
<td>Select to define the search result order. By skipping the dynamic ranking of full-text search results, documents are returned in the same indexing order as in the cached table. This order cannot be ensured after incremental update.</td>
</tr>
<tr>
<td>Use persistent cache</td>
<td>Optional</td>
<td>Enabled</td>
<td>This option enables a fast activate execution after a deactivation or system reboot. Note that you must specify a directory path for the persistent cache. Leave disabled if the cache should be temporary.</td>
</tr>
</tbody>
</table>
Determining cache utilization and cache size

The Percentage of initial cache utilization specifies the percentage of the cache to be held free for additional documents. The Maximum cache size specifies the maximum size of the cached table to be built during activate cache. These options depend on the following factors:

- The actual number of documents in the table.
- The expected number of updates.
- The average size of the SQL expressions you want to cache.

You can enter the recommended values for the Percentage of initial cache utilization and Maximum cache size, or let them be calculated each time you enter values in the Maximum number of documents or Average cache row length fields.

The Maximum number of documents value is initially set to the row count of the table. Modify this according to the number of documents and expected number of changes. Include all document updates, additions, and deletions.

When you add an SQL expression to the cache table list, the Average cache row length is calculated according to the length of the result. If you know that on average this value is smaller, modify the value.

For example, if your table has 10 entries and the sum of your column expressions is 100, then these values are initially set. If you expect that the maximum number of documents (including deleted ones) is 10 000, enter this figure. If you know that column expressions on average are smaller than the calculated value, such as a varchar(100) and a filled-in text size of 10, use this figure for the average row size.

To define the initial search result order, click on the Define button. Note that this button is only enabled if you select the Initial Search Result Order check box. A dialog displays all the specified SQL column expressions. To add a result order, click on the Add button and, in the dialog, specify the SQL result order.

To change, move, or remove an entry, select the expression and click on the appropriate buttons.
Creating a text index

To add SQL column expressions, click on the Add button, next to the Result Column table. In the dialog, specify the result column expression and name.

To change or remove an entry, click on the column expression which enables the appropriate buttons.

**Figure 16. Initial Search Result Order dialog**

**Figure 17. Change column expression dialog**

**Summary panel**
This panel provides an overview of the previously selected parameters.
Click on the **Show Command** button to view the commands that are run when you click on the **Finish** button. This action creates the text index.

### Maintaining a text index

To maintain the text indexes, select the text index in the control pane and click on the **Select** menu command. You can select one of the following commands from the menu:

1. The **ALTER** command, for altering a text index.
2. The **DROP** command, for dropping a text index.
3. The **UPDATE** command, for updating a text index.
4. The **SHOW INDEX EVENTS** command, for showing index events.
5. The **ACTIVATE INDEX MEMORY** command, for activating an index cache.
6. The **DEACTIVATE INDEX MEMORY** command, for deactivating an index cache.
7. The **SHOW STATUS** command, for showing the index status.

Note that the Activating and Deactivating commands only display if you create the index with a cache option.
Note
The information in each of the panels corresponds to the examples found in "Creating a text index" on page 55.

The Show Command button on each of the dialogs displays the command line alternative.

Altering a text index
Select the Alter command and a dialog displays a series of panels. These provide an overview of the parameters of the text index. Note that you can not change all of the parameters.

![Figure 19. Alter Index dialog: Name tab](image)

The Name panel displays the name and storage configurations for the index. You can change the index and work directories.

The Target panel displays the target and numeric attribute settings of the index. You cannot change these configurations.

The Text panel displays the text document configurations. You cannot change these settings.
The **Update** panel displays the update characteristics of the index configuration. You can alter the update schedule. If the index was created for incremental updates, you can also modify the commitcount and the minimum number of changes.

---

### Update Tab

<table>
<thead>
<tr>
<th>Incremental Update</th>
</tr>
</thead>
<tbody>
<tr>
<td>Commitcount: 0</td>
</tr>
<tr>
<td>Reorganize:</td>
</tr>
<tr>
<td>Manual</td>
</tr>
<tr>
<td>Minimum number of changes for Update: 1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Update Schedule</th>
</tr>
</thead>
<tbody>
<tr>
<td>Days</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Monday</td>
</tr>
<tr>
<td>Tuesday</td>
</tr>
<tr>
<td>Wednesday</td>
</tr>
<tr>
<td>Thursday</td>
</tr>
<tr>
<td>Friday</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

**Figure 20. Alter Index dialog: Update tab**

The **Update** panel displays the update characteristics of the index configuration. You can alter the update schedule. If the index was created for incremental updates, you can also modify the commitcount and the minimum number of changes.
Maintaining a text index

![Cache Table panel](image)

The Cache Table panel displays the cache option settings. If the result cache is already enabled, you can modify the persistent directory, or make the index cache temporary. You might also change the Maximum cache size and the Percentage of initial cache utilization.

Dropping a text index

Select the Drop command and a dialog displays the available text indexes.

![Drop Index dialog](image)
Maintaining a text index

Select the index and click on the OK button.

Updating a text index
Select the Update command and a dialog displays a number of update options.

![Update Index dialog](image)

Figure 23. Update Index dialog

You can specify a commitcount for the update operation. If you want to update the minimum specified during create or alter index, select the Minimum number of changes check box. To reorganize the index, select the check box. Note that if reorganization is suggested, the check box is enabled.

Showing Index events
Select the Show index events command and the contents of the event table display in the dialog.

![Index Events dialog](image)

Figure 24. Index Events dialog

To clear the index events, click on the Reset button.
Maintaining a text index

Activating a text index cache
Select the Activate index memory command and a dialog displays.

![Activate the cache dialog](image)

To activate the cache, click on the OK button. If you want to build the cache from scratch, select the check box.

Deactivating a text index cache
Select the Deactivate index memory command and a dialog displays.

![Deactivate the cached table dialog](image)

To release the cache, click on the OK button.

Showing Index status
Select the Show status command and a dialog displays the status of the text index.

![Index Status dialog](image)

This includes information on the number of indexed documents, the reorganization suggested flag, and additional index information.
Chapter 8. Searching

DB2 Net Search Extender provides the following methods for searching text:

**SQL scalar search functions**
These enable you to include text search subqueries in SQL queries.
Net Search Extender provides these functions in addition to those normally available in SQL.

**A stored procedure search function**
This enables you to return predefined result tables.

**An SQL Table-Valued Function**
You can use this search in a similar way to the stored procedure.

For SQL scalar search functions, the chapter describes the following areas:

- Searching for text, using the CONTAINS, NUMBEROFMATCHES, and SCORE functions.
  
  Refer to [Chapter 15, “SQL scalar search function and the SQL table-valued function”, on page 147](#) for a description of the syntax.

- Specifying search arguments by using examples with the CONTAINS function.
  
  Refer to [Chapter 14, “Syntax of search arguments”, on page 137](#) for a description of the syntax.

For the stored procedure search function, the chapter describes the following areas:

- Searching for text using the stored procedure search.
- For specifying search arguments, refer to [Chapter 14, “Syntax of search arguments”, on page 137](#) for a description of the parameters.

For the SQL Table-Valued Function, the chapter describes the following areas:

- Searching for text using the SQL Table-Valued Function.
- For specifying search arguments, refer to [Chapter 14, “Syntax of search arguments”, on page 137](#) for a description of the parameters.

There is also information on search performance considerations that may need to be taken into account.

Before searching, ensure that all the appropriate indexing steps, described in [Chapter 6, “Creating and maintaining a text index”, on page 31](#) involving the different data types are considered.
Searching

Note

The system shell interprets special characters such as ?, ( ), *, !, and ". Therefore, if the command contains these characters, you must use quotation marks or an escape character.

Here is an example of a UNIX command that uses special characters:

db2 "SELECT * from sample WHERE CONTAINS (DESCRIPTION, '"enable"') = 1"

Searching for text using SQL scalar search functions

Using examples, this section describes how to use the SQL scalar search functions in the following ways:

• Using the function CONTAINS to make a query.
• Using the function NUMBEROFMATCHES to determine how many matches were found in a text document.
• Using the function SCORE to get the relevancy of a found text document.

Refer to Chapter 15, “SQL scalar search function and the SQL table-valued function”, on page 147 for a description of the syntax.

Making a query

This example demonstrates how the CONTAINS function searches for text in column comment in table texttab. It returns 1 if the text satisfies the search argument, otherwise it returns 0.

SELECT AUTHOR,TITLE
FROM DB2EXT.TEXTTAB
WHERE CONTAINS(COMMENT, '"book"') = 1

In this example, you search for the term book in the column COMMENT.

Tip

If you create mixed-case identifiers for tables or columns, you must enclose these names in double quotes:

SELECT AUTHOR,TITLE
FROM "Texttab"
WHERE CONTAINS(COMMENT, '"book"') = 1

Searching and returning the number of matches found

Use the NUMBEROFMATCHES function to determine how often the search value is found in each text document.
Searching for text using SQL scalar search functions

```sql
SELECT AUTHOR, TITLE,
       NUMBEROFMATCHES(COMMENT, '"book"')
FROM DB2EXT.TEXTTAB
```

NUMBEROFMATCHES returns an integer value.

**Searching and returning the score of a found text document**

SCORE is an absolute value that indicates how well the document meets the search value relative to other found documents. The value indicates the number of matches that are found in the document in relation to the document’s size. In the following example, you can get the score of a found document by using the SCORE function:

```sql
WITH TEMPTABLE(docid,score)
    AS (SELECT docid,
          SCORE(COMMENT, '"book"')
          FROM DB2EXT.TEXTTAB)
SELECT *
FROM TEMPTABLE
WHERE score > 0
ORDER BY score ASC
```

SCORE returns a DOUBLE value between 0 and 1.

---

**Note**

Note that you cannot use the CONTAINS, SCORE, and NUMBEROFMATCHES search functions for indexes created on views.

---

**Specifying SQL search arguments**

The CONTAINS, NUMBEROFMATCHES, and SCORE functions all use search arguments. This section uses the CONTAINS function to show different examples of search arguments in DB2 Net Search Extender functions.

Refer to [Chapter 14, “Syntax of search arguments”, on page 137](#) for a description of the syntax.

**Searching for terms in any sequence**

You can have more than one term in a search argument. One way to combine several search terms is to connect them together using commas, like this:

```sql
SELECT AUTHOR, TITLE
FROM DB2EXT.TEXTTAB
WHERE CONTAINS(COMMENT, '("kid", "dinosaur")' ) = 1
```

This form of search argument finds text that contains any of the search terms. In logical terms, an OR operator connects the search terms.
Searching with the Boolean operators AND and OR

You can combine search terms with other search terms using the Boolean operators “&” (AND) and “|” (OR):

```
SELECT AUTHOR, TITLE
FROM DB2EXT.TEXTTAB
WHERE CONTAINS(COMMENT, "author" | "pulitzer") = 1
```

You can also combine several terms by using Boolean operators:

```
SELECT AUTHOR, TITLE
FROM DB2EXT.TEXTTAB
WHERE CONTAINS(COMMENT, "author" | "pulitzer" & "book") = 1
```

If you use more than one Boolean operator, these are evaluated from left to right. However, the logical AND operator (&) binds stronger than the logical OR operator (|). You can see this evaluation in the following example, which does not include parentheses:

"book" & "pulitzer"| year" & "author"

Therefore, Net Search Extender evaluates the boolean operators in the following way:

```
("book" & "pulitzer") | (year & "author")
```

So, to correctly evaluate the boolean operators, you must include parentheses:

```
"book" & ("pulitzer" | year") & "author"
```

You can also combine Boolean operators with search terms that are chained together using the comma separator:

```
("author", "pulitzer") & "book"
```

In this case, however, the comma is interpreted as a Boolean OR operator:

```
("author"| "pulitzer") & "book"
```

For additional information, also see “Searching with the Boolean operator NOT” on page 78.

Searching for parts of a term (character masking)

Masking characters, otherwise known as “wildcard” characters, offer a way to make a search more flexible. They do this by increasing the number of text documents that are found by a search.

DB2 Net Search Extender uses two masking characters: percent (%) and underscore (_).

- % represents any number of arbitrary characters. Here is an example of % used as a masking character in the middle of a search term:
Specifying search arguments

SELECT AUTHOR,TITLE
FROM DB2EXT.TEXTTAB
WHERE CONTAINS(COMMENT, '"thr\%er"') = 1

This search term finds text documents containing the word “thriller”.

• _ represents one character in a search term. The following example also finds text documents containing the word “thriller”.

SELECT AUTHOR, TITLE
FROM DB2EXT.TEXTTAB
WHERE CONTAINS(COMMENT, '"th\_iller"') = 1

Note
Use wildcard characters sparingly as they can increase the size of your result list significantly, thus decreasing performance and returning unexpected search results.

Searching for terms that already contain a masking character

If you want to search for a term that contains the “%” character or the “_” character, you must precede the character by a so-called escape character. Then you can identify the escape character using the ESCAPE keyword.

In the following example, the escape character is a "!":

SELECT AUTHOR, TITLE
FROM DB2EXT.TEXTTAB
WHERE CONTAINS(COMMENT, '"100!%" ESCAPE "!"') = 1

Searching for terms in a fixed sequence

If you search for “primary key”, you will only find the two terms if they are adjacent and occur in the sequence shown:

SELECT AUTHOR, TITLE
FROM DB2EXT.TEXTTAB
WHERE CONTAINS(COMMENT, '"primary key"') = 1

Searching for terms in the same sentence or paragraph

Here is an example of a search argument that finds text documents in which the search terms occur in the same sentence:

SELECT AUTHOR, TITLE
FROM DB2EXT.TEXTTAB
WHERE CONTAINS(COMMENT, '"web" IN SAME SENTENCE AS "disk"') = 1

You can also search for more than two words occurring together. In the next example, a search is made for two words occurring in the same paragraph:
Specifying search arguments

SELECT AUTHOR, TITLE
FROM DB2EXT.TEXTTAB
WHERE CONTAINS(COMMENT,
"computer" IN SAME PARAGRAPH AS "web") = 1

Searching for terms in sections of structured documents

Here is an example of a search argument that finds text documents where search term IBM occurs in the subsection H2 of structured documents.

SELECT CATEGORY, DATE
FROM DB2EXT.HTMLTAB
WHERE CONTAINS(HTMLFILE,
'SECTIONS ("H2") "IBM"') = 1

Note that section names are case-sensitive. Ensure that the section name in the model file and in the query are identical. See Chapter 9, “Working with structured documents”, on page 83 for more information.

Searching with the Boolean operator NOT

You can use the Boolean operator NOT to exclude particular text documents from the search:

SELECT AUTHOR, TITLE
FROM DB2EXT.TEXTTAB
WHERE CONTAINS(COMMENT,
'("author", "pulitzer") & NOT "book") = 1

This example excludes any text documents containing the term “book” from the search for “author” or “pulitzer”.

Fuzzy search

“Fuzzy” search searches for words that are spelled in a similar way to the search term.

SELECT AUTHOR, TITLE
FROM DB2EXT.TEXTTAB
WHERE CONTAINS(COMMENT,
'fuzzy form of 80 "pullitzer"') =1

In this example, the search could find an occurrence of the misspelled word Pulitzer.

The match level, in the example “80”, specifies the degree of accuracy. Use a fuzzy search when misspellings are possible in the document. This is often the case when the document was created using an Optical Character Recognition device, or phonetic input. Use 1 to 100, where 100 is an exact match and below 80 is “fuzziness”.

Thesaurus search

Thesaurus search is a powerful search-term expansion function in DB2 Net Search Extender. The additional terms you search for are taken from a
Specify search arguments

thesaurus that you build yourself, so you have direct control over the terms. For example, a search for “database”, and could find terms like “repository” and “DB2”.

Use this type of search for specific areas of interest in which you make frequent searches and produce significantly more effective search results.

See Chapter 10, “Using a thesaurus to expand search terms”, on page 85 for more information and a description of how to build a thesaurus. The following example demonstrates the syntax for using thesaurus expansion.

This example takes the term “product” and expands it, adding all relations of this term found in the thesaurus “nsesamplethes”. Here, “marketing” is added to the search.

```sql
SELECT CATEGORY, DATE
FROM DB2EXT.HTMLTAB
WHERE CONTAINS(HTMLFILE,
  'THESAURUS "nsesamplethes"
  EXPAND RELATED
  TERM OF "product"')) = 1
```

The next example takes the search term “product”. The search then expands with all the synonyms of the search term.

```sql
SELECT CATEGORY, DATE
FROM DB2EXT.HTMLTAB
WHERE CONTAINS(HTMLFILE,
  'THESAURUS "nsesamplethes"
  EXPAND SYNONYM
  TERM OF "product"')) = 1
```

Numeric attribute search

On numeric attributes that are stored in a structured document, you can search using the following syntax:

```sql
SELECT AUTHOR, TITLE
FROM DB2EXT.TEXTTAB
WHERE CONTAINS(COMMENT,
  'ATTRIBUTE "PRICE" between 9 and 20') = 1
```

Note that attribute names are case-sensitive. Ensure that the attribute name in the model file and in the query are identical. See Chapter 9, “Working with structured documents”, on page 83 for more information.

Free-text search

“Free-text search” is a search in which you express the search term as free-form text. A phrase or a sentence describes in natural language the subject to be searched for. The sequence of words in a free-text query is not relevant. However, for a set of query terms, at least one of the terms must occur in the documents to be searched.
Specifying search arguments

Note that the masking of characters or words is not supported for search strings in a free-text argument.

For example:

```
SELECT AUTHOR, TITLE
FROM DB2EXT.TEXTTAB
WHERE CONTAINS(COMMENT,
'IS ABOUT EN_US "something related to dinosaur"') = 1
```

Additional search syntax examples

To become familiar with additional search syntax examples, use the command line processor input file called search. This contains examples of DB2 Net Search Extender search functions that run against the sample table.

To run the example, use the following syntax:

```
db2 -tvf search
```

If the table and indexes have not been created, run one of the following:

- On UNIX platforms: `nsesample` in the `<INSTOWNERHOME>/sqllib/samples/db2ext` directory.
- On Windows platforms: `nsesample (.bat)` in the `<sqllib>/samples/db2ext` directory.

Searching for text using a stored procedure search

Use the stored procedure search interface when your application needs a subset of the text search result, but in a high performance way. Do not use the stored procedure if all the results are required, or a large number of documents needs to be indexed. The reason being that parts of the user table are copied into memory and therefore, a lot of real memory needs to be available.

You can use the stored procedure to first request results from 0 to 20, then 21 to 40, and so on, in a similar way to cursor navigation. Combining this cursor capability with the use of a cache (calculated during indexing), searching is extremely fast, especially as no join is necessary.

If you are going to use the stored procedure, ensure that you consider the following options:

- The cache-search-result options have been specified during CREATE INDEX.
- The present and future shared memory requirements, possibly involving incremental updates, have been fully considered. See "Creating a text index which the stored procedure search can use" on page 38 for further information.
Additional search syntax examples

- The cache of the index has been activated using `db2text activate` command.

The following is an example of a stored procedure search:

```
db2 "call db2ext.textSearch('"book"','DB2EXT','COMMENT',0,2,1,1,?,?)"
```

The first parameter is the search term. The syntax is exactly the same as you have in the SQL functions. Then specify the index name. If you have not masked the name, it is translated to uppercase. The following two numbers give you the probability of getting the result in slices. The next two integer values specify if score is requested and how often a term is found in the index. The last two values are output values.

See Chapter 16, “Stored procedure search function”, on page 155 for further details on the parameters.

---

**Note**

If you request larger result sets, you need a user tablespace. If there are none available, create a tablespace. The following example creates a tablespace on a UNIX platform:

```
db2 "create user temporary tablespace tempts managed by system
using ('/work/tempts.ts')"
```

---

**Searching for text using an SQL Table-Valued Function**

Use the SQL Table-Valued Function where you implement an interface, but do not need all results back, or do not have all the real memory required to use the stored procedure interface.

The SQL Table-Valued Function gives you the same cursor interface as the stored procedure to get only parts of the result. However, you still need to join the results with the user table. You can see this in the following example:

```
db2 "select docid , author, score from TABLE(db2ext.textsearch('"book" ',
'DB2EXT','COMMENT',3,2,cast(NULL as integer))) as t, db2ext.texttab u
where u.docid = t.docid""
```

The following are the values you could return from the SQL Table-Valued Function:

- `--> primKey <single primary key type>`
  
  the primary key

- `--> score DOUBLE`

  the score value of the found document total

- `--> NbResults INTEGER`
Additional search syntax examples

the total number of found results (same value for all rows)

--> numberofmatches INTEGER
--> the hitcount in this document

Note that only a single primary key column is allowed. See the "DB2EXT.TEXTSEARCH" on page 151 for further details on the parameters.

Performance considerations

To enhance performance during search, consider the following issues:

• When searching within SQL:
  – Use the explain statement to check the processing plan of the DB2 optimizer, if you notice a decrease in performance.
  – Parametric search can make searching faster. However, if the other search predicates are used, reduce the result size.
  – Use a result limit if you do not require all of the results.
  – Use small primary key columns, such as timestamp and integer instead of varchar types.

• When searching with the stored procedure:
  – As the specified cache table expression is copied from the database into memory, ensure that your workstation has enough memory available for this data. If there is insufficient memory, paging space is used, which decreases search performance.

Note

For the latest performance tips, go to the DB2 Net Search Extender Web site: www.ibm.com/software/data/db2/extenders/netsearch/index.html
Chapter 9. Working with structured documents

DB2 Net Search Extender allows you to index and search text or numeric fields, such as title, author, or description in a structured document. The documents can be in XML or HTML format, or contain user-defined tags.

Use markup tags and their field names in a document model to define which fields in the documents are indexed and, therefore, are available for searching. You can use the name of the field (also known as the section name) in queries against that field.

To be able to search in these fields, you must specify a format and model file when you create the text index containing the documents. See "CREATE INDEX" on page 114 for further information.

For more information on creating and defining document models, see Chapter 17, “Structured document support”, on page 159. For the document model syntax, see Appendix G, “Document model reference”, on page 217.
Chapter 10. Using a thesaurus to expand search terms

You can broaden a query by searching not only for a specific search term, but also for terms that are related to it. You can automate this process by using Text Search Engine’s functions for looking up and extracting the related search terms from a thesaurus. A thesaurus is a controlled vocabulary of semantically related terms that usually covers a specific subject area.

DB2 Net Search Extender lets you expand a search term by adding additional terms from a thesaurus that you have previously created. Refer to Chapter 14, “Syntax of search arguments”, on page 137 to find out how to use thesaurus expansion in a query.

To create a thesaurus for using it in a search application requires a thesaurus definition file that has to be compiled into an internal format, the thesaurus dictionary.

This chapter describes:

• **The structure of a thesaurus**
  A thesaurus is structured like a network of nodes linked together by relations. This section describes Text Search Engine’s predefined relations and how to define your own relations.

• **Creating and compiling a thesaurus**
  Here is a description of the syntax of a thesaurus definition file, and of the tools that you use to compile it into a thesaurus dictionary.

The structure of a thesaurus

A thesaurus is structured like a network of nodes linked together by relations. The Text Search Engine looks up a term in a thesaurus by starting at the term, then following a path through the term relations and delivering the terms found in the process.
Working with a thesaurus

Figure 28. An example of the structure of a thesaurus

Thesaurus entries are connected by relations. Relation names, such as HIGHER_THAN, let you restrict an expansion to certain named lines in the relation hierarchy. Some relations are bidirectional, others are unidirectional; HIGHER_THAN, for example, is the name of a unidirectional relation.

Predefined thesaurus relations

These are the relations that are predefined in the Text Search Engine:

- **Associative relations**
  
  An associative relation is a bidirectional relation between two terms that do not express the same concept but relate to each other.
  
  Predefined associative relation: RELATED_TO
  
  Examples:

  tennis RELATED_TO racket
  
  football RELATED_TO goal (sports)

- **Synonym relations**
  
  A synonym relation is a bidirectional relation between two terms that have the same or similar meaning and can be used as alternatives for each other. This relation can, for example, be used between a term and its abbreviation.
  
  Predefined synonym relation: SYNONYM_OF
  
  Examples:
Working with a thesaurus

spot SYNONYM_OF stain
US SYNONYM_OF United States

Figure 28 on page 86 shows two goal terms in the same thesaurus. One is specified with the comment (sports), the other with the comment (abstract). Even if terms have the same spelling, synonym relations can connect different word groups. You can model this by using different relations when defining the thesaurus.

• Hierarchical relations

A hierarchical relation is a unidirectional relation between two terms, one of which has a broader (more global) meaning than the other. Depending on its direction, the relation can be used to look up either more specialized or more global terms.

Predefined hierarchical relations:

– LOWER_THAN to model narrowing relations
  LOWER_THAN relations are for modelling a sequence of more specialized terms. The deeper you follow a narrowing relation, the more specific the terms become. For example, if you look up the term house along a LOWER_THAN relation, the result could be skyscraper palace church chapel cathedral and so on, in a list of increasingly specialized terms.

– HIGHER_THAN to model broadening relations
  HIGHER_THAN relations are for modelling a sequence of more and more global terms. The deeper you follow such a relation, the less specific the terms become. For example, if you look up the term house along a HIGHER_THAN relation, the result could be building construction object and so on, in a list of increasingly global terms.

Defining your own relations

Net Search Extender lets you define your own RELATED_TO, LOWER_THAN, and HIGHER_THAN thesaurus relations. Because each relation name must be unique, you must qualify such relations names by the addition of a unique number, like this: RELATED_TO(42).

You can use the same relation number to define a relationship of a different type, such as LOWER_THAN(42). The number 0 is used to refer to Net Search Extender’s predefined relations.

Creating and compiling a thesaurus

Use the following steps to create a thesaurus that can be used by the Text Search Engine functions:

1. Create a thesaurus definition file.
2. Compile the definition file into a thesaurus dictionary.
Creating a thesaurus definition file

To create your own thesaurus, your first step is to define its content in a definition file using a text editor.

Restrictions. The length of the file name, including the extension, must not exceed 256 characters. You can have several thesauri in the same directory, but it is recommended that you have a separate directory for each thesaurus.

A sample English thesaurus definition file nsesamplethes.def is provided. The thesaurus directory for Windows systems is: <sqllib>\db2ext\thes

On UNIX systems, the thesaurus directory is: <DB2_INSTOWNER>/sqllib/db2ext/thes

Here are the first few definition groups from that file:

```
:WORDS
   accounting
   .RELATED_TO account checking
   .RELATED_TO sale management
   .SYNONYM_OF account
   .SYNONYM_OF accountant

:WORDS
   acoustics
   .RELATED_TO signal processing

:WORDS
   administration
   .RELATED_TO management method
   .RELATED_TO personnel management

:WORDS
   advertising
   .RELATED_TO sale promotion
   .SYNONYM_OF advertisement
```

Figure 29. An extract from the sample thesaurus definition file

For the syntax of each definition group, see Chapter 18, “Thesaurus support”, on page 171

Each member must be written to a single line. Each associated term must be preceded by the relation name. If the member terms are related to each other, specify a member relation.
The length of the member terms and associated terms is restricted to 64 characters. Single-byte characters and double-byte characters of the same letter are regarded as the same. Capital and small letters are not distinct. A term can contain a blank character and either a single-byte character period "," or colon ":" can be used.

The user-defined relations are all based on the associative type. They are identified by unique numbers between 1 and 128.

Compiling a definition file into a thesaurus dictionary

To compile a thesaurus definition file, run the db2extth command. For the command syntax, see “DB2EXTTH (Utility)” on page 130.

Tip
See Appendix M, “Messages returned by the thesaurus tools”, on page 239.
Working with a thesaurus
Part 2. Reference
Chapter 11. Administration commands for the instance owner

This chapter describes the syntax of administration commands for the instance owner. Instance owner administration consists of checking the status of DB2 Net Search Extender locking and update services, and starting and stopping these services.

For additional information, see Chapter 5, “Net Search Extender instance services”, on page 27.

The commands are a variation of the DB2TEXT command and allow for the administration of DB2 Net Search Extender Services that are specific to a DB2 instance.

<table>
<thead>
<tr>
<th>Command</th>
<th>Purpose</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONTROL</td>
<td>Lists and deletes full-text index locks. Also lists the cache states.</td>
<td>94</td>
</tr>
<tr>
<td>START</td>
<td>Starts the DB2 Net Search Extender instance services.</td>
<td>96</td>
</tr>
<tr>
<td>STOP</td>
<td>Stops the DB2 Net Search Extender instance services.</td>
<td>97</td>
</tr>
</tbody>
</table>
CONTROL command

CONTROL

This command lets you list and delete full-text index locks managed by the Net Search Extender Instance Services. If the locking and update services are running, you can view their status as well as information on the activated cache.

Authorization

You must run this command as a DB2 instance owner on the server.

Command syntax

ControL CLEAR | set-of-locks |
LIST | set-of-locks |
SHOW-CACHE-STATUS-FOR | index-specification |
STATUS |

set-of-locks:
—ALL-LOCKS-FOR | database-specification |
| index-specification |

index-specification:
| database-specification | INDEX | index-schema-"." | index-name |

Database-specification:
—DATABASE | database-name |

Command parameters

CLEAR
Use CLEAR to force a cleanup for a set of locks. Use this command carefully after checking what leads to a locking problem.

LIST
Use LIST to get information about the current locks held for a specific index or database. If there is an update lock, you can get information about the documents that have been processed.

set-of-locks
Works with locks only in the specified database or index.

SHOW CACHE STATUS FOR
Shows the activation status for a cached table of the specified index. This can be either: "Not Activated" or "Currently Activated". If the cache is activated, it displays details about cache memory usage. For
CONTROL command

example, the maximum cache size (in megabytes), the maximum number of documents to insert, and the space left in the cache table (in kilobytes).

STATUS
By using the STATUS keyword, the command displays whether the locking and update Net Search Extender Instance Services are up and running.

DATABASE database name
The name of the database on the server that is being used.

INDEX index-schema.index-name
The schema and name of the text index that is currently being used. This is specified in the CREATE INDEX command.

Usage
When an administration command error message indicates that there is a locking problem, ensure that no conflicting task is running. For example, attempting an ALTER command while an UPDATE command is running. Then free all the locks for the index.

Use SHOW CACHE STATUS for an incremental index update to check that the specified memory size is still large enough to hold all the update information during the next update, or to check if an activation has been done.
START command

START

This command starts a demon that controls the locking of full-text indexes and the automatic updating of full-text indexes on the DB2 server.

Note
As the command does not activate any temporary cached table for indexes, individual ACTIVATE CACHE commands are necessary for searching with a stored procedure.

Authorization
You must run this command as a DB2 instance owner on a server.

Command syntax
```
START
```

Command parameters
None.

Usage
On Windows, the command starts a service \texttt{db2ext-<InstanceName>}. You can also start this command using normal Window methods.

For locking of full-text indexes, you can modify a configuration file to meet your requirements. See “Locking services” on page 27 for further information.
STOP command

STOP

This command stops the locking and update services of Net Search Extender.

Authorization
You must run this command as a DB2 instance owner on a server.

Command syntax

STOP (FORCE)

Command parameters

FORCE
Stops services even if processes are holding locks or if the cached table is activated for any index. If you do not specify FORCE, the command will fail in these cases.

Usage
Stopping the Net Search Extender Instance Services will not allow any further use of specific Net Search Extender commands. When restarting the services, you must activate the temporary cache again.

Note that the activated cache or running Net Search Extender commands will not stop services.
STOP command
Chapter 12. Administration commands for the database administrator

This chapter describes the syntax of administration commands for the database administrator. Database administration consists of setting up databases for use by DB2 Net Search Extender and then disabling this setup.

Chapter 6, “Creating and maintaining a text index”, on page 31 describes how to use these commands.

Only the ENABLE DATABASE and DISABLE DATABASE commands are a variation of the DB2TEXT command, although all these commands allow for administration on the database level.

<table>
<thead>
<tr>
<th>Command</th>
<th>Purpose</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENABLE DATABASE</td>
<td>Enables the current database to create full-text indexes.</td>
<td>100</td>
</tr>
<tr>
<td>DISABLE DATABASE</td>
<td>Resets preparation work completed by DB2 Net Search Extender for a database.</td>
<td>102</td>
</tr>
<tr>
<td>DB2EXTDL (utility)</td>
<td>The default UDF to retrieve the content of a data link text column.</td>
<td>104</td>
</tr>
</tbody>
</table>

**Tip**

The executable db2text causes an implicit connection to be made to the default database specified in the environment variable DB2DBDFT.
ENABLE DATABASE command

ENABLE DATABASE

This command enables a database to create and exploit full-text indexes on text columns.

Authorization

You must run this command as a database administrator to enable the database. This requires you having SYSADM authority to be able to grant DBADM to the DB2 instance owner.

Command syntax

```
ENABLE-DATABASE-FOR-TEXT
connection-options
```

connection-options:

```
CONNECT-TO database-name
USER userid USING password
```

Command parameters

CONNECT TO database-name
The name of the database that is a target for this command. You can omit this parameter, if DB2DBDFT is set and the user is running the command under a user ID with the necessary DB2 authorizations.

USER userid USING password
Use a password and userid to connect to the database.

Usage

This command prepares the connected database for use by DB2 Net Search Extender. It is a mandatory step before you can create a DB2 Net Search Extender index on tables/columns in the database.

You can view the database defaults established after running the command by using the DB2EXT.DB0EFAULTS catalog view.

Changes to the database

This command grants DBADM authority to the DB2 instance owner associated with the DB2 instance of the enabled database.

The ENABLE DATABASE command creates various database objects in the schema DB2EXT, such as DB2 Net Search Extender catalogs, UDFs, and stored procedures. After running the command, the following catalog views are available:
ENABLE DATABASE command

- `db2ext.dbdefaults`
- `db2ext.textindexes`
- `db2ext.textindexformats`
- `db2ext.indexconfiguration`
- `db2ext.proxyinformation`

Note that DB2 Text Information Extender views are also available for backward compatibility reasons. See Appendix C, "Net Search Extender information catalogs", on page 181 for further information.

Also note that the above tables are located in the default tablespace of the database, known as `IBMDEFAULTGROUP`. This is distributed over all the nodes defined in `db2nodes.cfg`

Changes to the file subsystem
None.
DISABLE DATABASE command

DISABLE DATABASE

This command undoes DB2 Net Search Extender changes to a database.

Authorization

You must run this command as a database administrator to disable the database. This requires you having DBADM authority.

Command syntax

```plaintext
DISABLE-DATABASE-FOR-TEXT connection-options
```

connection-options:

```plaintext
CONNECT-TO database-name
USER userid USING password
```

Command parameters

CONNECT TO database-name

The name of the database that is a target for this command. You can omit this parameter, if DB2DBDFT is set and the user is running the command under a user ID with the necessary DB2 authorizations.

USER userid USING password

Use a password and userid to connect to the database.

FORCE

Forces the dropping of all DB2 Net Search Extender indexes in the database. See “DROP INDEX” on page 128 for more information.

Usage

This command resets the connected database, so that it can no longer be used by other DB2 Net Search Extender commands. If full-text indexes exist in the database, this command fails unless the FORCE option is used.

This command does not remove DBADM authority from the DB2 instance owner.

Changes to the database

The following modifications made in the database to enable DB2 Net Search Extender are deleted:

- The DB2 Net Search Extender catalog views in the database.
- All the database objects created by DB2 Net Search Extender.

Changes to the file system and shared memory

If you use the FORCE option, the index files are deleted.
If you use the FORCE option, the cache is deleted for any activated cache of indexes. See “DROP INDEX” on page 128 for further information.
DB2EXTDL (utility) command

DB2EXTDL (utility)

By default, the UDF that retrieves the content of a Data Link text column returns a 100 KB BLOB. Depending on the size of the largest document in the database referenced by the Data Link, you might increase or decrease this value.

Authorization

You must run this command as a database administrator to enable the database. This requires you having SYSADM authority to be able to grant DBADM to the DB2 instance owner.

Command syntax

```
DB2EXTDL new-result-size
```

Command parameters

- `new-result-size`
  
  The new result size of the UDF to retrieve the Data Link content in kilobytes. This is a positive integer of <2097152.
Chapter 13. Administration commands for the text table owner

This chapter describes the syntax of administration commands for the text table owner.

Chapter 6, “Creating and maintaining a text index”, on page 31 describes how to use these commands.

The commands are a variation of the DB2TEXT command. These allow the owner of a table to create and manipulate full-text indexes on columns of the table.

<table>
<thead>
<tr>
<th>Command</th>
<th>Purpose</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACTIVATE CACHE</td>
<td>Activates the cache so that search operations using the stored procedure are possible</td>
<td>106</td>
</tr>
<tr>
<td>ALTER INDEX</td>
<td>Changes the characteristics of an index</td>
<td>108</td>
</tr>
<tr>
<td>CLEAR EVENTS</td>
<td>Deletes index events from an index event table used during index update</td>
<td>112</td>
</tr>
<tr>
<td>CREATE INDEX</td>
<td>Creates a full-text index</td>
<td>114</td>
</tr>
<tr>
<td>DEACTIVATE CACHE</td>
<td>Deactivates the cache so that search operations using the stored procedure are no longer possible</td>
<td>126</td>
</tr>
<tr>
<td>DB2EXTTH (Utility)</td>
<td>Compiles the thesaurus definition file</td>
<td>130</td>
</tr>
<tr>
<td>DROP INDEX</td>
<td>Drops a full-text index for a text column</td>
<td>128</td>
</tr>
<tr>
<td>ENABLE DATABASE</td>
<td>Enables the current database to create full-text indexes</td>
<td>100</td>
</tr>
<tr>
<td>UPDATE INDEX</td>
<td>Starts the indexing process based on the current contents of the text columns</td>
<td>132</td>
</tr>
<tr>
<td>HELP</td>
<td>Displays the list of DB2TEXT command options</td>
<td>135</td>
</tr>
<tr>
<td>COPYRIGHT</td>
<td>Displays the Net Search Extender product and copyright information</td>
<td>136</td>
</tr>
</tbody>
</table>

Tip
The executable db2text causes an implicit connection to be made to the default database specified in the environment variable DB2DBDFT.
**ACTIVATE CACHE command**

**ACTIVATE CACHE**

This command activates the cached table from either the DB2 user table or the persistent cache. After completion, search operations using the stored procedure are possible. See Chapter 16, “Stored procedure search function”, on page 155 for further information.

**Authorization**

According to the DB2 catalog views, the user ID in this command must have CONTROL privilege on the table for which the full-text index was created.

**Command syntax**

```
ACTIVATE CACHE FOR INDEX
  index-schema."" index-name FOR-TEXT
  RECREATE |connection-options|
```

**connection-options:**

```
CONNECT-TO database-name USER userid USING password
```

**Command parameters**

**index-schema**

The schema of the text index, as specified in the `CREATE INDEX` command. If no schema is specified, the user ID of the DB2 connection is used.

**index-name**

The name of the text index, as specified in the `CREATE INDEX` command.

**RECREATE**

Applies only to indexes using a persistent cache; an existing cache is deleted. If an update without activation has completed, the persistent cache is automatically reconstructed from the database.

**CONNECT TO database-name**

The name of the database that is target for this command. You can omit this parameter, if `DB2DBDFT` is set and the user is running the command on the server. Note that the user ID must have the required DB2 authorizations.
USER userid USING password
Use a password and userid to connect to the database. If not specified, a connection is attempted from the current user ID without a password.

Usage
You cannot issue the command if one of the following commands is running on the index:
- UPDATE INDEX
- ALTER INDEX
- DROP INDEX
- CLEAR EVENTS
- DEACTIVATE CACHE

Note
Activation of a cached table may require its recreation from scratch, even though a persistent cache was used. This occurs if an update operation was performed whilst the persistent cache was deactivated.

The amount of memory taken to build the cache is dynamically calculated from the current number of documents and the size of the result columns. Use the PCTFREE value to increase the calculated minimal amount of memory by a factor of 100/(100-PCTFREE). The PCTFREE value is specified in the CREATE or ALTER INDEX command.

Thereby, PCTFREE describes the percentage of the allocated cache that is reserved for insert operations while the cache is activated. Note that for each ACTIVATE CACHE command, the actual memory size is re-evaluated.

This command is only available if the index was created with a CACHE TABLE option. See “CREATE INDEX” on page 114 for further information.

Changes to the file system
Files for implementing the persistent cache are created.
ALTER INDEX command

ALTER INDEX

The command changes the characteristics of a full-text index, for example, the update options and the storage options.

Authorization

According to DB2 catalog views, the user ID in this command must have CONTROL privilege on the table for which the full-text index was created.

Command syntax

\[
\text{ALTER INDEX} \quad \text{INDEX-NAME} \quad \text{FOR-TEXT}\]

storage-options:

\[
\text{INDEX-DIRECTORY} \quad \text{WORK-DIRECTORY} \quad \text{CACHE TABLE} \quad \text{PERSISTENT}\]

update-characteristics:

\[
\text{UPDATE-FREQUENCY} \quad \text{UPDATE-MINIMUM}\]

update-frequency:

\[
0 \ldots 6 \quad 0 \ldots 23 \quad 0 \ldots 59
\]
ALTER INDEX command

**connection-options:**

```
CONNECT-TO database-name
USER userid USING password
```

**Command parameters**

**index-schema**

The schema of the text index as specified in the CREATE INDEX command. If no schema is specified, the user ID of the DB2 connection is used.

**index-name**

The name of the text index as specified in the CREATE INDEX command.

**INDEX DIRECTORY directory**

The directory path where the text index is stored. As the directory will contain index data, ensure that the directory has read/write and run permissions for the DB2 instance owner user ID.

A subdirectory, NODE0000, is created under the directory. Any index files from the previous index directory are deleted.

**WORK DIRECTORY workdirectory**

Stores temporary files during search and administration operations. You can change the separate work directory independently of a new index directory.

If the directory does not exist, it is created for the DB2 instance owner user ID. If it exists, ensure that the directory has read/write permissions on UNIX platforms for the instance owner.

A subdirectory, NODE0000, is created under the directory. Any temporary index files from the previous index directory are deleted.

**CACHE TABLE PERSISTENT IN directory**

Specifies that after a deactivation or system reboot, the cached table in CREATE INDEX is persistent. In either case, this allows for a fast ACTIVATE CACHE execution. The persistent cache is stored in the specified directory.

The previously created persistent cache is moved to a new location. This location always requires a deactivated index.

**CACHE TABLE TEMPORARY**

Specifies that the cached result table is now temporary and any previously existing persistent cache has been deleted. Note that this change requires a deactivated index.
ALTER INDEX command

MAXIMUM CACHE SIZE memsize
Specifies the new maximum size of the cached table to be built during ACTIVATE CACHE. Specify the memsize parameter in megabytes as a positive integer.

If the integer is too small, the ACTIVATE CACHE command fails. The actual cache size is calculated during the ACTIVATE CACHE command. This change requires a deactivated index.

PCTFREE percentage
Specifies the percentage of the cache held free for additional documents. The percentage must be an integer value less than 100 and greater or equal to 0. Note that the previous persistent cache is deleted and that this change requires a deactivated index. See “ACTIVATE CACHE” on page 106.

UPDATE FREQUENCY
Using the following parameters, the index update frequency determines when the update occurs:

- **D.** The day(s) of the week when the index is updated: * (everyday) or 0..6 (0=Sunday)
- **H.** The hour(s) when the index is updated: * (every hour) or 0..23
- **M.** The minute(s) when the index is updated: 0..59
- **NONE.** No further index updates occur. This is intended for a text column in which no further changes are made.

If you do not specify the UPDATE FREQUENCY keyword, the frequency settings are left unchanged.

UPDATE MINIMUM minchanges
The minimum number of changes allowed for text documents before the index is incrementally updated. If you do not specify the UPDATE MINIMUM keyword, the setting does not change.

Note that you can only change the UPDATE MINIMUM if you did not create the index using the RECREATE ON UPDATE option.

COMMITCOUNT FOR UPDATE count
For update processing, you can specify a commitcount. See “UPDATE INDEX” on page 132 for further information. This applies to both the UPDATE command and the UPDATE FREQUENCY specification, which schedules update processing.

Note that you can only change COMMITCOUNT if you did not create the index using the RECREATE ON UPDATE option.

CONNECT TO database-name
The name of the database that is target for this command. You can
ALTER INDEX command

omit this parameter, if DB2DBDFT is set and the user is running the command on the server. Note that the user ID must have the required DB2 authorizations.

USER userid USING password
Use a password and userid to connect to the database. If not specified, a connection is attempted from the current user ID without a password.

Usage
You cannot issue the command if one of the following commands is running on the index:
• ALTER INDEX
• CLEAR EVENTS
• ACTIVATE CACHE
• DROP INDEX
• UPDATE INDEX
• DEACTIVATE CACHE

Changes to the database
Change DB2 Net Search Extender catalog views.

Changes to the file system
Creation of NODE0000 subdirectories in the index, and work directories.
CLEAR EVENTS command

CLEAR EVENTS

This command deletes indexing events from an index’s event view. Use the event view for administration purposes. The name of the event view is found in the EVENTVIEWNAME column of the DB2EXT.TEXTINDEXES view.

Authorization

According to DB2 catalog views, the user ID in this command must have CONTROL privilege on the table for which the full-text index was created.

Command syntax

```plaintext
CLEAR-EVENTS-FOR-INDEX [connection-options] index-schema "" index-name FOR-TEXT
```

connection-options:

```plaintext
CONNECT TO database-name USER userid USING password
```

Command parameters

index-schema

The schema of the text index as specified in the CREATE INDEX command. If no schema is specified, the user ID of the DB2 connection is used.

index-name

The name of the text index as specified in the CREATE INDEX command.

CONNECT TO database-name

The name of the database that is target for this command. You can omit this parameter, if DB2DBDFT is set and the user is running the command on the server. Note that the user ID must have the required DB2 authorizations.

USER userid USING password

Use a password and userid to connect to the database. If not specified, a connection is attempted from the current user ID without a password.

Usage

When you schedule regular updates using the UPDATE FREQUENCY option in the CREATE or ALTER INDEX commands, regularly check the event table. Use
CLEAR EVENTS command

CLEAR EVENTS to clean up the event tables, after you have checked the reason for the event and removed the source of the error.

Try to ensure consistency between the contents of the text columns in the table and the index, especially when re-indexing documents.

You cannot issue the command if one of the following commands is running on the index:
- UPDATE INDEX
- ALTER INDEX
- ACTIVATE CACHE
- DEACTIVATE CACHE
- DROP INDEX
CREATE INDEX command

CREATE INDEX

This command creates a full-text index on a text column for use in DB2 Net Search Extender full-text queries.

Authorization

According to DB2 catalog views, the user ID in this command must have the CONTROL privilege on the table where the full-text index was created.

Command syntax

```
CREATE INDEX index-name FOR TEXT ON table-name
	([text-column-name] function-name(-text-column-name-)
	|attribute-list|	|text-default-information|
	|update-characteristics|	|storage-options|
	|cache-search-result-options|	|index-configuration-options|
	|connection-options|
```

attribute list:

```
|attribute-list|

|ATTRIBUTES ( SQL-column-expression AS attribute-name )|
```

text-default-information:

```
|text-default-information|

|CCSID ccsid |

|LANGUAGE language |

|FORMAT format|

|model-information|
```

model-information:
CREATE INDEX command

update-characteristics:

- UPDATE-FREQUENCY <update-frequency>
- RECREATE INDEX ON UPDATE

incremental-update-characteristics:

- UPDATE-MINIMUM <minchanges>
- REORGANIZE AUTOMATIC
- MANUAL
- COMMITCOUNT-FOR-UPDATE <count>

update-frequency:

- D (<0...6>)
- H (<0...23>)
- M (<0...59>)

storage-options:

- INDEX DIRECTORY <directory>
- WORK DIRECTORY <workdirectory>
- ADMINISTRATION-TABLES-IN <tablespace-name>

cache-search-results-options:

- CACHE TABLE <SQL-column-expression>
- PERSISTENT IN <directory>
- TEMPORARY IN <directory>
CREATE INDEX command

PCTFREE — percentage

MAXIMUM CACHE SIZE — memsize

INITIAL SEARCH RESULT ORDER — (SQL-order-by-list)

KEY COLUMNS FOR INDEX ON VIEW — (SQL-columnname-list)

index-configuration-options:

INDEX CONFIGURATION — (option-value)

connection-options:

CONNECT-TO database-name

USER userid USING password

Command parameters

index schema
   The schema of the text index. Use this as a DB2 schema name for the index-specific administration tables. If no schema is specified, the user ID of the DB2 connection is used. Note that the index schema must be a valid DB2 schema name.

index name
   The name of the index. Together with the index schema, this uniquely identifies a full-text index in a database. It also serves as the name of the index event table.

   See Appendix C, “Net Search Extender information catalogs”, on page 181 for details. Note that the index name must be a valid DB2 index name.

table schema
   The schema for which the table, nickname, or view the index is created. If no schema is specified, the user ID of the DB2 connection is used.
CREATE INDEX command

table name
The name of the text table, nickname, or view in the connected database that contains the column the full-text index is created for.

Note that when the table name does not refer to a DB2 base table, there are the following restrictions:
• The view only allows a stored procedure search. Therefore, you must specify the key columns for the index or views.
• For incremental index updates, a log table is created. You must manually fill this, if any changes occur to the data in the nickname table or view. With base tables this is done automatically, so the user must not touch the log table. For the layout of the log table, see Appendix C, “Net Search Extender information catalogs”, on page 181.
• The DB2 predicates CONTAINS, SCORE, and NUMBEROFMATCHES are only allowed for indexes on base tables or nicknames, but not on views.
• Indexes on views are only allowed if you specify cache-search-result options in the command. If you create indexes on views, you must specify the KEY COLUMNS FOR INDEX ON VIEW clause.

text-column-name
The name of the column containing the text used for creating the full-text index. The column must be one of the following types:
• CHAR (FOR BIT DATA)
• VARCHAR (FOR BIT DATA)
• LONG VARCHAR (FOR BIT DATA)
• CLOB
• DBCLOB
• BLOB
• GRAPHIC
• VARGRAPHIC
• LONG VARGRAPHIC
• DATALINK

If the column type is none of these, specify a transformation function using function-schema.function-name to convert the column type.

Note that, if you use a Data Link column, the referenced content is be fetched for indexing. This is via the protocol that is part of the Data Link value, for example, http. When using protocols other than "file" or "unc", ensure that you support these with servers that are part of the Data Link values. As proxy servers might be necessary to get the file content, the database administrator can specify them in the DB2EXT.PROXYINFORMATION table before index creation.
CREATE INDEX command

Note that several indexes on the same columns are allowed, but only with the following conditions:

The index is created on a view
Therefore, you cannot use the index in the CONTAINS, SCORE, or NUMBER OF DOCUMENTS search arguments.

The index is created on a table
If all the indexes are synchronized, they have identical properties on the same column in the following CREATE INDEX command details:
• Function name and schema
• ATTRIBUTES
• CCSID
• LANGUAGE
• FORMAT
• DOCUMENTMODEL
• INDEX CONFIGURATION

Therefore, it does not matter which index is chosen by the CONTAINS, SCORE, or NUMBER OF DOCUMENTS arguments.

function-schema.function-name
The schema and the name of a user-defined function used to access text documents that are in a column of an unsupported type. The function performs a column type conversion, using the one input parameter of an arbitrary column type. It returns the value of one of the Net Search Extender supported types.

ATTRIBUTES (SQL-column-expression AS Attribute-name, ...)
Ensures that the content of a column expression is indexed in addition to the text column. This content can also be searched by the ATTRIBUTE clause in a search statement. The SQL-column expressions have to be defined using unqualified column names of the table on which the index is created. The only data types allowed are double. Cast operators can be used in the column expressions, but implicit casting of DB2 is not possible. The attribute-names must follow the rules for attribute-names in document models and must be different from attribute names in the indexes model-definition file.

Determine the attribute names for expressions by using the following rules:
• If explicitly named by the SQL AS clause in the column expression, use the specified name. An example would be: ATTRIBUTES (C1+C2 AS myname)
• If a column of the specified table is used without AS, the name of the column is used. For example: CACHE TABLE (C1)
CREATE INDEX command

• If an expression is used without AS and which does not refer to a named column, CREATE INDEX reports an error.

For example: ATTRIBUTES (CAST(JULIAN_DAY(date) AS DOUBLE) as day, (price1+price2)/2 as avg_price)

Note that attributes without quotes are mapped to uppercase and must be specified in this way during search.

CCSID ccсид
The Coded Character Set Identifier is used when indexing text documents. The default value is from the DB2EXT.DBDEFAULTS view where DEFAULTNAME=’CCSID’.

LANGUAGE language
The language that determines the end-of-sentence and end-of-paragraph detection when indexing documents. For a list, see Appendix E, “Supported languages”, on page 195. The default value is from the DB2EXT.DBDEFAULTS view where DEFAULTNAME=’LANGUAGE’.

FORMAT format
The format of text documents in the column, for example, HTML. This information is necessary for indexing documents. See “Document formats and supported code pages” on page 23 for a list of document formats that are supported for structured documents.

For structured document formats, you can specify information in a document model file. If no document model is specified, the text of the document is indexed using a default document model. See “Document models” on page 159.

If the format keyword is not specified, the default value is from the DB2EXT.DBDEFAULTS view where the DEFAULTNAME=’FORMAT’.

DOCUMENTMODEL documentmodel-name IN modelfilepath
The modelfilepath specifies the location of a model file. This contains a model definition for the format in the FORMAT clause. It must be readable by the DB2 instance owner. A document model enables you to index and search specific sections of a document. You can define markup tags and section names in a document model. A document model is bound to a document format that supports HTML, XML, or GPP structures. You can only specify one document model in a model file.

As document models do not need to be referenced in search conditions, use all the section names in the model file instead. For details on document models, see Chapter 9, “Working with structured
CREATE INDEX command

Note that as the document model is only read during the CREATE INDEX command, any later changes are not recognized for this index.

**USING CCSID ccsid**
Specify a CCSID to interpret the contents of the model file. The default value is from the DB2EXT.DBDEFAULTS view where DEFAULTNAME='MODELCCSID'.

**UPDATE FREQUENCY**
The index update frequency determines when the update occurs. If changes to the user table are less than that specified by the UPDATE MINIMUM option, the index is not updated. If you do not specify the UPDATE FREQUENCY, the default NONE is used, so that no further index updates are made. This is useful when there are to be no further changes to a text column.

- **D.** The day(s) of the week when the index is updated: * (everyday) or 0..6 (0=Sunday)
- **H.** The hour(s) when the index is updated: * (every hour) or 0..23
- **M.** The minute(s) when the index is updated: 0..59
- **NONE.** No further index updates are made. The update must be started manually.

The default value is from the DB2EXT.DBDEFAULTS view where DEFAULTNAME='UPDATEFREQUENCY'.

**UPDATE MINIMUM minchanges**
The minimum number of changes allowed to text documents before the index is updated automatically by the UPDATE FREQUENCY. Positive integer values are allowed. The default value is taken from the DB2EXT.DBDEFAULTS view, where DEFAULTNAME='UPDATEMINIMUM'.

Note that this value is ignored in a DB2TEXT UPDATE command. This option cannot be used with the RECREATE INDEX ON UPDATE option, as the number of changes is not available without a log table and triggers for incremental update.

**REORGANIZE AUTOMATIC/MANUAL**
Updates performed using the update frequency will only recognize the index if REORGANIZE AUTOMATIC is specified. This step is completed automatically according to the value of select REORGSUGGESTED from DB2EXT.TEXTINDEXES after the update.

REORGANIZE MANUAL can only be performed with a manual UPDATE command, using the REORGANIZE option.
CREATE INDEX command

If the REORGANIZE clause is omitted, the default is taken from the DB2EXT.DBDEFAULTS view, where DEFAULTNAME='AUTOMATICREORG'.

For further information on the REORGANIZE option, see "UPDATE INDEX" on page 132.

COMMITCOUNT FOR UPDATE count

For incremental update processing, a commitcount can be specified, see "UPDATE INDEX" on page 132 for further information. If not specified, a default value is taken from the DB2EXT.DBDEFAULTS view, where DEFAULTNAME='COMMITCOUNT'.

The COMMITCOUNT FOR UPDATE value for the index can be found in DB2EXT.TEXTINDEXES.COMMITCOUNT. This can be changed for each index using the ALTER INDEX command. It also applies to the scheduled update processing according to the UPDATE FREQUENCY specification. A value of 0 means that the update is completed in one transaction, with values >0 specifying the number of documents to process in one transaction.

RECREATE INDEX ON UPDATE

This does not allow incremental index updates, but recreates the index when an update operation is performed (by command or scheduled update). See the Usage Notes on "UPDATE INDEX" on page 132 for additional information.

Note

No triggers are created on the user table and no log table is created.

INDEX DIRECTORY directory

The directory path in which the text index is to be stored. As the directory will contain index data, ensure that the directory has read/write and execute permissions for the DB2 instance owner user ID.

The default value is taken from the DB2EXT.DBDEFAULTS view, where DEFAULTNAME='INDEXDIRECTORY'. A subdirectory, NODE0000, is created under the directory to distinguish indexes on logical nodes of a server.

WORK DIRECTORY directory

A separate work directory may be specified optionally, that will be used to store temporary files during index search and administration operations. The directory must exist and have read/write and execute permissions for the DB2 instance owner user ID.
CREATE INDEX command

The default value is taken from the DB2EXT.DBDEFAULTS view, where DEFAULTNAME=’WORKDIRECTORY’. A subdirectory, NODE0000, is created under the directory to distinguish indexes on logical nodes of a server.

ADMINISTRATION TABLES IN tablespace-name
The name of the regular table space for administration tables created for the index. The table space must exist. If not specified, the tablespace of the user table is chosen, if the index is created on a base table.

In case of a nickname or a view, a default tablespace is chosen by DB2.

CACHE TABLE (SQL-column-expression-list)
A cached table is built in addition to the index, which consists of the specified column expressions. This cache is used to return the result set via a stored procedure search without joining full-text search results with a DB2 table. Note that a regular DB2 search using the full-text index with the CONTAINS function is always possible.

Define the SQL-column expressions using unqualified column names of the table the index is created on. The allowed SQL-column expression types are all built-in and user-defined distinct types. The column names in the result set are determined using the following rules:

• If explicitly named by the SQL AS clause in the column expression, the specified name is used. For example: CACHE TABLE (C1+C2 AS myname)
• If a column of the specified table is used without the AS clause, the name of the column is used. For example: CACHE TABLE (C1)
• If an expression is used without AS, and which does not refer to a named column, CREATE INDEX reports an error.
• No duplicate column names are allowed.

CLOB data types are not supported as cache data types. You need to cast these to VARCHARS.

Note
Note that if the column names of the result set are not disjunct, the CREATE INDEX command returns an error. Also note that the cached table is not implicitly activated after creation, for example search by stored procedure is not possible until DB2TEXT ACTIVATE CACHE is performed.
PERSISTENT IN directory
Specifies that the cache is also created persistent and could be activated shortly after a deactivation or a system reboot. The persistent cache is stored in the specified directory.

Note that if the directory is not specified, the default is taken from the DB2EXT.DBDEFAULTS view, where DEFAULTNAME='CACHEDIRECTORY'.

TEMPORARY
Specifies that the cache is not stored persistent. If neither PERSISTENT or TEMPORARY is specified, the default is taken from the DB2EXT.DBDEFAULTS view, where DEFAULTNAME='USEPERSISTENTCACHE'.

MAXIMUM CACHE SIZE memsize
Specifies the maximum size of the cached table to be built during DB2TEXT ACTIVATE CACHE. The memsize parameter must be specified in megabytes as a positive integer. There is no default value for memsize. If the integer is too small, the ACTIVATE CACHE command will fail. The actual cache size is calculated during the ACTIVATE CACHE command.

The limit for the maximum cache size on the different platforms is:
- Windows: 1024 MB (1 GB = 1073741824 bytes)
- AIX: 1536 MB (1.5 GB = 1610612736 bytes)
- Solaris, Linux, HP-UX: 2048 MB (2 GB = 2147483647 bytes)

PCTFREE percentage
Specifies the percentage of the cache to be held free for additional documents. The percentage must be an integer value lower than 100 and greater or equal to 0. If not specified, the default is taken from the DB2EXT.DBDEFAULTS view, where DEFAULTNAME='PCTFREE'.

See “ACTIVATE CACHE” on page 106 for details.

INITIAL SEARCH RESULT ORDER (SQL-order-by-list)
Specifies the order used for retrieving the user table contents during initial indexing. When using this option and skipping the dynamic ranking of full-text search results, the documents are returned in their indexing order, as stored in the cached result table.

For further information, see Chapter 16, “Stored procedure search function”, on page 155.
CREATE INDEX command

Note

The index order can not be ensured for the new or changed documents after incremental update. For example: INITIAL RESULT ORDER(length(column1) asc, column2+column3 desc)

KEY COLUMNS FOR INDEX ON VIEW (SQL-columnname-list)

If indexes on views are created, the KEY COLUMNS FOR INDEX ON VIEW clause must be specified, otherwise it MUST NOT be specified. The list of column names specifies the columns that UNIQUELY identify a row in the view.

As this uniqueness cannot be checked by DB2 as in case of primary keys, the user is responsible to ensure the equivalent uniqueness. The specified columns build part of the log table for the index.

INDEX CONFIGURATION (option-value), ...

These are the index configuration values. The default values are underlined.

<table>
<thead>
<tr>
<th>Option</th>
<th>Values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TreatNumbersAsWords</td>
<td>0 or 1</td>
<td>Interprets sequences of digits as separate words, even if they are adjacent to characters. For example, the 0 default means that tea42at5 is considered as one word.</td>
</tr>
<tr>
<td>IndexStopWords</td>
<td>0 or 1</td>
<td>Considers or ignores stopwords during indexing. Currently, the stopword list is an UCS-2 file &lt;language&gt;.tsw in directory &lt;instance&gt;/sqllib/db2ext/resources. Changes to this file have no effect after index creation. Also note that &lt;language&gt; is the LANGUAGE value from the CREATE INDEX command.</td>
</tr>
</tbody>
</table>

CONNECT TO database-name

The name of the database that is target for this command. You can omit this parameter, if DB2DBDFT is set and the user is running the command on the server. Note that the user ID must have the required DB2 authorizations.

USER userid USING password

Use a password and userid to connect to the database. If not specified, a connection is attempted from the current user ID without a password.

Changes to the database

- Change DB2 Net Search Extender catalog views.
CREATE INDEX command

- Create an index log table in specified table space. This is only if the RECREATE INDEX option is not specified plus according DB2 index.
- Create an index event table in specified table space.
- Deferred to first update: Creation of triggers on the user text table. This is only if the RECREATE INDEX option is not specified.

Changes to the shared memory
Deferred to ACTIVATE execution: If CACHE TABLE clause is used, a cache for the result table is built in shared memory.

Changes to the file system
- Subdirectories NODE0000 are created under index, work and cache directories.
- The directory <internal index name> is created under <indexdirectory>/NODE0000 where indexdirectory refers to the corresponding parameter of this command and NODE0000 is related to the node number in a distributed DB2 environment.

Usage
- Creation of a full-text index requires a primary key on the user table. In DB2 Net Search Extender Version 8.1, a multicolumn DB2 primary key can be used without type limitations. The number of primary key columns is limited to 14, the total length of all primary key columns is limited to 1024 - 14 = 1010 bytes.
- The total size of the SQL expressions for ATTRIBUTES, CACHE TABLE and INITIAL SEARCH RESULT ORDER must not exceed 24K bytes.
- Initial index updates are always done as one logical transaction, there is no commitcount in this case.

Note
That after creating the index, the length of primary key columns or the view key columns must not be changed by ALTER TABLE commands.

The synchronization between user table, full-text index and the cached result table is completed during the update index command. For further information, see the “UPDATE INDEX” on page 132.
DEACTIVATE CACHE command

DEACTIVATE CACHE

This command releases a cached table. A persistent cache is kept to be reused on the next ACTIVATE command. Until the next activation, search operations via the stored procedure are no longer possible on the deactivated cache.

Authorization

According to the DB2 catalog views, the userid in this command must have the CONTROL privilege on the table the full-text index was created for.

Command syntax

DEACTIVATE-CACHE-FOR-INDEX

index-schema

index-name

FOR-TEXT

|connection-options|

connection-options:

CONNECT-TO database-name

USER userid USING password

Command parameters

index-schema
The schema of the text index as specified in the CREATE INDEX command. If no schema is specified, the userid of the DB2 connection is used as schema name.

index-name
The name of the text index as specified in the CREATE INDEX command.

CONNECT TO database-name
The name of the database that is target for this command. You can omit this parameter, if DB2DBDFT is set and the user is running the command on the server. Note that the user ID must have the required DB2 authorizations.

USER userid USING password
Use a password and userid to connect to the database. If not specified, a connection is attempted from the current user ID without a password.

Usage

Note that this command could not be issued when one of the following commands is running on the index:

- ACTIVATE CACHE
DEACTIVATE CACHE command

- DEACTIVATE CACHE
- UPDATE INDEX
- ALTER INDEX
- DROP INDEX
- CLEAR EVENTS

**Note**
After deactivation of a persistent cache, the cache is made inaccessible for search by the stored procedure. However, this can be used for fast ACTIVATE, unless an update was performed in the meantime.

In this case, the persistent cache is automatically recreated from scratch using the ACTIVATE CACHE command.
**DROP INDEX command**

**DROP INDEX**

This command drops a full-text index for a text column. If the cache for the index is activated, it is deleted using this command.

**Authorization**

According to DB2 catalog views, the userid in this command must have the CONTROL privilege on the table the full-text index was created for. Alternatively, the user can be the database administrator (DBADM).

Alternatively, the database administrator (DBADM), can drop the index as they must be able to disable the database using the FORCE option.

**Command syntax**

```
DROP INDEX index-name FOR TEXT
```

```
|connection-options|
```

**connection-options:**

```
CONNECT TO database-name
USER userid USING password
```

**Command parameters**

**index schema**

The schema of the text index as specified in the CREATE INDEX command. If no schema is specified, the userid of the DB2 connection is used as the schema name.

**index-name**

The name of the index as specified in the CREATE INDEX command. With the index schema, it uniquely identifies the full-text index in a database.

**CONNECT TO database-name**

The name of the database that is target for this command. You can omit this parameter, if DB2DBDFT is set and the user is running the command on the server. Note that the user ID must have the required DB2 authorizations.

**USER userid USING password**

Use a password and userid to connect to the database. If not specified, a connection is attempted from the current user ID without a password.
DROP INDEX command

Usage

The index is deleted, irrespective of the activation status of its cached table. For additional information, see “ACTIVATE CACHE” on page 106 for more information.

Note that the command could not be issued when one of the following commands is running on the index:

- UPDATE INDEX
- CLEAR EVENTS
- ALTER INDEX
- ACTIVATE CACHE
- DEACTIVATE CACHE
- DROP INDEX

Note

Indexes must be manually dropped before or after the user table in DB2 is dropped. If not, the results are not correctly cleaned up.

Changes to the database

- Change DB2 Net Search Extender catalog views
- Drop the DB2 index
- Drop the index log/event tables
- Delete triggers on the user text table

Changes to the shared memory

The cached table is deleted.

Changes to the file subsystem

- The directory <internal index name> is deleted in the index and the work directories of the dropped index
- Delete a persistent cache for the index
DB2EXTTH command

DB2EXTTH (Utility)

This independent utility compiles a thesaurus definition file. After running the thesaurus compiler, the THESAURUS-related features of the search argument syntax can be used.

Authorization

None. This command is not necessarily restricted for the table owner, but makes only sense in the context of querying.

Command syntax

```
```

Command parameters

- **-f definition-file-name**
  The name of the file containing the thesaurus definition. The file name must contain either the absolute path or the relative path to the file. The file name is restricted to 8+3 characters, the extension being optional.

  The thesaurus dictionary is generated in the same directory as the definition file and has the same name. The only difference is that the dictionary has the following extensions: wdf, wdv, grf, grv, MEY, ROS, NEY, SOS, and Ikn, where n is a digit. Note that if existing thesaurus files have the same name, they are overwritten.

- **-ccsid code page**
  The code page in which the thesaurus definition file is written. See Appendix L, “Thesaurus supported CCSIDs”, on page 237 for a list of the supported code pages for a thesaurus.

- **-quiet**
  Output information is not displayed.

- **-copyright**
  Returns the internal build number of the product. Use this number when reporting problems.

- **-h, -H, or -?**
  Displays help information.

Usage

Use this command to compile a thesaurus definition file into a binary thesaurus definition format.
Note
The format is the same as in DB2 Text Information Extender Version 7.2 with the following changes:

- The new relationships BROADER and NARROWER are equivalent to the HIGHER_THAN and LOWER_THAN relationships previously used in Text Information Extender. To refer to these older search relationships, the new relationships must be used.

- Also note that the thesaurus dictionary files must be stored in <os-dependent>/sqllib/db2ext/thes to be usable during search, unless the thesaurus is fully qualified in the query.

For additional information, see Chapter 10, “Using a thesaurus to expand search terms”, on page 85.
UPDATE INDEX command

**UPDATE INDEX**

This command immediately starts the indexing process by bringing the index up to date to reflect the current contents of the text columns with which the index is associated.

While the update is being performed, search using the CONTAINS predicate is possible. For an index with an activated cached result table, search by stored procedure is also possible during update. However, columns in the cached table may show new values, even though the changed text is not yet committed to the full-text index.

Using the RECREATE INDEX ON UPDATE option will clear the index before recreation. Until completion of the update, empty results will be returned.

**Authorization**

According to the DB2 catalog views, the user ID in this command must have the CONTROL privilege on the table the full-text index was created for.

**Command syntax**

```
UPDATE INDEX index-schema-"." index-name FOR TEXT [REORGANIZE]

COMMITCOUNT count [USING UPDATE MINIMUM] [connection-options]
```

connection-options:

```
CONNECT TO database-name [USER userid USING password]
```

**Command parameters**

- **index-schema**
  - The schema of the text index. This is specified in the CREATE INDEX command. If no schema is specified, the user ID of the DB2 connection is used.

- **index-name**
  - The name of the text index. This is specified in the CREATE INDEX command.

- **REORGANIZE**
  - If a text column is frequently updated, then subsequent updates to the index can become inefficient. To make the update process efficient again, reorganize the index. Use the DB2EXT.TEXTINDEXES view to determine if an index needs to be reorganized.
UPDATE INDEX command

Use the REORGANIZE AUTOMATIC option of the CREATE INDEX command to avoid manually checking and reorganizing the index.

**Note**
The reorganization process takes place after a regular update.

**USING UPDATE MINIMUM**
Uses the UPDATE MINIMUM settings from the CREATE INDEX command and starts an incremental update only if the specified number of changes was reached. The default is to unconditionally start the update.

See "CREATE INDEX" on page 114 for additional information.

**COMMITCOUNT count**
An INTEGER value \( \geq 0 \) displays the number of documents processed in one transaction by the search engine and by DB2 for incremental index updates.

However, for initial updates, such as the first update after the CREATE INDEX command, or any update with RECREATE INDEX ON UPDATE option, there is only one logical transaction which ignores COMMITCOUNT. This may be changed using the ALTER INDEX command.

**CONNECT TO database-name**
The name of the database that is target for this command. You can omit this parameter, if DB2DBDFT is set and the user is running the command on the server. Note that the user ID must have the required DB2 authorizations.

**USER userid USING password**
Use a password and userid to connect to the database. If not specified, a connection is attempted from the current user ID without a password.

**Usage**
This command runs synchronously. The duration depends on the number of files to be indexed and the number of documents already indexed. The status of the update can be seen through a view that is created for each index. The name of this view can be retrieved from DB2EXT.TEXTINDEXES in column EVENTVIEWNAME. For further information, refer to Appendix C, “Net Search Extender information catalogs”, on page 181.

To determine if an update is still running and how many documents have been committed to the index, use the DB2EXT.TEXTINDEXES
UPDATE INDEX command

(NUMBERDOCS) view. Information on starting, committing changes, and finishing update processing can also be retrieved from the event view associated with the index.

You cannot issue the command if one of the following commands is running on the index:

- CLEAR EVENTS
- ALTER INDEX
- DROP INDEX
- ACTIVATE CACHE
- DEACTIVATE CACHE
- UPDATE INDEX

After updating an index with a deactivated persistent cached result table, the persistent cache is deleted, such that the next ACTIVATE CACHE command recreates it based on the database content.

If the user interrupts this command, all processes involved in the update function will stop. If a commitcount was used in an incremental update, some updates may be visible in the index, while others may require a new update command.

Note

As the command works in two separate phases for index creation on all partitions and initial index updates, issue a db2text drop index command to ensure that the index is not partly available. If this command is not issued, the next update, which can be triggered by update command or the update frequency option, would perform a complete re-indexing to ensure a consistent state.

Changes to the database

- Inserts to the event table
- Delete from the index log table
HELP command

HELP

This displays the list of available DB2TEXT commands, or the syntax of an individual DB2TEXT command.

Authorization

None required.

Command syntax

```
HELP command

? reasoncode
```

Command parameters

HELP or ?

Provides help for the specified command or reason code.

command

The first keywords that identify a DB2TEXT command:

- ENABLE
- DISABLE
- CREATE
- DROP
- ALTER
- UPDATE
- CLEAR
- START
- STOP
- CONTROL
- ACTIVATE
- DEACTIVATE

reasoncode

Reason code from a DB2 Net Search Extender command.

Usage

If more than the first keyword is specified, the rest are ignored and the syntax of the identified command is displayed.

If no 'command' parameter is specified after '?' or 'HELP' (or no parameter at all), DB2TEXT lists all available DB2TEXT command parameters. For example:

```
DB2TEXT ? CTE0110
```
COPYRIGHT command

COPYRIGHT

Provides Net Search Extender product and copyright information.

Authorization
None required.

Command syntax

COPYRIGHT

Command parameters

COPYRIGHT / LEVEL
Provides the version copyright statement, version number, and build information for the product.
Chapter 14. Syntax of search arguments

A search argument is the condition that you specify when searching for terms in text documents. It consists of search parameters and one or more search terms.

Examples of search arguments are given in “Specifying SQL search arguments” on page 75 and in a file called search. See the “Additional search syntax examples” on page 80.

The SQL scalar search functions that use search arguments are:

CONTAINS
This function uses a search argument to search for text in a particular text document. It returns the INTEGER value 1 if the document contains the text, or any relation specified in the search argument. Otherwise, it returns 0.

NUMBEROFMATCHES
This function uses a search argument to search in text documents and returns an INTEGER value indicating how many matches resulted per document.

SCORE
This function uses a search argument to search in text documents. It returns a value for each document found, indicating how well the found document is described by the search argument.

Note
You use the same syntax in the search arguments of the stored procedure search and the SQL Table-Valued Function.
Syntax of search arguments

Search argument

Search argument syntax

RESULT LIMIT $number$
STOP SEARCH AFTER $number$
DOCUMENT
DOCUMENTS

boolean-search-expression
freetext-argument

Boolean-search-expression:

search-term
boolean-search-expression operator-or search-term

search-term:

search-factor
operator-and search-factor
operator-minus positive-search-factor

search-factor:

NOT positive-search-factor

Positive-search-factor:

SEARCH
SECTION ( "section-name" )
SECTIONS WEIGHT $number$

search-primary

Search-primary:

text-literal
context-condition
thesaurus-invocation

Operator-and:
Syntax of search arguments

Operator-or:

Operator-accum:

Operator-minus:

Context-condition:

Context-argument:

Text-literal-list:

Text-literal:

Context-unit:

Text-literal:
Syntax of search arguments

thesaurus-invocation:

\[ \text{THESAURUS} \text{"thesaurus-name" EXPAND} \]

Attribute-factor:

\[ \text{ATTRIBUTE} \text{"attribute-name" BETWEEN valueFrom AND valueTo} \]

freetext-argument:

\[ \text{IS-ABOUT} \text{"word-or-phrase" ESCAPE"escape-character"} \]

Examples

Examples are given in “Specifying SQL search arguments” on page 75

Search parameters

RESULT LIMIT number
A keyword specifying the maximum number of results to be returned by the full-text search.

The RESULT LIMIT should be used together with the SCORE function to ensure that the returned results are scored and only the best results are processed.

STOP SEARCH AFTER number DOCUMENTS(S)
A keyword specifying the search threshold. The search is stopped when the number of documents is reached during the search and an intermediate result is returned. A lower value will increase the search performance, but may lead to fewer results and omit documents with a potentially high rank.

Note that there is no default value and the number value must be a positive integer.

boolean-search-expression
The search-terms and search-factors can be combined using boolean
operators NOT, AND, OR, ACCUM, and MINUS according to the syntax diagrams. The operators have the following precedence order (with the strongest first): NOT > MINUS = ACCUM = AND > OR. This can be seen in the following example:

"Pilot" MINUS "passenger" & "vehicle" | "transport" & "public"

is evaluated as:

\[ (("Pilot" \text{ MINUS} "passenger") \text{ &} ("vehicle")) | ("transport" \text{ &} "public") \]

The operator ACCUM evaluates to true, if one of the boolean arguments evaluates to true (which is comparable to the OR operator). The rank value is computed by accumulating rank values from both operands. The ACCUM operator has the same binding (precedence) as AND. The operator MINUS evaluates to true, if the left operand evaluates to true. The rank value is computed by taking the rank value for the left operand and subtracting a penalty, if the right operand evaluates to true.

**search-primary**
A search-primary consisting of a thesaurus-invocation evaluates to true, if any of the expanded text-literals is found in the (specified section of the document). A search-primary, consisting of a text-literal-list evaluates to true, if any of the text-literals is found in the (specified section of the document).

**SECTION(S) section-name**
A keyword specifying one or more sections in a structured document that the search is to be restricted to. The section name must be specified in a model file specified at index creation time, see "CREATE INDEX" on page 114.

Section names are case sensitive. Ensure that the section name in the model file and query is identical.

This model describes the structure of documents that contain identifiable sections, so the content of these sections can be individually searched. Section names cannot be masked using masking characters. The positive-search-factor using the SECTION clause evaluates to true, if the search primary is found in one of the sections.

**context-argument IN SAME context-unit AS context-argument AND context-argument ...**
This condition lets you search for a combination of text-literals occurring in the same paragraph or same sentence. Context arguments are always equivalent to text-literal-lists, and thesaurus expansion may be used to expand a text-literal to such a list.
Syntax of search arguments

The condition evaluates to true, if there is a context-unit (paragraph respectively sentence) in the document, which contains at least one of the text-literals of each expanded context-argument. This can be seen in the following example:

\[ ("a","b") \text{ IN SAME PARAGRAPH AS } ("c","d") \]
\[ \text{AND THESAURUS } "t1" \text{ EXPAND SYNONYM TERM OF } "e". \]

Assuming e1, e2 as synonyms of e, the following paragraphs would match:

".. a c e ..", ".. a c e1..", "a c e2..",
".. a d e ..", ".. a d e1..", "a d e2..",
".. b c e ..", ".. b c e1..", "b c e2..",
".. b d e ..", ".. b d e1..", "b d e2..".

PRECISE FORM OF
A keyword that causes the word (or each word in the phrase)
following PRECISE FORM OF to be searched for exactly as typed.
This form of search is case-sensitive; that is, the use of upper- and lowercase letters is significant. For example, if you search for mouse, you do not find "Mouse".

STEMMED FORM OF
A keyword that causes the word (or each word in the phrase)
following STEMMED FORM OF to be reduced to its word stem before the search is carried out. This form of search is not case-sensitive. For example, if you search for mouse, you find “Mouse”.

The way in which words are reduced to their stem form is language-dependent. Currently, only English is supported and the word must follow regular inflection endings.

FUZZY FORM OF
A keyword for making a “fuzzy” search, which is a search for terms that have a similar spelling to the search term. This is particularly useful when searching in documents that were created by an Optical Character Recognition (OCR) program. Such documents often include misspelled words. For example, the word economy could be recognized by an OCR program as economy. Note that the first three characters must match and that fuzzy search cannot be used if a word in the search atom contains a masking character.

match level
An integer from 1 to 100 specifying the degree of similarity, where 100 is more similar than 1. 100 specifies an "exact match", and 60 is already considered a very "fuzzy value". The fuzzier the match level is, the longer the lapsed search time, since more documents qualify for the search. The default match level is 70.
Syntax of search arguments

**WEIGHT number**

Associates a text-literal with a weight value to change the default score. The allowed weight values are integers between 0 (the lowest score weighting) and 1000 (the highest); the default value is 100.

**word-or-phrase**

A word or phrase to be searched for. The characters that can be used within a word are language-dependent. It is also language-dependent whether words need to be separated by separator characters. For English and most other languages, each word in a phrase must be separated by a blank character.

To search for a character string that contains double quotation marks, type the double quotation marks twice. For example, to search for the text "wildcard" character, use:

"**wildcard**" character"

Note that in the example, it is only possible to search for one set of quotation marks. You cannot search for two quotation marks in a sequence. There is also a maximum length of 128 bytes for each word or phrase.

**Masking characters**

A word can contain the following masking characters:

- `_` (underscore)
  Represents any single character.

- `%` (percent)
  Represents any number of arbitrary characters. If a word consists of a single %, then it represents an optional word of any length. A word cannot be composed exclusively of masking characters, except when a single % is used to represent an optional word. If you use a masking character, you cannot use THESAURUS. Masking characters cannot follow a non-alphanumeric character.

**ESCAPE escape-character**

A character that identifies the next character as one to be searched for and not as one to be used as a masking character. For example, if an escape-character is $, then $%, $_, and $%$ represent %, _ and $ respectively. Any % and _ characters not preceded by $ represent masking characters.

**THESAURUS thesaurus-name**

A keyword used to specify the name of the thesaurus to be used to expand a text-literal. The thesaurus name is the file name (without its extension) of a thesaurus that has been compiled using the thesaurus.
Syntax of search arguments

compiler. It must be located in <os-dependent>/sql1ib/db2ext/thes. Alternatively, the path can be specified preceding the file name.

EXPAND relation
Specifies which relation is used to expand the text-literal using the thesaurus. The thesaurus has predefined relations described in the DB2EXTTH command. These are referred to using the following keywords:

- SYNONYM, a symmetrical relationship expressing equivalence.
- RELATED, a symmetrical relationship expressing association.
- BROADER, a directed hierarchical relationship that can be followed by specified depth levels.
- NARROWER, a directed hierarchical relationship that can be followed by specified depth levels.

For user-defined relations, use RELATION(number), that corresponds to the relation definition in DB2TEXTTH.

TERM OF text-literal
The text-literal, to which other search terms are to be added from the thesaurus.

count LEVELS
A keyword used to specify the number of levels (the depth) of terms in the thesaurus that are to be used to expand the search term for the given relation. If you do not specify this keyword, a count of 1 is assumed. The value of depth must be a positive integer value.

ATTRIBUTE Attribute-name
Searches for documents having attributes matching the specified condition. The attribute-name refers to the name of an attribute expression in the CREATE INDEX command, or to an attribute definition in the document model file.

The attribute-factor is allowed for attributes of type double only. The precision of the value is guaranteed for 15 digits. Numbers of 16 characters and above are rounded. Usage of masking characters is not allowed in attribute-name, valueFrom and, valueTo. For an explanation, see the following:

BETWEEN valueFrom AND valueTo
A BETWEEN attribute factor evaluates to true if the value of the attribute is greater than (not equal to) valueFrom and lower than (not equal to) valueTo.

>valueFrom
A “>” attribute factor evaluates to true if the value of the attribute is greater than (not equal to) valueFrom.
Syntax of search arguments

<valueTo
    A "<" attribute factor evaluates to true if the value of the
    attribute is lower than (not equal to) valueTo.

If the attribute name in the CREATE INDEX command is specified with
quotes, or is defined in a model file, the specified attribute name must
match exactly. Whereas, if no quotes are specified in the CREATE INDEX
command, the attribute name must be in uppercase.

IS ABOUT language word-or-phrase
    An option that lets you specify a free-text search argument. It should
    be used to get a different kind of score algorithm as it checks the
    positioning of the terms within the documents. The closer together the
    terms used in the word-or-phrase are, the more terms are included in
    the document and the higher the score value returned.

    The values allowed for language are described in Appendix E,
    “Supported languages”, on page 195 and are only relevant for the
    Thai language. If not specified, the language en_US is used as default.
    The language is used only for tokenization of the word-or-phrase.

    Note that IS ABOUT is useful only if the score values are requested
    and the search results are ordered by score values.
Syntax of search arguments
Chapter 15. SQL scalar search function and the SQL table-valued function

DB2 Net Search Extender provides SQL scalar search functions and a SQL table-valued function for searching text documents stored in DB2 Universal Database.

This chapter describes the following SQL search functions.

**A summary of the search functions**

<table>
<thead>
<tr>
<th>Search function</th>
<th>Purpose</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONTAINS</td>
<td>Searches for text in a particular document.</td>
<td>148</td>
</tr>
<tr>
<td>NUMBEROFMATCHES</td>
<td>Searches and returns the number of matches found.</td>
<td>149</td>
</tr>
<tr>
<td>SCORE</td>
<td>Searches and returns the score value of a found text document.</td>
<td>150</td>
</tr>
<tr>
<td>DB2EXT:TEXTSEARCH</td>
<td>The SQL table-valued function returns a table of found primary keys, a number of matches, and/or score values.</td>
<td>151</td>
</tr>
</tbody>
</table>

See [Chapter 8, “Searching”, on page 73](#) for examples of using the SQL scalar search functions and the SQL table-valued function.
CONTAINS scalar function

The CONTAINS scalar function searches for text in a text document indexed by Net Search Extender. It returns the INTEGER value 1 if the document contains the text, or any relation specified in the search argument. Otherwise, it returns 0.

**Function syntax**

```
CONTAINS (column-name, search-argument)
```

**Function parameters**

- **column name**
  The name of a column. The column must have an associated text index. You can create text indexes by using the administration command `DB2TEXT CREATE INDEX`.

- **search-argument**
  A string of type VARCHAR containing the terms to be searched. See Chapter 14, “Syntax of search arguments”, on page 137.

**Note**

You cannot use the CONTAINS query on a text index created on a view.
The NUMBEROFMATCHES scalar function searches in text documents and returns an INTEGER value indicating how many matches resulted per document.

**Function syntax**

```
NUMBEROFMATCHES(column-name, search-argument)
```

**Function parameters**

- **column name**
  The name of a column. The column must have an associated text index. You can create text indexes by using the administration command `DB2TEXT CREATE INDEX`.

- **search-argument**
  A string of type VARCHAR containing the terms to be searched. See Chapter 14, “Syntax of search arguments”, on page 137.

**Note**

You cannot use the NUMBEROFMATCHES query on a text index created on a view.
SCORE scalar function

The SCORE scalar function searches in text documents and returns a score value for each document found, indicating how well the found document is described by the search argument.

SCORE returns a DOUBLE value. As the search term appears more frequently in the document, the score of the document increases.

Function syntax

```
SCORE( column-name, search-argument )
```

Function parameters

- **column name**
  - The name of a column. The column must have an associated text index. You can create text indexes by using the administration command `DB2TEXT CREATE INDEX`.

- **search-argument**
  - A string of type VARCHAR containing the terms to be searched. See Chapter 14, “Syntax of search arguments”, on page 137.

**Note**

You cannot use the SCORE query on a text index created on a view.
In addition to the stored procedure search and the SQL scalar search functions, Net Search Extender provides an SQL table-valued function which looks very similar to the stored procedure.

**Function syntax**

```sql
db2ext.textSearch
(
    Query VARCHAR(4096),
    IndexSchema VARCHAR(128),
    IndexName VARCHAR(128),
    resultFirstRow INTEGER,
    ResultNumberRows INTEGER,
    PrimKeyBinding <supported types>,// same type as above
)
return table
(
    primKey <supported types>,// same type as above
    numberOfmatches INTEGER,
    score DOUBLE,
    totalNbResults INTEGER
)
```

**Function parameters**

- **query**
  
  See Chapter 14, “Syntax of search arguments”, on page 137 for additional information.

- **indexSchema, indexName**
  
  Identifies the index to search. For more information, see “CREATE INDEX” on page 114.

- **resultFirstRow**
  
  The result list of the query is returned in parts. This parameter describes which row of the query result list is the first one to be entered into the result table of the table-valued function. The value should be >= 0.

  Note that the number 0 identifies the first row in the query result list.

- **resultNumberRows**
  
  This parameter describes how many rows of the query result list are entered into the result table of the table-valued function, and where 0 means that all the results need to be returned.

  Note that this is different from the result limit query parameter that determines the maximum size of the query result list.
SQL table-valued function

**primaryKeyBinding**

The type of this parameter determines the type of the primaryKey Output parameter. If the text index has been created for a base table with a primary key of type `<type1>`, then primaryKeyBinding must also be of type `<type1>`.

Additionally, the parameter determines the scope of the text search. If primaryKeyBinding is set to NULL ("CAST(NULL as `<type1>`)", the scope of the search will be all the documents stored in the index. Alternatively, you can restrict the search to documents primaryKeyBinding is bound to.

For example, if primaryKeyBinding is set to CAST(5 as BIGINT), you restrict the search to the single document with the BIGINT primary key value of "5".

Note that only single column primary keys of the following types are supported: SMALLINT, INTEGER, BIGINT, REAL, DOUBLE, VARCHAR FOR BIT DATA, DATE, TIME, and TIMESTAMP.

**Function parameters**

The following output parameters are stored in a temporary table which needs to be joined to your normal table if further results are requested. Note that the NUMBEROFMATCHES, SCORE, and TOTALNUMBEROFRESULTS are only calculated if they are requested in your select statement.

**primKey**
The primary key of the found document.

**numberofmatches**
NUMBEROFMATCHES is an INTEGER value indicating how many matches resulted for each document.

**score**
Score returns a DOUBLE value. As the search term increases in frequency in the document, the document score increases.

**totalNumberOfResults**
The query result list denotes how many results were found. Note that each row has the same value.

Also note that when you use the STOP SEARCH AFTER, or the RESULT LIMIT together with the SCORE syntax in a query, this number is no longer reliable.

**Usage**

With the SQL table-valued function you are able to search on views in the same way you do with the stored procedure search. The exception being that no shared memory is needed, so the index does not need to be activated.
This function is primarily for those users who have used an SQL query within the stored procedure search. However, the restriction is that only a single column primary key on base tables is supported.

The following example shows how you can work on a multi-column primary key table:

```sql
SELECT s.id FROM db2ext.sample s, table (db2ext.textSearch(
    "characteristics",
    'DB2EXT',
    'COMMANDS',
    1,
    20,
    cast(NULL as INTEGER))) t
WHERE s.id = t.primkey
```

In this example, you must first create a view on this table with a single unique key and then create the index on this view.
SQL table-valued function
Chapter 16. Stored procedure search function

Net Search Extender provides a stored procedure search for returning predefined result tables. The result table is specified in the cache table section during create index. Use the stored procedure search when you need to return a small number of results in a specific order.

An example would be an Internet application where the first 20 rows are returned, but the rest of the results can also be returned in increments of 20 rows.
Stored procedure search function

DB2EXT.TEXTSEARCH (for stored procedure search)

Function syntax

```sql
db2ext.TextSearch(
    IN query VARCHAR(4096),
    IN indexSchema VARCHAR(128),
    IN indexName VARCHAR(128),
    IN resultFirstRow INTEGER,
    IN resultNumberRows INTEGER,
    IN scoringFlag INTEGER,
    IN searchTermCountsFlag INTEGER,
    OUT searchTermCounts VARCHAR(4096),
    OUT totalNumberOfResults INTEGER )
```

Function parameters

The following are input parameters.

**Query**  See Chapter 14, "Syntax of search arguments", on page 137 for further information.

**indexSchema, indexName**

To identify the index to search. Refer to "CREATE INDEX" on page 114.

**resultFirstRow**

The query result list is returned in parts. The parameter describes which row of the query result list is the first one to be put into the result set of the stored procedure. The first row in the query result list is identified by the number 0.

**resultNumberRows**

This parameter describes how many rows of the query result list are put into the result set of the stored procedure.

This is not to be confused with the "result limit" expression in the query, which determines the maximum flag size of the query result list.

The value should be >= 0. Where 0 means that all the results need to be returned.

**Note**

If a larger result set is requested, ensure that a temporary user tablespace is available. If there is none available, then create a tablespace. The following example creates a tablespace on a UNIX platform:

```sql
db2 "create user temporary tablespace temptst managed by system using ('/work/tempts.ts')"
```
Stored procedure search function

**scoringFlag**
0 means there is no scoring and 1 means there is scoring. If scoring is requested, an add row with the score value is returned.

**searchTermCountsFlag**
This controls the searchTermCounts processing. If searchTermCountsFlag is 0, the searchTermCounts is not calculated.

**Function parameters**
The following are output parameters.

**searchTermCounts**
The number of occurrences of each search term query in the index. These counts are returned as a blank separated list in the order of search terms in the query.

See the **searchTermCountsFlag** for information.

**totalNumberOfResults**
The total number of results found in the query result list.

Also note that when you use the STOP SEARCH AFTER, or the RESULT LIMIT together with the SCORE syntax in a query, this number is no longer reliable.

**Usage**
The columns in the result set returned by the stored procedure are given by the RESULT COLUMNS option of the DB2TEXT CREATE INDEX command. If scoringFlag=1, then a column of type double is added. This column contains the SCORE value.

Use the following options to increase the performance of a second query with the same string as the first query. Note that this must be in a different cursor window with no totalNumberOfResults required:

- If you do not require SCORE, add the following syntax: STOP SEARCH AFTER x DOCUMENTS, where x is the resultFirstRow + resultNumberOfRows.
- If you require SCORE, add the following syntax: STOP SEARCH AFTER y DOCUMENTS, where y is equal to the totalNumberOfResults in the first query.
Stored procedure search function
Chapter 17. Structured document support

Structured documents consist of document models and document file definitions.

**Document models**

A document model primarily controls what parts of a document’s structure need to be indexed and how they are indexed. Its purpose is to:

- Identify text fields that should be distinguished in the source document
- Determine the type of such a text field
- Assign a field name to the text field

When the document model identifies text as belonging to a text field, the text is considered to be part of the textual content of the document, and terms are extracted and stored in the index.

The elements of a document model vary depending on the parser used for that document format:

- For HTML format, a document model uses the HTML tag names to define which tags should be indexed, and how to handle meta-tag information.
- For XML format, there is no predefined set of tags, so a document model must first define which tags are of interest. XML elements of the same name can also be distinguished based on what other elements they are embedded in.
- For GPP (general purpose parser) format, the document model interacts even more deeply with the parser, because it has to determine the boundaries of the text fields. Here the field definition must specify strings for detecting the boundaries of fields.

See the relevant "Defining a Document Model" section for information.

For information on the document model syntax in the form of a Document Type Definition (DTD), and text field limitations, see Appendix G, “Document model reference”, on page 217.

**Default document models**

For HTML and XML documents, Net Search Extender provides default document models that are used if you do not define a document model. For structured plain text documents, you must provide and specify a document model.
Document model

If you use one of the default document models:

- All fields are indexed, and no special information, such as meta information, is extracted.
  - For HTML, each field is assigned the name of the corresponding tag.
  - For XML, the generated field name is the complete tag path, for example `play/role/name`.
- No numeric attribute is indexed (as no numeric attribute is defined in the default document model).

Table 6. Behavior of the default document models for the supported document formats

<table>
<thead>
<tr>
<th>Document type</th>
<th>Behavior of the default document model</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTML</td>
<td>Accepts these as text fields: <code>&lt;a&gt;</code> <code>&lt;address&gt;</code> <code>&lt;au&gt;</code> <code>&lt;author&gt;</code> <code>&lt;h1&gt;</code> <code>&lt;h2&gt;</code> <code>&lt;h3&gt;</code> <code>&lt;h4&gt;</code> <code>&lt;h5&gt;</code> <code>&lt;h6&gt;</code> <code>&lt;title&gt;</code>. Field name is the tag name, for example &quot;address&quot;.</td>
</tr>
<tr>
<td>XML</td>
<td>Accepts all tags as text fields. Field name is the tag path name in Xpath notation, for example &quot;/play/title&quot;.</td>
</tr>
<tr>
<td>Structured plain text</td>
<td>No default document model.</td>
</tr>
</tbody>
</table>

For each type of document, a document model is defined. As the models are all different, an example and explanation is provided for each one.

Note

Although the default document models do correctly process documents, for better indexing and search you should define your own document models.

With the default document model, the text of a document is fully indexed regardless of whether or not it is part of a text field. This means that unrestricted text searches include a search of that text.

Defining a document model for structured plain-text documents

Here is an example of a general-purpose (GPP) structured plain-text document:

```
[head]Handling structured documents[/head]
[year]2002[/year]
[abstract]This document describes the concept of structured documents and the use of document models to...[/abstract]
```
Here is an example of a GPP document model:

```xml
<?xml version="1.0"?>
<GPPModel>

<GPPFieldDefinition
name="Head"
start="[head]"
end="[/head]"
exclude="YES" />

<GPPFieldDefinition
name="Abstract"
start="[abstract]"
end="[/abstract]"
exclude="NO" />

<GPPAttributeDefinition
name="year"
start="[year]"
end="[/year]"
type="NUMBER" />

</GPPModel>
```

The first line, `<xml version="1.0"?>` specifies that the document model is written using XML tags. Note that this model is not written for XML format documents.

Each field is defined within a `GPPFieldDefinition` or `GPPAttributeDefinition` tag, which contain element parameters.

All the definitions must be contained within the `<GPPModel>` tag.

**Element parameters**

These are the parameters of the document model elements:

- **name** You assign a name to the text field or document attribute for each definition. The names enable you to restrict a search query to the content of a specific text field or document attribute. Using the above examples, you could search for documents containing the word structure in the text field named Abstract.

- **start** A boundary string in code page UTF-8 that marks the beginning of the text field or document attribute. There are no rules for specifying boundary strings; they can be any arbitrary UTF-8 string. Here are some examples: start="introduction:", start="note!", start="$$...".

  Nonprintable characters and the special XML characters "<" and "&" must be specified using the standard XML escape character ("\&lt;" for "<", and "\&amp;" for ").

- **end** Optional. A boundary string in code page UTF-8 that marks the end
Document model

of the text field or document attribute. If you do not specify an end
tag, the next found start tag is assumed to be the end of the field. If
no subsequent start tag is found, the field extends to the end of the
document, and no further fields are identified.

**type**

The type of document attribute must always be "NUMBER". The
parameter does not apply to field definitions.

**exclude**

YES or NO. A parameter that determines whether the text in a field
should be excluded and not indexed. This parameter does not apply
to attribute definitions. In the example, the field definition "head"
would be excluded, but definition "abstract" included.

**Restrictions:**

- There must not be two field definitions or attribute definitions having the
  same start tag. However, a field definition and attribute definition may
  have the same start tag and end tags.
- A start tag must not be a proper prefix of another. For example, you cannot
  have a start tag "author" and a start tag "authority".
- Start tags and end tags must not be empty strings.

For information on the Document Type Definitions, see “DTD for document
models” on page 217

For additional restrictions, see “Limitations for text fields and document
attributes” on page 220

**What happens when a GPP document is indexed**

The general-purpose parser scans the document looking for one of the start
boundary strings. When it finds a start boundary string, it parses the
subsequent field until it finds the corresponding end boundary string.

The content of the field is then indexed according to the definition term, that
is, as a text field or document attribute. If the text field and document
attribute have the same start and end boundary strings, the content of the
field is indexed as both a text field and a document attribute.

No nesting of fields is allowed; if a new start boundary string is found in a
field before the end boundary string has been reached, the new start
boundary string is interpreted as normal text.

If no corresponding end boundary string is found, the field is assumed to
extend to the end of the document, and an appropriate reason code is
reported.
Document model

If no end boundary string is specified in the document model, the new start boundary string signals the end of the previous field.

Defining a document model for HTML documents

The HTML parser converts the text to code page UTF-8. It performs HTML tag recognition and classifies them into tag classes:

- Tagged information to be ignored, such as font information
- Tags that provide positional information, such as <p>; for new paragraph
- Tags that provide structural information, such as <Title>

It recognizes all character entity references defined in HTML 4, like "&auml;" (ä) and resolves them to the corresponding code points in UTF-8.

It recognizes meta tags and parses the meta tag text.

Here is an example of an HTML document:

```html
<HTML>
<HEAD>
<META NAME="year" CONTENT="2002">
<TITLE> The Firm </TITLE>
</HEAD>
<BODY>
<H1>Synopsis</H1>

<H1>Prologue</H1>
</BODY>
</HTML>
```

Here is an example of an HTML document model:

```xml
<?xml version="1.0"?>
<HTMLModel>

<HTMLFieldDefinition
name="subtitle"
tag="title"
exclude="YES" />

<HTMLFieldDefinition
name="header1"
tag="h1"
exclude="YES" />

<HTMLAttributeDefinition
name="year"
tag="meta" />

- This is the start of text field
- This is the end of the text field
- This is the start of the document attribute
```

Chapter 17. Structured document support 163
Document model

```xml
<meta-qualifier="year">
  type="NUMBER" />
</HTMLModel>
```

This is the end of the document attribute

The first line, `<?xml version="1.0"?>`, specifies that the document model is written using XML tags. Note that this model is not written for XML format documents.

Each field is defined within a `HTMLFieldDefinition` or `HTMLAttributeDefinition` tag, which contain element parameters.

All the text field definitions must be contained within the `<HTMLModel>` tag.

**Element parameters**

These are the parameters of the document model elements:

- **name**
  
  You assign a name to the text field or document attribute for each definition. The names enable you to restrict a search query to the content of a specific text field or document attribute. Using the above examples, you could search for documents containing the word "firm" in the text field named "subtitle".

- **tag**
  
  Identifies an element whose start and (implied) end tags mark the text field or document attribute. The text that is inside an element of that name makes up the content of the defined field.

  The case of the tag is disregarded.

  Using the above examples, the text following any `H1` tag is indexed as being part of the field "header1". In which case, "synopsis" and "prologue" would be indexed.

- **meta-qualifier**
  
  This tag has to be used with the `tag` element. By specifying `tag="meta"`, the value of the content that matches the meta-qualifier is extracted.

  In the HTML document example, the meta tag has the following elements:

  ```xml
  <META NAME="Author" CONTENT="J. Grisham">
  ```

  In the document model example, meta-qualifier="author". Therefore, the content "J. Grisham" is indexed as the value of the string attribute "author".

- **type**
  
  The type of document attribute must be "NUMBER". The parameter does not apply to field definitions.
Document model

**exclude** YES or NO. A parameter that determines whether the text in field definitions should be excluded and not indexed. This parameter does not apply to attribute definitions. In the example, the field definition “header1” would be excluded, but definition “subtitle” included.

All other text of a document is indexed, but not as part of any field.

For information on the Document Type Definitions, see “DTD for document models” on page 217

For restrictions, see “Limitations for text fields and document attributes” on page 220

---

**Defining a document model for XML documents**

Net Search Extender does not attempt to detect the code page of an XML document. The CCSID specified during `CREATE INDEX`, or if not specified, the DB2 code page.

Here is an example of an XML document:

```xml
<?xml version="1.0"?>
<purchaseOrder orderDate="2001-01-20">
    <shipAddress countryCode="US">
        <name>Alice Smith</name>
        <street>123 Maple Street</street>
        <city>Mill Hill</city>
        <state>CA</state>
        <zip>90999</zip>
    </shipAddress>
    <item partNo="123" quantity="1">
        <name>S&B Lawnmower Type ABC-x</name>
        <price>239.90</price>
        <shipDate>2001-01-25</shipDate>
    </item>
    <item partNo="987" quantity="1">
        <name>Multifunction Rake ZYX</name>
        <price>69.90</price>
        <shipDate>2001-01-24</shipDate>
    </item>
</purchaseOrder>
```

Here is an example of an XML document model:

```xml
<?xml version="1.0"?>
<XMLModel>
    <XMLFieldDefinition name="addresses" locator="/purchaseOrder/shipAddress" exclude="no" />
</XMLModel>
```
Document model

The first line,  
<XMLFieldDefinition
    name="customerName"
    locator="/shipAddress/name"
    exclude="yes"/>

<XMLAttributeDefinition
    name="partNumber"
    type="NUMBER"
    locator="/purchaseOrder//item/@partNo" />

</XMLModel>

The first line, `<xml version="1.0">`, specifies that the model is written using XML. Each field is defined within a XMLFieldDefinition or XMLAttributeDefinition tag, which contains element parameters.

Note that all the text field definitions must be contained within the `<XMLModel>` tag. For restrictions, see “Limitations for text fields and document attributes” on page 220.

Element parameters

These are the parameters of the document model elements:

name  You assign a name to the text field or document attribute for each definition. These names enable you to restrict a search query to the content of a specific text field or document attribute.

You can use one of the following variables in a name. The variable is replaced by a string generated from the matching element in the source document.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(NAME)</td>
<td>The actual qualified name (QName) of the XML element that matched the XPath.</td>
</tr>
<tr>
<td>$(LOCALNAME)</td>
<td>The actual local name (without prefix) of the XML element that matched the XPath.</td>
</tr>
<tr>
<td>$(PATH)</td>
<td>The actual absolute path as a sequence of slashes and tags of the XML element that matched the XPath.</td>
</tr>
</tbody>
</table>

type  The type of document attribute must be "NUMBER". The parameter does not apply to field definitions.

locator  Expressions in the XPath language that select the parts of the source documents to be used as search fields.

These locators are from the example. For further information, see the syntax in “The semantics of locator (XPath) expressions” on page 218.
Document model

**purchaseOrder | salesOrder**

All purchaseOrder elements and salesOrder elements

**shipAddress**

All shipAddress elements

**\***

All elements (this is the abbreviation of child::* – see the syntax for further information)

**name/item**

All item elements that have a name parent

**purchaseOrder/item**

All item elements that have a purchaseOrder ancestor

**/**

The root node

**comment()**

All comment nodes

**processing-instruction()**

All processing instructions

**attribute::* (or @*)**

All attribute nodes

**NCName**

An XML name not containing colons

**QName**

An NCName that can be preceded by NCName: (an NCName followed by a colon), like this: NCName:NCName

A literal is a string enclosed either in single or double quotes. For an exact definition of terminal tokens see the XML recommendations.

The XPath locators are similar to XML Stylesheet Language Transformation (XSLT) patterns. They comprise exactly the subset of XSLT patterns that do not contain any predicates nor the functions ‘id’ and ‘key’ nor the node tests ‘text()’ and ‘node()’.

**ignore**

YES or NO. Use the parameter to make exceptions to the locator.

Sometimes you may want to specify a general locator, such as *, to match the nodes you want to index. But you may also specify that some nodes matching a more specific locator should not be indexed.

To formulate this, include a field definition with the more specific locator for the nodes to be ignored during indexing. You then give this locator a higher priority than the one with the general locator, and specify ignore="yes". This indicates to the indexer that it must not generate field information for the matching nodes.

Note that when such an ignored node is embedded in a field-generating node, the content of the ignored node gets indexed, because it also belongs to the contents of the field-generating node.
priority
A real number between -1 and +1 that specifies the priority to be given to a definition found by a particular locator.

If you do not specify a priority, the default priorities are used:

- Multiple alternatives separated by | are treated as a set of definitions, one for each alternative.
- Locators that match by a single name; that is, locators of either of the following forms have default priority 0:
  - ChildOrAttributeAxisSpecifier QName
  - ChildOrAttributeAxisSpecifier processing-instruction(Literal)
- Locators of the form ChildOrAttributeAxisSpecifier NCName:* have default priority -0.25.
- Other locators of the form ChildOrAttributeAxisSpecifier NodeTest have default priority -0.5.
- Any other locator has default priority 0.5.

Note that the more specific the locator is, the higher the default priority. For example, the unspecific locator * gives a low priority to the found definition, whereas a name is a more specific locator and gives a higher priority.

Also note that when a node is matched by more than one locator, you can determine which of the definitions are chosen by assigning priorities to them. The definition with the highest priority is chosen. If two definitions have the same priority, the latest is chosen.

This conflict resolution is the same as that used in XML Stylesheet Language Transformation (XSLT).

exclude
YES or NO. A parameter that determines whether the text in field definitions should be excluded and are not indexed. This parameter does not apply to attribute definitions.

In the example, the field definition "customerName" would be excluded, but definition "addresses" included.
Document model

What happens when an XML document is indexed

The following table shows what is put into the index.

Table 7. Entries in the text index

<table>
<thead>
<tr>
<th>Field name</th>
<th>Indexed text</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>addresses</td>
<td>123 Maple Street Mill Hill CA 90999</td>
<td>[1]</td>
</tr>
<tr>
<td>customerName</td>
<td>Alice Smith</td>
<td>[2]</td>
</tr>
<tr>
<td>partNumber</td>
<td>123 987</td>
<td>[3]</td>
</tr>
</tbody>
</table>

Note that in [1], where the text of the shipAddress element is indexed under field name addresses, the name element (Alice Smith) is not indexed. This is because the name element is itself in the document model, and indexed under field name customerName [2]. This means that, although embedded elements are allowed in the document model, they are not indexed as part of the embedding text field; they are indexed separately.

The content of fields is determined by the following rules:

- For a field whose locator matches a comment or a processing instruction, the field content is the actual comment text or processing instruction text.
- For a field that matches an XML element or the root node, the field content consists of any text from any embedded element except for elements that are matched by other fields.

The document must contain well-formed XML, but it is not necessary for a DTD to be specified in the XML document. No DTD validation or entity resolution is carried out; Net Search Extender only matches the XML document against the document model.

For information on the Document Type Definitions, see “DTD for document models” on page 217.

For restrictions, see “Limitations for text fields and document attributes” on page 220.
Chapter 18. Thesaurus support

Here is the syntax of each definition group:

Syntax of a thesaurus definition

- :WORDS
- :SYNONYM
- :RELATED
  (number)
- :HIGHER_THAN
  (number)
- :LOWER_THAN
  (number)

Note that \n is not part of the syntax, but represents the end of a line in the thesaurus definition file.

You can insert comment lines in a thesaurus definition file like this:

```
# my comment text
```

:WORDS

A keyword that begins a group of related words.

:SYNONYM,
:RELATED [(number)],
:HIGHER_THAN [(number)],
:LOWER_THAN [(number)]

A relation name.

Relation names consist of a relation type and a number. If the number is omitted, zero is assumed, which is the system-provided relation name. :SYNONYM is always the system-provided relation name.

Relation names that begin with a colon, such as :SYNONYM, precede a list of words that are related to each other by the same relation. For example:
Thesaurus support

:WORDS
:SYNONYM
  air steward
  cabin staff member
  flight attendant

member-term
A term to be included in the thesaurus dictionary.
- Maximum length is 64 bytes (42 bytes for code page UTF-8).
- Single-byte characters and double-byte characters of the same letter are regarded as the same.
- Uppercase and lowercase characters are not distinguished.
- A term can contain a blank character.
- The single-byte character period "." or colon ":" cannot be used.

This parameter can be useful if you do not want a thesaurus lookup to include words that have a weak relation to the looked up term. Strength is a numerical value from 1 to 100. The default value is 100.

.SYNONYM_OF,
.RELATED_TO [(number)]
A relation name. Relation names consist of a relation type and a number. If the number is omitted, zero is assumed, which is the system-provided relation name. The relation name .SYNONYM is always the system-provided relation name.

Relation names that begin with a period, such as .SYNONYM_OF, define the relation between one word and another. For example:

:WORDS
  air steward
  .SYNONYM_OF cabin staff member
  .SYNONYM_OF flight attendant

The optional number identifies a user-defined relation. This must be a unique number from the whole thesaurus definition file (currently 1 to 128). For example: RELATED_TO(42).

If you want to use symbolic names for thesaurus relations in your application instead of the relation name and number, your application must handle the name-to-number mapping. For example, if you define the relation opposite_of as RELATED_TO(1), your application must map this name to the internal relation name RELATED_TO(1).

associated-term
Each associated term must be preceded by the relation name. The associated term is related to each member term with respect to the specified relation. If all member terms are related to each other, this can be specified using a member relation.
- Maximum length is 64 bytes (42 bytes for code page UTF-8).
Thesaurus support

- Single-byte characters and double-byte characters of the same letter are regarded as the same.
- Uppercase and lowercase characters are not distinguished.
- A term can contain a blank character.
- The single-byte character period "." or colon ":" cannot be used.

Here is an example of an associated term:

```
:WORDS:SYNONYM
  reject
  decline
    RELATED_TO(1) accept
```
Part 3. Appendixes
Appendix A. Migration

DB2 Net Search Extender Version 8.1 has been extensively altered to include the search interface and functions of DB2 Text Information Extender Version 7.2. Accordingly, there are two options available:

• Moving from Net Search Extender Version 7.2 to Net Search Extender Version 8.1

If you have been using Net Search Extender Version 7.2 and do not require all the new features, simply continue using the old interfaces by installing the Net Search Extender Version 7.2 compatibility interfaces available on the CD-ROM, or by download. See the `release.txt` for further information. To use the new Net Search Extender functionality, you need to change your administration scripts to correspond to the altered administration syntax and change your search syntax to the new search interfaces.

Note that there is no automatic way to migrate from Net Search Extender Version 7.2 to Net Search Extender Version 8.1.

Also note that these compatibility interfaces are deprecated and will not be available in future releases.

• Moving from Text Information Extender Version 7.2 to Net Search Extender Version 8.1

Install Net Search Extender Version 8.1 and migrate your DB2 instance from Version 7.2 to Version 8.1 using `db2iupdt` for both UNIX and Windows. Then migrate your instance using `db2extiupdt`, followed by your databases using `db2extiupdt <dbname>`. This program reads all `db2ext` tables and generates a list of required indexes with the new syntax: `<dbname>migto81.lst`. All the old indexes are dropped and recreated according to the `<dbname>migto81.lst`. Note that this process can take a considerable amount of time.

For HP-UX or Linux, this step is not necessary as Text Information Extender is not available on these platforms.

**Note**

Note that for this migration tool, the " character is not supported for any database object on Windows.

Also note that DB2 Text Information Extender Version 7.2 is no longer available.
Migration

Note
For the latest migration information, check the release.txt file on the CD-ROM and DB2 Net Search Extender Web site.
Appendix B. Using large amounts of memory

Using the cache for a stored procedure search requires a large amount of memory and different memory requirements for the following platforms:
- AIX
- Windows
- The Solaris Operating Environment

**AIX**

Configuring the system limits:
- Check the system limits using the command `ulimit -a`
- If there are values other than "unlimited", use the following steps:
  - Log on as root.
  - Back up the file `/etc/security/limits` and then edit the file to raise the hard limits.
  - Set all values to the "unlimited" (value -1) for the DB2 instance owner used.

Configuring the shared memory limits:
- On AIX, there is no need to configure the shared memory limits.

Configuring the swap space:
- Obtain the system RAM size using the command `lsattr -E -l sys0`
- Obtain the size of the swap space using the `lspv -a` command.
- Set the swap space size to at least 1.5 - 2 times of either the RAM amount of your system, or use the `MAXIMUM CACHE SIZE` parameter you provide in the `CREATE INDEX` command. Use the SMIT utility to select the larger number.
  
  Note that the maximum cache size limit on AIX is 1536 MB (1.5 GB = 1610612736 bytes).

**Windows 32-bit**

Adjusting the size of the paging file:
- Set the Windows virtual-memory paging file size to at least 1.5 - 2 times of either the RAM amount of your system, or use the `MAXIMUM CACHE SIZE` parameter you provide in the `CREATE INDEX` command. Select the larger number. See the Windows documentation for information on changing the size of the paging file.
Using large amounts of memory

Note that the maximum cache size limit on Windows is 1000 MB (1 GB = 1073741824 bytes).

The Solaris Operating Environment

Configuring the system limits:
- Check the system limits using the command: ulimit -a
- Then use the following steps:
  - Log on as root.
  - Back up the file /etc/system and then edit the file to raise the hard limits.
  - Add or check that the following lines are set to at least the minimum values shown:
    - rlim_fd_cur -> Default 64, recommended >= 1024
    - rlim_fd_cur_max -> Default 1024, recommended >= 4096

Configuring the shared memory limits:
- Check current settings using command sysdef -i
- Edit the file /etc/system to set the shared memory size limit using: set shmsys:shminfo_shmmax=0xffffffff
  You may also have to increase the following parameter values:
  - set shmsys:shminfo_shmmni=512
  - set shmsys:shminfo_shmseg=128 and then reboot the system.

Configuring the swap space:
- Obtain the system RAM size using the command /usr/sbin/prtconf
- Obtain the size of the swap space using the swap -l command.
- Set the swap space size to at least 1.5 - 2 times of either the RAM amount in your system, or use the MAXIMUM CACHE SIZE parameter you provide in the CREATE INDEX command. Select the larger number.
  Refer to the Solaris system documentation for information on how to add swap space.
  Note that the maximum cache size limit on Solaris is 2000 MB (2 GB = 2147483647 bytes).
Appendix C. Net Search Extender information catalogs

DB2 Net Search Extender stores important information about defaults, configurations, text indexes, and formats in catalog tables. To view this information, you can query some views on tables.

The following views and tables reflect the current configuration of your system:

- **Database level information views:**
  - `db2ext.dbdefaults`
  - `db2ext.proxyinformation` table

- **Index level information views:**
  - `db2ext.textindexes`
  - `db2ext.textindexformats`
  - `db2ext.indexconfiguration`

Note that for compatibility reasons, DB2 Text Information Extender views are still available. These are `db2ext.textcolumns`, `db2ext.models`, and `db2ext.formats`.

- **Table views for a text index:**
  - Event view
  - Log table view

**Views for database-level information**

The view `db2ext.dbdefaults` displays all the default values for the database.

The defaults on the database level cannot be changed and are available as attribute-value pairs in this view:

```
db2ext.dbdefaults
```

```
db2 select DEFAULTNAME, DEFAULTVALUE from DB2EXT.DBDEFAULTS
```

*Table 8. db2ext.dbdefaults view*

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Default Value</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>CCSID</td>
<td>CCSID of database</td>
<td>Default CCSID for documents. This is applied if no CCSID is specified in the CREATE INDEX command.</td>
</tr>
</tbody>
</table>
Table 8. db2ext.dbdefaults view (continued)

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Default Value</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>FORMAT</td>
<td>TEXT</td>
<td>Document default format. This is applied if no format is specified in the CREATE INDEX command.</td>
</tr>
<tr>
<td>INDEXDIRECTORY</td>
<td>See the path name under Notes</td>
<td>Directory for full-text index files. This is applied if no index directory is specified in the CREATE INDEX command. The path name is: $DB2EXT_INSTOWNERHOMEDIR/sqllib/db2ext/indexes</td>
</tr>
<tr>
<td>LANGUAGE</td>
<td>EN_US</td>
<td>The document language.</td>
</tr>
<tr>
<td>MODELCCSID</td>
<td>CCSID of database</td>
<td>CCSID of document model files.</td>
</tr>
<tr>
<td>UPDATECOMMITCOUNT</td>
<td>0</td>
<td>The number of changes processed in one transaction during an update.</td>
</tr>
<tr>
<td>UPDATEFREQUENCY</td>
<td>NONE</td>
<td>When to check for updates in new indexes.</td>
</tr>
<tr>
<td>UPDATEMINIMUM</td>
<td>1</td>
<td>The minimum number of changes before update is executed.</td>
</tr>
<tr>
<td>WORKDIRECTORY</td>
<td>See the path name under Notes</td>
<td>Directory for index temporary files. The path name is: &lt;os_dependent&gt;/sqllib/db2ext/indexes</td>
</tr>
<tr>
<td>CACHEDIRECTORY</td>
<td>See the path name under Notes</td>
<td>Default directory for PERSISTENT CACHE option of the CREATE INDEX command. The path name is: &lt;os_dependent&gt;/sqllib/db2ext/memory</td>
</tr>
<tr>
<td>PCTFREE</td>
<td>50</td>
<td>The percentage of the cache left free for future inserts.</td>
</tr>
<tr>
<td>USERPERSISTENTCACHE</td>
<td>1</td>
<td>Use the persistent cache.</td>
</tr>
<tr>
<td>AUTOMATICREORG</td>
<td>1</td>
<td>The REORGANIZE option in the CREATE INDEX command. This means automatic reorganization.</td>
</tr>
<tr>
<td>TREATNUMBERSASWORDS</td>
<td>0</td>
<td>Do not interpret sequences as separate words, even if they are adjacent characters. For example, the 0 default means that tea42at5 is considered as one word.</td>
</tr>
<tr>
<td>INDEXSTOPWORDS</td>
<td>1</td>
<td>Ignore stopwords during indexing.</td>
</tr>
</tbody>
</table>
If you are using datalinks and want to access files using a proxy server, you must specify proxy information in the `db2ext.proxyinformation` table.

Table 9. `db2ext.proxyinformation` view

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROXYHOST</td>
<td>VARCHAR(254)</td>
<td>The host name of the proxy server.</td>
</tr>
<tr>
<td>PROXYPORT</td>
<td>VARCHAR(6)</td>
<td>The used port for the proxy server.</td>
</tr>
<tr>
<td>PROXYTYPE</td>
<td>VARCHAR(10)</td>
<td>The type of the proxy server (either PROXY or SOCKS).</td>
</tr>
<tr>
<td>PROXYTIMEOUT</td>
<td>INTEGER</td>
<td>The timeout in seconds.</td>
</tr>
</tbody>
</table>

You can insert a maximum of one row. If you have a proxy server on port 123, you can insert a row with the following SQL statement:

db2 insert into db2ext.proxyinformation values ('proxy1', '123','PROXY', 10)

---

Views for database-level information

**db2ext.proxyinformation table**

You can query the view to obtain information about the indexes. This is an example using the index schema:

---

Views for index-level information

You can query information at an index-level using the following DB2 Net Search Extender views:

- `db2ext.textindexes`
- `db2ext.textindexformats`
- `db2ext.indexconfiguration`
- `<index eventview name schema>.<index eventview name>`

For backward compatibility reasons, DB2 Text Information Extender views `db2ext.textcolumns`, `db2ext.formats`, and `db2ext.models` are still supported, but deprecated.

Note that in the `db2ext.textcolumns` view the `OPERATION`, `OPERATIONBEGIN`, and `OPERATIONEND` columns are no longer supported.

**db2ext.textindexes view**

Each database enabled for DB2 Net Search Extender contains a `db2ext.textindexes` view. This contains information on settings, statistics, and defaults for the created text indexes in this database.

When you create a text index, new entries are created in `db2ext.textindexes`. When you drop the text indexes, these entries are deleted.

You can query the view to obtain information about the indexes. This is an example using the index schema:
Views for index-level information

```
db2 "select COLNAME from DB2EXT.TEXTINDEXES where INDSHEMA='myschema' and INDNAME='myindex'"
```

Note, however, that you cannot modify the view using normal SQL data manipulation commands, or explicitly create or drop the catalog view. Additional contents of the view are found in the following table.

*Table 10. db2ext.textindexes view*

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>INDSHEMA</td>
<td>VARCHAR(128)</td>
<td>Schema name of the text index.</td>
</tr>
<tr>
<td>INDNAME</td>
<td>VARCHAR(128)</td>
<td>Name of the text index.</td>
</tr>
<tr>
<td>TABSCHEMA</td>
<td>VARCHAR(128)</td>
<td>The table name of the schema for base tables, nicknames, and views.</td>
</tr>
<tr>
<td>TABNAME</td>
<td>VARCHAR(128)</td>
<td>Alias name the index was created on.</td>
</tr>
<tr>
<td>COLNAME</td>
<td>VARCHAR(128)</td>
<td>Column the index was created on.</td>
</tr>
<tr>
<td>CCSID</td>
<td>INTEGER</td>
<td>Document CCSID for this index.</td>
</tr>
<tr>
<td>LANGUAGE</td>
<td>VARCHAR(5)</td>
<td>Document language for this index.</td>
</tr>
<tr>
<td>FUNCTIONSCHEMA</td>
<td>VARCHAR(128)</td>
<td>Schema of the column mapping function.</td>
</tr>
<tr>
<td>FUNCTIONNAME</td>
<td>VARCHAR(18)</td>
<td>Name of the column mapping function.</td>
</tr>
<tr>
<td>INDEXDIRECTORY</td>
<td>VARCHAR(256)</td>
<td>Directory for full-text index files.</td>
</tr>
<tr>
<td>WORKDIRECTORY</td>
<td>VARCHAR(256)</td>
<td>Directory for index temporary files.</td>
</tr>
<tr>
<td>CACHEDIRECTORY</td>
<td>VARCHAR(256)</td>
<td>Directory for persistent cache (if persistentcache=1).</td>
</tr>
<tr>
<td>UPDATEFREQUENCY</td>
<td>VARCHAR(300)</td>
<td>Trigger criterion for applying automatic updates to this index.</td>
</tr>
<tr>
<td>UPDATEMINIMUM</td>
<td>INTEGER</td>
<td>Minimum number of documents that must be changed before an update is performed.</td>
</tr>
<tr>
<td>EVENTVIEWSCHEMA</td>
<td>VARCHAR(128)</td>
<td>Schema of the event view created for this index.</td>
</tr>
<tr>
<td>EVENTVIEWNAME</td>
<td>VARCHAR(128)</td>
<td>Name of the event view created for this index.</td>
</tr>
<tr>
<td>LOGVIEWSCHEMA</td>
<td>VARCHAR(128)</td>
<td>Schema of the log view created for an index.</td>
</tr>
<tr>
<td>LOGVIEWNAME</td>
<td>VARCHAR(128)</td>
<td>Name of the log view created for an index (important for incremental update on views).</td>
</tr>
<tr>
<td>COMMITCOUNT</td>
<td>INTEGER</td>
<td>Default for commitcount update.</td>
</tr>
</tbody>
</table>
Views for index-level information

Table 10. db2ext.textindexes view (continued)

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>NUMBER_DOCS</td>
<td>INTEGER</td>
<td>Total number of documents currently in the index.</td>
</tr>
<tr>
<td>REORG_SUGGESTED</td>
<td>INTEGER</td>
<td>Indicates if performance can be improved by running UPDATE INDEX REORGANIZE.</td>
</tr>
<tr>
<td>REORGAUTOMATIC</td>
<td>INTEGER</td>
<td>1, if the index gets automatically reorganized during the update operation.</td>
</tr>
<tr>
<td>RECREATEONUPDATE</td>
<td>INTEGER</td>
<td>1, if the index gets automatically reorganized during the update operation.</td>
</tr>
<tr>
<td>CREATIONTIME</td>
<td>TIMESTAMP</td>
<td>Time of index creation.</td>
</tr>
<tr>
<td>UPDATETIME</td>
<td>TIMESTAMP</td>
<td>Time of last update. If UPDATETIME is equal to CREATIONTIME, then no update has been processed.</td>
</tr>
<tr>
<td>PERSISTENTCACHE</td>
<td>INTEGER</td>
<td>1, if persistent cache is used.</td>
</tr>
<tr>
<td>MAXIMUMCACHESIZE</td>
<td>INTEGER</td>
<td>Maximum size of cache.</td>
</tr>
<tr>
<td>PCTFREE</td>
<td>INTEGER</td>
<td>Percentage of cache left free for future inserts.</td>
</tr>
<tr>
<td>CACHETABLE</td>
<td>VARCHAR(32000)</td>
<td>Column expression list for the CACHE TABLE.</td>
</tr>
<tr>
<td>RESULTORDER</td>
<td>VARCHAR(32000)</td>
<td>SQL-order-by for INITIAL RESULT ORDER.</td>
</tr>
<tr>
<td>ATTRIBUTES</td>
<td>VARCHAR(32000)</td>
<td>Column expression list for ATTRIBUTES.</td>
</tr>
<tr>
<td>VIEWKEYCOLUMNS</td>
<td>VARCHAR(32000)</td>
<td>Key columns for index on view.</td>
</tr>
</tbody>
</table>

db2ext.indexconfiguration view

Index configuration parameters are available in the db2ext.indexconfiguration view. The view is available through normal SQL query facilities. This is an example using the index name:

db2 "select VALUE from DB2EXT.INDEXCONFIGURATION where INDSCHEMA='myschema' and INDNAME='myindex' and PARAMETER = 'INDEXSTOPWORDS'"
Views for index-level information

Additional contents of the view are found in the following table.

**Table 11. db2ext.indexconfiguration view**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>INDSCHEMA</td>
<td>VARCHAR(128)</td>
<td>Schema name of the index.</td>
</tr>
<tr>
<td>INDNAME</td>
<td>VARCHAR(128)</td>
<td>Name of the index.</td>
</tr>
<tr>
<td>PARAMETER VALUE</td>
<td>VARCHAR(30)</td>
<td>TREATNUMBERASWORDS 0 or 1 INDEXSTOPWORDS 0 or 1</td>
</tr>
<tr>
<td>FORMAT</td>
<td>VARCHAR(512)</td>
<td>The model is bound to this format.</td>
</tr>
</tbody>
</table>

For further information, see the CONFIGURATION option of the CREATE INDEX command.

**db2ext.textindexformats view**

Format and model information for indexes is available in the db2ext.textindexformats view. This is an example using the index name:

```
db2 "select FORMAT from DB2EXT.TEXTINDEXFORMATS where INDSCHEMA='myschema' and INDNAME='myindex'"
```

Additional contents of the view are found in the following table.

**Table 12. db2ext.textindexformats view**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>INDSCHEMA</td>
<td>VARCHAR(128)</td>
<td>Schema name for the index (used as prefix for indexname and schemaname in the log table).</td>
</tr>
<tr>
<td>INDNAME</td>
<td>VARCHAR(128)</td>
<td>Index name specified in CREATE INDEX command.</td>
</tr>
<tr>
<td>FORMAT</td>
<td>VARCHAR(30)</td>
<td>The model is bound to this format.</td>
</tr>
<tr>
<td>MODELNAME</td>
<td>VARCHAR(30)</td>
<td>The name of a document model.</td>
</tr>
<tr>
<td>MODELFILE</td>
<td>VARCHAR(256)</td>
<td>File containing the model definition.</td>
</tr>
<tr>
<td>MODELCCSID</td>
<td>INTEGER</td>
<td>CCSID of MODELFILE.</td>
</tr>
<tr>
<td>DEFAULT</td>
<td>INTEGER</td>
<td>Currently 1, as multiple formats in an index are not currently supported.</td>
</tr>
</tbody>
</table>
Views for index-level information

Table views for a text index

You can query information at an index level using these DB2 Net Search Extender views:

- Event view
- Log table view

Event view

This view allows you to get information on indexing status and error events, and when problems occur during indexing, for example, a document not being found. These index update events are then written to the index's event table.

The schema and name are stored in the `db2ext.textindexes` view. To get the name of the event view, use the following example:

```sql
db2 "select EVENTVIEWSchema, EVENTVIEWNAME from DB2EXT.TEXTINDEXES
   where INDSchema = 'myschema' and INDNAME = 'myindex"
```

The event view for an index consists of the following columns.

### Table 13. The event view

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPERATION</td>
<td>INTEGER</td>
<td>The operation on the user table is be reflected in full-text index (insert = 0/ update = 1/ delete = 2).</td>
</tr>
<tr>
<td>TIME</td>
<td>TIMESTAMP</td>
<td>Timestamp of event entry creation.</td>
</tr>
<tr>
<td>REASON/SEVERITY</td>
<td>INTEGER</td>
<td>Reason code.</td>
</tr>
<tr>
<td>MESSAGE</td>
<td>VARCHAR(1024)</td>
<td>Additional text information.</td>
</tr>
<tr>
<td>KEY1, ... KEY14</td>
<td>VARCHAR</td>
<td>First primary key column of user table to the last primary key column (a maximum of 14).</td>
</tr>
</tbody>
</table>

The events can be cleared by the `DB2TEXT CLEAR EVENTS` command, see “CLEAR EVENTS” on page 112 for further information.

**Note**

Informational events, such as starting, committing, and finishing update processing are also available in this view.

In this case, Key1, ... Key14 and OPERATION all have NULL values.
Views for index-level information

In the case of indexes on views, the PK01, ..., PK14 columns relate to the columns specified in the KEY COLUMNS clause of the CREATE INDEX command.

Log table view

The purpose of the log table is to store the change operations on the user table or view that require synchronization with the external full-text index.

For indexes created on regular tables or nickname tables, there are triggers created on the user table to feed change information into the log table. The update command reads the entries and deletes them after successful synchronization.

However, in the case of indexes on views, triggers cannot fill the log table. As you can update the view, the user is responsible for this task.

Table 14. The log table view

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPERATION</td>
<td>INTEGER</td>
<td>The type of change on the user table that requires index synchronization: (0 = insert, 1 = update, 2 = delete).</td>
</tr>
<tr>
<td>TIME</td>
<td>TIMESTAMP</td>
<td>The timestamp for the creation of a row in this table.</td>
</tr>
<tr>
<td>PK01 ... PKNm</td>
<td>Same as user table</td>
<td>In case of errors, the column where the problem occurred. They are a copy of the primary key columns of the user table or the equivalent key columns in case of an index on a view.</td>
</tr>
</tbody>
</table>

The user who creates the table is able to select, update, insert, and delete this view.
Appendix D. Supported CCSIDs

The following CCSIDs are supported on DB2 Net Search Extender.

<table>
<thead>
<tr>
<th>CCSID</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>37</td>
<td>USA/Canada - CECP</td>
</tr>
<tr>
<td>273</td>
<td>Germany F.R./Austria - CECP</td>
</tr>
<tr>
<td>274</td>
<td>Old Belgium Code Page</td>
</tr>
<tr>
<td>277</td>
<td>Denmark, Norway - CECP</td>
</tr>
<tr>
<td>278</td>
<td>Finland, Sweden - CECP</td>
</tr>
<tr>
<td>280</td>
<td>Italy - CECP</td>
</tr>
<tr>
<td>284</td>
<td>Spain/Latin America - CECP</td>
</tr>
<tr>
<td>285</td>
<td>United Kingdom - CECP</td>
</tr>
<tr>
<td>290</td>
<td>Japanese (Katakana) Extended</td>
</tr>
<tr>
<td>297</td>
<td>France - CECP</td>
</tr>
<tr>
<td>301</td>
<td>Japan DB PC</td>
</tr>
<tr>
<td>367</td>
<td>ASCII</td>
</tr>
<tr>
<td>420</td>
<td>Arabic Bilingual</td>
</tr>
<tr>
<td>423</td>
<td>Greece - 183</td>
</tr>
<tr>
<td>424</td>
<td>Israel (Hebrew)</td>
</tr>
<tr>
<td>437</td>
<td>US English</td>
</tr>
<tr>
<td>500</td>
<td>International #5</td>
</tr>
<tr>
<td>737</td>
<td>MS DOS Greek</td>
</tr>
<tr>
<td>806</td>
<td>Hindi</td>
</tr>
<tr>
<td>813</td>
<td>Greek</td>
</tr>
<tr>
<td>819</td>
<td>Latin-1</td>
</tr>
<tr>
<td>833</td>
<td>Korean Extended</td>
</tr>
<tr>
<td>836</td>
<td>Simplified Chinese Extended</td>
</tr>
<tr>
<td>838</td>
<td>Thai with Low Tone Marks &amp; Ancient Characters</td>
</tr>
</tbody>
</table>
### CCSIDs

<table>
<thead>
<tr>
<th>CCSID</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>848</td>
<td>PC, Cyrillic, Ukrainian with Euro</td>
</tr>
<tr>
<td>850</td>
<td>Latin-1</td>
</tr>
<tr>
<td>852</td>
<td>Latin-2</td>
</tr>
<tr>
<td>855</td>
<td>Bulgarian</td>
</tr>
<tr>
<td>857</td>
<td>Turkish</td>
</tr>
<tr>
<td>858</td>
<td>Personal Computer - Multilingual with Euro</td>
</tr>
<tr>
<td>860</td>
<td>Portuguese</td>
</tr>
<tr>
<td>862</td>
<td>Hebrew</td>
</tr>
<tr>
<td>863</td>
<td>Canadian</td>
</tr>
<tr>
<td>864</td>
<td>Arabic</td>
</tr>
<tr>
<td>866</td>
<td>Russian</td>
</tr>
<tr>
<td>867</td>
<td>Israel - Personal Computer</td>
</tr>
<tr>
<td>869</td>
<td>Greek</td>
</tr>
<tr>
<td>870</td>
<td>Latin 2 - EBCDIC Multilingual</td>
</tr>
<tr>
<td>871</td>
<td>Iceland - CECP</td>
</tr>
<tr>
<td>872</td>
<td>Cyrillic - PC with Euro</td>
</tr>
<tr>
<td>874</td>
<td>Thai</td>
</tr>
<tr>
<td>875</td>
<td>Greece</td>
</tr>
<tr>
<td>891</td>
<td>Korea - Personal Computer</td>
</tr>
<tr>
<td>895</td>
<td>Japan 7-Bit Latin</td>
</tr>
<tr>
<td>901</td>
<td>PC Baltic Multi with Euro</td>
</tr>
<tr>
<td>902</td>
<td>8-bit Estonia with Euro</td>
</tr>
<tr>
<td>912</td>
<td>Latin-2</td>
</tr>
<tr>
<td>915</td>
<td>Russian</td>
</tr>
<tr>
<td>916</td>
<td>Hebrew</td>
</tr>
<tr>
<td>920</td>
<td>Turkish</td>
</tr>
<tr>
<td>921</td>
<td>Latvian, Estonian</td>
</tr>
<tr>
<td>922</td>
<td>Estonian</td>
</tr>
<tr>
<td>923</td>
<td>Latin 9</td>
</tr>
<tr>
<td>924</td>
<td>Latin 9 EBCDIC</td>
</tr>
<tr>
<td>927</td>
<td>Taiwan PC</td>
</tr>
<tr>
<td>CCSID</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>930</td>
<td>Japan EBCDIC</td>
</tr>
<tr>
<td>932</td>
<td>Japanese, combined SBCS/DBCS</td>
</tr>
<tr>
<td>933</td>
<td>Korean</td>
</tr>
<tr>
<td>935</td>
<td>Chinese (simplified)</td>
</tr>
<tr>
<td>937</td>
<td>Chinese (traditional)</td>
</tr>
<tr>
<td>938</td>
<td>Taiwan PC</td>
</tr>
<tr>
<td>939</td>
<td>Japanese</td>
</tr>
<tr>
<td>941</td>
<td>Japan OPEN</td>
</tr>
<tr>
<td>942</td>
<td>Japanese, combined SBCS/DBCS</td>
</tr>
<tr>
<td>943</td>
<td>Japanese, combined SBCS/DBCS</td>
</tr>
<tr>
<td>948</td>
<td>Chinese (traditional), combined SBCS/DBCS</td>
</tr>
<tr>
<td>949</td>
<td>Korean</td>
</tr>
<tr>
<td>950</td>
<td>Chinese (traditional), combined SBCS/DBCS</td>
</tr>
<tr>
<td>954</td>
<td>Japanese</td>
</tr>
<tr>
<td>964</td>
<td>Chinese (traditional), combined SBCS/DBCS</td>
</tr>
<tr>
<td>970</td>
<td>Korean</td>
</tr>
<tr>
<td>1025</td>
<td>Cyrillic, Multilingual</td>
</tr>
<tr>
<td>1026</td>
<td>Latin #5 - Turkey</td>
</tr>
<tr>
<td>1027</td>
<td>Japanese (Latin) Extended</td>
</tr>
<tr>
<td>1040</td>
<td>Korean Extended - Personal Computer</td>
</tr>
<tr>
<td>1041</td>
<td>Japanese Extended - Personal Computer</td>
</tr>
<tr>
<td>1043</td>
<td>Traditional Chinese Extended - PC</td>
</tr>
<tr>
<td>1046</td>
<td>Arabic</td>
</tr>
<tr>
<td>1047</td>
<td>Latin 1/Open Systems</td>
</tr>
<tr>
<td>1051</td>
<td>H-P Emulation, Roman 8</td>
</tr>
<tr>
<td>1088</td>
<td>Revised Korean - Personal Computer</td>
</tr>
<tr>
<td>1089</td>
<td>Arabic</td>
</tr>
<tr>
<td>1112</td>
<td>Baltic - Multilingual, EBCDIC</td>
</tr>
<tr>
<td>1115</td>
<td>People’s Republic of China (PRC)-PC</td>
</tr>
<tr>
<td>1122</td>
<td>Estonia, EBCDIC</td>
</tr>
<tr>
<td>1123</td>
<td>Cyrillic, Ukraine</td>
</tr>
</tbody>
</table>
### CCSIDs

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1124</td>
<td>Ukrainian</td>
</tr>
<tr>
<td>1125</td>
<td>Ukrainian</td>
</tr>
<tr>
<td>1131</td>
<td>Vietnamese</td>
</tr>
<tr>
<td>1137</td>
<td>Devanagari EBCDIC</td>
</tr>
<tr>
<td>1140</td>
<td>USA, Canada, etc. ECECP</td>
</tr>
<tr>
<td>1141</td>
<td>Austria, Germany ECECP</td>
</tr>
<tr>
<td>1142</td>
<td>Denmark, Norway ECECP</td>
</tr>
<tr>
<td>1143</td>
<td>Finland, Sweden ECECP</td>
</tr>
<tr>
<td>1144</td>
<td>Italy ECECP</td>
</tr>
<tr>
<td>1145</td>
<td>Spain, Latin America (Spanish) ECECP</td>
</tr>
<tr>
<td>1146</td>
<td>UK ECECP</td>
</tr>
<tr>
<td>1147</td>
<td>France ECECP</td>
</tr>
<tr>
<td>1148</td>
<td>International ECECP</td>
</tr>
<tr>
<td>1149</td>
<td>Iceland ECECP</td>
</tr>
<tr>
<td>1153</td>
<td>EBCDIC Latin 2 Multilingual with Euro</td>
</tr>
<tr>
<td>1154</td>
<td>EBCDIC Cyrillic, Multilingual with Euro</td>
</tr>
<tr>
<td>1155</td>
<td>EBCDIC Turkey with Euro</td>
</tr>
<tr>
<td>1156</td>
<td>EBCDIC Baltic Multi with Euro</td>
</tr>
<tr>
<td>1157</td>
<td>EBCDIC Estonia with Euro</td>
</tr>
<tr>
<td>1158</td>
<td>EBCDIC Cyrillic, Ukraine with Euro</td>
</tr>
<tr>
<td>1159</td>
<td>T-Chinese EBCDIC</td>
</tr>
<tr>
<td>1160</td>
<td>Thai with Low Tone Marks &amp; Ancient Characters</td>
</tr>
<tr>
<td>1161</td>
<td>Thai with Low Tone Marks &amp; Ancient Chars - PC</td>
</tr>
<tr>
<td>1162</td>
<td>Thai MS Windows</td>
</tr>
<tr>
<td>1163</td>
<td>Vietnamese ISO-8 with Euro</td>
</tr>
<tr>
<td>1164</td>
<td>Vietnamese EBCDIC with Euro</td>
</tr>
<tr>
<td>1200</td>
<td>UCS2</td>
</tr>
<tr>
<td>1208</td>
<td>UTF8</td>
</tr>
<tr>
<td>1250</td>
<td>Latin-2, Belorussian</td>
</tr>
<tr>
<td>1251</td>
<td>Russian</td>
</tr>
<tr>
<td>1252</td>
<td>Latin-1</td>
</tr>
</tbody>
</table>
### CCSIDs

<table>
<thead>
<tr>
<th>Code</th>
<th>Language / Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1253</td>
<td>Czech</td>
</tr>
<tr>
<td>1254</td>
<td>Turkish</td>
</tr>
<tr>
<td>1255</td>
<td>Hebrew</td>
</tr>
<tr>
<td>1256</td>
<td>Arabic</td>
</tr>
<tr>
<td>1258</td>
<td>Vietnamese</td>
</tr>
<tr>
<td>1275</td>
<td>Apple, Latin 1</td>
</tr>
<tr>
<td>1280</td>
<td>Apple Greek</td>
</tr>
<tr>
<td>1281</td>
<td>Apple Turkish</td>
</tr>
<tr>
<td>1282</td>
<td>Apple Central European</td>
</tr>
<tr>
<td>1283</td>
<td>Apple Cyrillic</td>
</tr>
<tr>
<td>1351</td>
<td>Japan OPEN</td>
</tr>
<tr>
<td>1363</td>
<td>Korean</td>
</tr>
<tr>
<td>1364</td>
<td>Korean</td>
</tr>
<tr>
<td>1381</td>
<td>Chinese (simplified), combined SBCS/DBCS</td>
</tr>
<tr>
<td>1383</td>
<td>Chinese (simplified), combined SBCS/DBCS</td>
</tr>
<tr>
<td>1386</td>
<td>Chinese (simplified), combined SBCS/DBCS</td>
</tr>
<tr>
<td>1388</td>
<td>Chinese (simplified), combined SBCS/DBCS</td>
</tr>
<tr>
<td>1390</td>
<td>Japanese</td>
</tr>
<tr>
<td>1392</td>
<td>China GB18030</td>
</tr>
<tr>
<td>1399</td>
<td>Japan EBCDIC</td>
</tr>
<tr>
<td>4909</td>
<td>Greece/Latin ASCII</td>
</tr>
<tr>
<td>4930</td>
<td>Korea DB EBCDIC</td>
</tr>
<tr>
<td>4933</td>
<td>China EBCDIC</td>
</tr>
<tr>
<td>4971</td>
<td>Greece EBCDIC</td>
</tr>
<tr>
<td>5026</td>
<td>Japanese Katakana</td>
</tr>
<tr>
<td>5035</td>
<td>Japanese Latin</td>
</tr>
<tr>
<td>5039</td>
<td>Japanese, combined SBCS/DBCS</td>
</tr>
<tr>
<td>5123</td>
<td>Japan Latin EBCD</td>
</tr>
<tr>
<td>5210</td>
<td>China SB PC</td>
</tr>
<tr>
<td>5346</td>
<td>Windows Latin-2</td>
</tr>
</tbody>
</table>

Appendix D. Supported CCSIDs
## CCSIDs

<table>
<thead>
<tr>
<th>CCSID</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>5347</td>
<td>Windows Cyrillic</td>
</tr>
<tr>
<td>5348</td>
<td>Windows Latin-1</td>
</tr>
<tr>
<td>5349</td>
<td>Windows Greece</td>
</tr>
<tr>
<td>5350</td>
<td>Windows Turkey</td>
</tr>
<tr>
<td>5351</td>
<td>Windows Hebrew with Euro</td>
</tr>
<tr>
<td>5352</td>
<td>Windows Arabic</td>
</tr>
<tr>
<td>5353</td>
<td>Windows Baltic</td>
</tr>
<tr>
<td>5354</td>
<td>Vietnamese</td>
</tr>
<tr>
<td>8482</td>
<td>Japanese Katakana</td>
</tr>
<tr>
<td>9027</td>
<td>Taiwan DB EBCD</td>
</tr>
<tr>
<td>9044</td>
<td>Latin-2 PC</td>
</tr>
<tr>
<td>9048</td>
<td>Hebrew PC</td>
</tr>
<tr>
<td>9049</td>
<td>Turkey PC</td>
</tr>
<tr>
<td>9061</td>
<td>Greece PC</td>
</tr>
<tr>
<td>9238</td>
<td>Arabic - PC</td>
</tr>
<tr>
<td>12712</td>
<td>Hebrew EBCDIC</td>
</tr>
<tr>
<td>13121</td>
<td>Korea SB EBCDIC</td>
</tr>
<tr>
<td>13488</td>
<td>UCS2</td>
</tr>
<tr>
<td>16684</td>
<td>Japanese Katakana DB EBCDIC</td>
</tr>
<tr>
<td>17248</td>
<td>Arabic PC</td>
</tr>
<tr>
<td>17584</td>
<td>UCS-2</td>
</tr>
<tr>
<td>18030</td>
<td></td>
</tr>
<tr>
<td>21427</td>
<td>Taiwan BIG-5</td>
</tr>
<tr>
<td>33722</td>
<td>IBMuecJP</td>
</tr>
<tr>
<td>61955</td>
<td></td>
</tr>
<tr>
<td>65500</td>
<td></td>
</tr>
<tr>
<td>65501</td>
<td></td>
</tr>
</tbody>
</table>
## Appendix E. Supported languages

These are the language parameters that you can specify in DB2 Net Search Extender. In Net Search Extender, the only significant language-specific processing is done on documents written in the Thai language.

<table>
<thead>
<tr>
<th>Code</th>
<th>Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>AR_AA</td>
<td>Arabic/Arabic Speaking</td>
</tr>
<tr>
<td>BE_BY</td>
<td>Belorussian/Belarus</td>
</tr>
<tr>
<td>BG_BG</td>
<td>Bulgarian/Bulgaria</td>
</tr>
<tr>
<td>CA_ES</td>
<td>Catalan/Spain</td>
</tr>
<tr>
<td>CS_CZ</td>
<td>Czech/Czech Republic</td>
</tr>
<tr>
<td>DA_DK</td>
<td>Danish/Denmark</td>
</tr>
<tr>
<td>DE_CH</td>
<td>German/Switzerland</td>
</tr>
<tr>
<td>DE_DE</td>
<td>German/Germany</td>
</tr>
<tr>
<td>EL_GR</td>
<td>Greek/Greece</td>
</tr>
<tr>
<td>EN_AU</td>
<td>English/Australia</td>
</tr>
<tr>
<td>EN_BE</td>
<td>English/Belgium</td>
</tr>
<tr>
<td>EN_GB</td>
<td>English/U.K.</td>
</tr>
<tr>
<td>EN_US</td>
<td>English/U.S.</td>
</tr>
<tr>
<td>EN_ZA</td>
<td>English/South Africa</td>
</tr>
<tr>
<td>ES_ES</td>
<td>Spanish/Spain</td>
</tr>
<tr>
<td>ET_EE</td>
<td>Estonian/Estonia</td>
</tr>
<tr>
<td>FI_FI</td>
<td>Finnish/Finland</td>
</tr>
<tr>
<td>FR_BE</td>
<td>French/Belgium</td>
</tr>
<tr>
<td>FR_CA</td>
<td>French/Canada</td>
</tr>
<tr>
<td>FR_CH</td>
<td>French/Switzerland</td>
</tr>
<tr>
<td>FR_FR</td>
<td>French/France</td>
</tr>
<tr>
<td>HE_IL</td>
<td>Hebrew/Israel</td>
</tr>
<tr>
<td>HI_IN</td>
<td>Hindi/India</td>
</tr>
<tr>
<td>HR_HR</td>
<td>Croatian/Croatia</td>
</tr>
<tr>
<td>HU_HU</td>
<td>Hungarian/Hungary</td>
</tr>
</tbody>
</table>
**Languages**

<table>
<thead>
<tr>
<th>Code</th>
<th>Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID_ID</td>
<td>Indonesian/Indonesia</td>
</tr>
<tr>
<td>IT_CH</td>
<td>Italian/Switzerland</td>
</tr>
<tr>
<td>IW_IL</td>
<td>Hebrew/Israel</td>
</tr>
<tr>
<td>IT_IT</td>
<td>Italian/Italy</td>
</tr>
<tr>
<td>JA_JP</td>
<td>Japanese/Japan</td>
</tr>
<tr>
<td>KO_KR</td>
<td>Korean/Korea</td>
</tr>
<tr>
<td>LT_LT</td>
<td>Lithuanian/Lithuania</td>
</tr>
<tr>
<td>LV_LV</td>
<td>Latvian/Latvia</td>
</tr>
<tr>
<td>MK_MK</td>
<td>Macedonian/FYR Macedonia</td>
</tr>
<tr>
<td>MS_MY</td>
<td>Malay/Malaysia</td>
</tr>
<tr>
<td>NB_NO</td>
<td>Norwegian Bokmal/Norway</td>
</tr>
<tr>
<td>NL_BE</td>
<td>Dutch/Belgium</td>
</tr>
<tr>
<td>NL_NL</td>
<td>Dutch/Netherlands</td>
</tr>
<tr>
<td>NN_NO</td>
<td>Norwegian Nynorsk/Norway</td>
</tr>
<tr>
<td>NO_NO</td>
<td>Norwegian/Norway</td>
</tr>
<tr>
<td>PT_BR</td>
<td>Portuguese/Brazil</td>
</tr>
<tr>
<td>PL_PL</td>
<td>Polish/Poland</td>
</tr>
<tr>
<td>PT_PT</td>
<td>Portuguese/Portugal</td>
</tr>
<tr>
<td>RO_RO</td>
<td>Romanian/Romania</td>
</tr>
<tr>
<td>RU_RU</td>
<td>Russian/Russia</td>
</tr>
<tr>
<td>SH_SP</td>
<td>Serbian (Latin)/Serbia</td>
</tr>
<tr>
<td>SK_SK</td>
<td>Slovak/Slovakia</td>
</tr>
<tr>
<td>SL_SI</td>
<td>Slovenian/Slovenia</td>
</tr>
<tr>
<td>SQ_AL</td>
<td>Albanian/Albania</td>
</tr>
<tr>
<td>SR_SP</td>
<td>Serbian (Cyrillic)/Serbia</td>
</tr>
<tr>
<td>SV_SE</td>
<td>Swedish/Sweden</td>
</tr>
<tr>
<td>TA_IN</td>
<td>Tamil/India</td>
</tr>
<tr>
<td>TE_IN</td>
<td>Telugu/India</td>
</tr>
<tr>
<td>TH_TH</td>
<td>Thai/Thailand</td>
</tr>
<tr>
<td>TR_TR</td>
<td>Turkish/Turkey</td>
</tr>
<tr>
<td>UK_UA</td>
<td>Ukrainian/Ukraine</td>
</tr>
<tr>
<td>Code</td>
<td>Language</td>
</tr>
<tr>
<td>--------</td>
<td>----------------</td>
</tr>
<tr>
<td>VI_VN</td>
<td>Vietnamese/Vietnam</td>
</tr>
<tr>
<td>ZH_CN</td>
<td>Chinese/PRC</td>
</tr>
<tr>
<td>ZH_TW</td>
<td>Chinese/Taiwan</td>
</tr>
</tbody>
</table>
Languages
Appendix F. Net Search Extender messages

DB2 Net Search Extender provides the following message types:

- Information and warning messages
- Error messages

Note that the SQL states returned from the search function are 38600 plus the CTE error number.

### Information and warning messages

<table>
<thead>
<tr>
<th>CTE0001</th>
<th>Operation completed successfully.</th>
</tr>
</thead>
<tbody>
<tr>
<td>CTE0002</td>
<td>The update and locking services are up and running.</td>
</tr>
<tr>
<td>CTE0003</td>
<td>Index update started.</td>
</tr>
<tr>
<td>CTE0004</td>
<td>Index update ended.</td>
</tr>
<tr>
<td>CTE0005</td>
<td>Index update commit: &quot;%1&quot;,&quot;%2&quot;,&quot;%3&quot; documents inserted, updated, and/or deleted successfully.</td>
</tr>
<tr>
<td>CTE0006</td>
<td>Problem accessing text index. Check db2diag.log for details.</td>
</tr>
<tr>
<td>CTE0007</td>
<td>The section &quot;%1&quot; does not occur in any of the documents or is not a valid document model section name.</td>
</tr>
<tr>
<td>CTE0008</td>
<td>Index reorganization started.</td>
</tr>
<tr>
<td>CTE0009</td>
<td>Index reorganization ended.</td>
</tr>
<tr>
<td>CTE0010</td>
<td>The attribute &quot;%1&quot; is not valid.</td>
</tr>
<tr>
<td>CTE0011</td>
<td>Cache activation started.</td>
</tr>
<tr>
<td>CTE0012</td>
<td>Cache activation ended.</td>
</tr>
<tr>
<td>CTE0013</td>
<td>Persistent cache was removed.</td>
</tr>
<tr>
<td>CTE0014</td>
<td>Cache deactivated.</td>
</tr>
</tbody>
</table>

### Error messages

<table>
<thead>
<tr>
<th>CTE0100</th>
<th>A DB2 operation failed. DB2 information: &quot;%2&quot; &quot;%4&quot;.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Explanation: A DB2 error occurred that does not allow further processing.</td>
<td></td>
</tr>
<tr>
<td>What to do:</td>
<td>For more detailed information on this DB2 error, use db2 ? SQLxxx.</td>
</tr>
</tbody>
</table>

© Copyright IBM Corp. 1995, 2003
## Error messages

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
<th>Reason Code</th>
<th>What to Do</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>CTE0101</td>
<td>A search engine operation failed.</td>
<td>&quot;%2&quot;, &quot;%3&quot;, &quot;%4&quot;, &quot;%5&quot;, &quot;%6&quot;</td>
<td>For more detailed information, see the Search Engine reason code descriptions.</td>
<td>A Search Engine error occurred that does not allow further processing.</td>
</tr>
<tr>
<td>CTE0102</td>
<td>A general system function failed.</td>
<td>&quot;%2&quot;</td>
<td>Call the db2 alter table to ensure the existence of a primary key. Then try to create the index again.</td>
<td>You tried to create an index on a table that does not have a primary key.</td>
</tr>
<tr>
<td>CTE0103</td>
<td>An internal error occurred. Location: &quot;%1&quot;, &quot;%2&quot;.</td>
<td></td>
<td>You specified a directory which does not exist.</td>
<td>You specified a directory which does not exist.</td>
</tr>
<tr>
<td>CTE0104</td>
<td>Memory allocation error (search engine).</td>
<td></td>
<td>Change the layout of the table before creating the index again. Use smaller key columns, which also benefit performance.</td>
<td>The internal representation of the key columns exceeds the maximum size.</td>
</tr>
<tr>
<td>CTE0105</td>
<td>Memory allocation error.</td>
<td></td>
<td>Change the layout of the table before creating the index again.</td>
<td>A maximum number of 14 key columns is supported.</td>
</tr>
<tr>
<td>CTE0106</td>
<td>Table &quot;%1&quot;.&quot;%2&quot; has no primary key.</td>
<td></td>
<td></td>
<td>You tried to create an index on a table that does not have a primary key.</td>
</tr>
<tr>
<td>CTE0107</td>
<td>Directory &quot;%1&quot; does not exist.</td>
<td></td>
<td>Create the directory, ensure accessibility to the instance owner. Then try to specify the directory again.</td>
<td>You specified a directory which does not exist.</td>
</tr>
<tr>
<td>CTE0108</td>
<td>The internal size &quot;%4&quot; of the key columns on object &quot;%1&quot;.&quot;%2&quot; is larger than maximum allowed size of &quot;%3&quot;.</td>
<td></td>
<td>Change the layout of the table before creating the index again.</td>
<td>The internal representation of the key columns exceeds the maximum size.</td>
</tr>
<tr>
<td>CTE0109</td>
<td>The number of key columns &quot;%3&quot; on object &quot;%1&quot;.&quot;%2&quot; is larger than the allowed maximum of &quot;%4&quot;.</td>
<td></td>
<td>Change the layout of the table before creating the index again.</td>
<td>A maximum number of 14 key columns is supported.</td>
</tr>
<tr>
<td>CTE0111</td>
<td>The file &quot;%1&quot; is not readable.</td>
<td></td>
<td>Check the access rights for the file. Take into account that the Stored Procedure runs as a fenced user ID, which may also require rights to work on this file.</td>
<td>The file specified cannot be read.</td>
</tr>
</tbody>
</table>
### Error messages

<table>
<thead>
<tr>
<th>Code</th>
<th>Message</th>
<th>Explanation</th>
<th>What to do</th>
</tr>
</thead>
<tbody>
<tr>
<td>CTE0112</td>
<td>The file &quot;%1&quot; cannot be opened.</td>
<td>The file specified could not be opened.</td>
<td>Verify that the file is correctly specified.</td>
</tr>
<tr>
<td>CTE0113</td>
<td>Error converting model file &quot;%1&quot; to UTF8 encoding.</td>
<td>The specified CCSID or the default database CCSID does not match the model file CCSID.</td>
<td>Ensure correct specification of the model file CCSID.</td>
</tr>
<tr>
<td>CTE0114</td>
<td>Unable to register document model &quot;%1&quot; in file &quot;%2&quot;.</td>
<td>The model file could not be used.</td>
<td>Check that the model file syntax is correctly specified.</td>
</tr>
<tr>
<td>CTE0115</td>
<td>A locking problem occurred. Lock Manager information: &quot;%1&quot; &quot;%2&quot;.</td>
<td>An internal locking problem occurred.</td>
<td>Check the current locks using the db2text control command. Using the same command, clean up the pending locks. If this does not help, stop and restart the locking and update services.</td>
</tr>
<tr>
<td>CTE0116</td>
<td>Operation conflicts with existing lock.</td>
<td>You have tried to use a command that is currently not allowed when other commands are running on the index.</td>
<td>Check the locks held on this index to see which commands are currently running. Wait until the other commands have finished. If the operation is no longer running but the lock is still active, clean up the locks for the index and try again.</td>
</tr>
<tr>
<td>CTE0117</td>
<td>All available lock space for databases is used. Change the configuration.</td>
<td>You tried to work on more databases than are configured in your lock file.</td>
<td>Change the number of databases you want to work in parallel with in your lock configuration file db2extlm.cfg. Restart the update and locking services using the db2text stop and db2text start commands.</td>
</tr>
<tr>
<td>CTE0118</td>
<td>All available lock space for indexes on a databases is used. Change the configuration.</td>
<td>You tried to work on more indexes for one database than are configured in your lock file.</td>
<td>Change the number of indexes you want to work in parallel with in your lock configuration file db2extlm.cfg. Restart the update and locking services using the db2text stop and db2text start commands.</td>
</tr>
<tr>
<td>CTE0119</td>
<td>All available space for locks on an index is used.</td>
<td>The operations you are running require more locks for one index than are configured in your lock configuration file.</td>
<td>Change the number of locks you want to work in parallel with in your lock configuration file db2extlm.cfg. Restart the update and locking services using the db2text stop and db2text start commands.</td>
</tr>
<tr>
<td>CTE0120</td>
<td>Update and locking services configuration file error.</td>
<td>The configuration file db2extlm.cfg is in error.</td>
<td>Check the db2extlm.cfg file and correct the error. Restart the update and locking services using the command db2text start.</td>
</tr>
</tbody>
</table>
### Error messages

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
<th>Explanation</th>
<th>What to do</th>
</tr>
</thead>
<tbody>
<tr>
<td>CTE0121</td>
<td>The update and locking services configuration file cannot be opened.</td>
<td>The file <code>db2extlm.cfg</code> could not be opened.</td>
<td>Check if the file exists and that it can be accessed. If the file cannot be accessed, try to update your db2 instance using <code>db2iupdt</code>.</td>
</tr>
<tr>
<td>CTE0122</td>
<td>A syntax error was found in the update and locking services configuration file.</td>
<td>A syntax error was found in the update and locking services configuration file.</td>
<td>Check the update and locking services configuration file for errors.</td>
</tr>
<tr>
<td>CTE0126</td>
<td>The update and locking service input file &quot;)%1&quot; is corrupted.</td>
<td>A required file for update and locking services is corrupted.</td>
<td>Check if the file exists and if it can be accessed. If you can access the file, rename the file and restart the update and locking services. The file should be created again. However, this action removes all of the specified frequency updates for create index.</td>
</tr>
<tr>
<td>CTE0127</td>
<td>An update and locking service error has occurred. Reason code: &quot;)%1&quot;.</td>
<td>An internal error has occurred in the update and locking service area.</td>
<td>Stop DB2 and Net Search Extender and then clean up your shared resources. Try to start both again. If this does not work, report the problem to your IBM representative.</td>
</tr>
<tr>
<td>CTE0129</td>
<td>NULL values are not allowed to be passed as parameters.</td>
<td>DB2 has passed a NULL value to an internal user-defined function.</td>
<td></td>
</tr>
<tr>
<td>CTE0130</td>
<td>The specified search argument exceeds the maximum length. The current search argument length is &quot;)%1&quot; and the maximum supported length is &quot;)%2&quot;.</td>
<td>The length of the specified search argument is &quot;)%1&quot;. The maximum length must not exceed &quot;)%2&quot;.</td>
<td>Reduce the length of your search argument to &quot;)%2&quot;.</td>
</tr>
<tr>
<td>CTE0131</td>
<td>The user-defined function &quot;&quot;)%1&quot;. &quot;)%2&quot; does not exist.</td>
<td>The specified user-defined function does not exist in this database.</td>
<td>Check the name specified for this user-defined function, or register the user-defined function in the database you are using.</td>
</tr>
<tr>
<td>CTE0132</td>
<td>The text index &quot;&quot;)%1&quot;. &quot;)%2&quot; does not exist.</td>
<td>The specified text index does not exist in this database.</td>
<td>Check the name specified and the database you are using. Use the <code>db2ext.textcolumns</code> view to see the existing text indexes.</td>
</tr>
<tr>
<td>CTE0133</td>
<td>The text index &quot;&quot;)%1&quot;. &quot;)%2&quot; already exists.</td>
<td>The text index that you specified already exists in this database.</td>
<td>Check the name specified and the database you are using. Use the <code>db2ext.textcolumns</code> view to see the existing text indexes.</td>
</tr>
<tr>
<td>Error code</td>
<td>Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CTE0135</td>
<td>The object &quot;%1&quot;. &quot;%2&quot; does not exist.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Explanation:</td>
<td>The specified object name does not exist in this database.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>What to do:</td>
<td>Check the object name specified and the database you are using.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CTE0136</td>
<td>The column &quot;%1&quot; does not exist in &quot;%2&quot;.&quot;%3&quot;.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Explanation:</td>
<td>The specified column does not exist.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>What to do:</td>
<td>Check the column name that you specified. Check the table, view, or database you are using.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CTE0137</td>
<td>The table space &quot;%1&quot; does not exist.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Explanation:</td>
<td>The specified tablespace does not exist in this database.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>What to do:</td>
<td>Check the name specified and the database you are using.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CTE0138</td>
<td>The table space &quot;%1&quot; is not regular.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Explanation:</td>
<td>The specified table space is not regular. The event table can only be created in a regular table space.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>What to do:</td>
<td>Use this command again with a regular table space.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CTE0139</td>
<td>The environment variable &quot;%1&quot; is not set.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Explanation:</td>
<td>A required environment variable is not set.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>What to do:</td>
<td>Check your environment, specify the required variable, and use the command again.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CTE0140</td>
<td>The database &quot;%1&quot; is already enabled for text.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Explanation:</td>
<td>The database you specified is already enabled for text.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>What to do:</td>
<td>Check the name that you specified. Also check the DB2DBDFT variable that implies an implicit connection.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CTE0141</td>
<td>The database &quot;%1&quot; is not enabled for text.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Explanation:</td>
<td>The database you specified is not enabled for text.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>What to do:</td>
<td>Check the database name you specified and the DB2DBDFT variable. If the database name is correct, use the command db2text enable database for text.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CTE0142</td>
<td>The command requires control authority on &quot;%1&quot;. &quot;%2&quot; granted to user &quot;%3&quot;.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Explanation:</td>
<td>You do not have the authority to use this command.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>What to do:</td>
<td>Only the owner of this table can use this command or provide you with the required authorization.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CTE0143</td>
<td>The command requires database administration authority for user &quot;%1&quot;.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Explanation:</td>
<td>You do not have the required authority to use this command.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>What to do:</td>
<td>Only the owner of the database can use this command or provide you with the required authorization.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CTE0144</td>
<td>There is at least one text index active in database &quot;%1&quot;.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Explanation:</td>
<td>You cannot disable your database until all text indexes are dropped.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>What to do:</td>
<td>See the db2ext.textcolumns view for the existing indexes. Drop the existing indexes using the DROP INDEX command or...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Error messages

specify the FORCE option with the DISABLE DATABASE command.

**CTE0145**  The CCSID "%1" is not supported.
Explanation:  The CCSID that you specified is not supported.
What to do:  Specify a valid CCSID.

**CTE0146**  The language "%1" is not supported.
Explanation:  The specified language is not supported.
What to do:  Specify a valid language.

**CTE0147**  The format "%1" is not supported.
Explanation:  The specified format is not supported.
What to do:  Specify a valid format.

**CTE0148**  The specified format "%1" does not accept a model file.
Explanation:  The format "%1" does not support model files.
What to do:  Use a format that accepts a model file, or remove the model file from your command.

**CTE0149**  Too many terms (beginning with "%1") are specified for the index update frequency.
Explanation:  The syntax for the update frequency is not correct.
What to do:  Ensure that the DAY, HOUR, and MINUTE parameters are only specified once.

**CTE0150**  The search string is empty.
Explanation:  You specified an empty search string.
What to do:  Ensure that the DAY, HOUR, and MINUTE parameters are only specified once.

**CTE0151**  Token "%1" is unexpected. Check the command syntax.
Explanation:  The syntax of the command is not correct.
What to do:  Check the command syntax and verify that the token you are using is allowed in the specific command.

**CTE0152**  Token "%1" is too long.
Explanation:  The token is too long.
What to do:  Check the command syntax and verify that the token is reduced to the maximum size allowed.

**CTE0153**  Token "%1" occurs twice in the update frequency.
Explanation:  You specified an incorrect syntax for the update frequency.
What to do:  Ensure that the DAY, HOUR, and MINUTE parameters are only specified once.

**CTE0154**  The value "%1" for "%2" is out of range. The valid range is "%3" - "%4".
Explanation:  You specified an incorrect value. The value should be in the allowed range.
What to do:  Update your command. Change the value to match those in the allowed range.

**CTE0155**  The search string is empty.
Explanation:  You specified an empty search string.
What to do:  Ensure that the DAY, HOUR, and MINUTE parameters are only specified once.

**CTE0156**  Syntax error near "%1".
Explanation:  You specified an incorrect search syntax.
What to do: Check the syntax near %1. Correct and try again.

CTE0158 The freetext search string is missing.
Explanation: Specify a freetext string.
What to do: Check that the search string after "is about" includes valid alphanumeric characters.

CTE0159 Search string exceeds the allowed length of "%1".
Explanation: The search string is too long.
What to do: Reduce the size of the search string and try again.

CTE0160 No section name has been specified in the search string.
Explanation: You need to specify a valid section name.
What to do: Add a valid section name and try again.

CTE0162 The escape command could not be processed.
Explanation: Your search string includes too many special characters that can be used as masking characters.
What to do: Reduce the number of special characters in your search term, or avoid the escape command. The following special characters can be used: * + . : ; { } ~ ! ? ` = \n
CTE0163 No thesaurus name specified in thesaurus clause.
Explanation: A thesaurus search is requested without a thesaurus name.
What to do: Specify a thesaurus name in your search argument.

CTE0164 Syntax error in thesaurus relation "%1".
Explanation: The specified syntax for the thesaurus relation is not correct.
What to do: Update the thesaurus relation according to the syntax specification.

CTE0166 Freetext must be the last statement in search query.
Explanation: It is not allowed to have further operators after the "is about" token.
What to do: Rewrite the query string. The last operator must be "is about".

CTE0167 Syntax error in free text query "%1".
Explanation: The syntax for the free text string is not correct.
What to do: Update the free text string according to the syntax specification.

CTE0168 A left parenthesis in a section statement is missing.
Explanation: The syntax for the section statement is not correct.
What to do: Update the section statement according to the syntax specification.

CTE0169 A comma or right parenthesis is missing in a section statement.
Explanation: The syntax for the section statement is not correct.
What to do: Update the section statement according to the syntax specification.

CTE0170 A closing double quote is missing.
Explanation: The specified syntax for the search term is not correct.
What to do: Update the search term according
Error messages

to the syntax specification.

CTE0171 An open double quote for a section name is missing.

Explanation: The syntax for the section statement is not correct.
What to do: Update the section statement according to the syntax specification.

CTE0172 The closing double quote for the section name is missing.

Explanation: The syntax for the section statement is not correct.
What to do: Update the section statement according to the syntax specification.

CTE0173 One escape character must be defined in an escape clause.

Explanation: There can be no more than one character in an escape clause.
What to do: Remove the additional characters in the escape clause.

CTE0174 A blank character is not allowed as an escape character.

Explanation: It is not allowed to have a blank character in an escape clause.
What to do: Change the escape clause to a clause with a valid character.

CTE0175 An escape clause is defined but no mask character is found in the search phrase.

Explanation: An escape clause is specified without using a mask character.
What to do: Remove the escape clause.

CTE0176 The succeeding character of an escape character in the phrase is neither the same character nor a mask character.

Explanation: The character after the escape character must be either a masking character or the escape character itself.
What to do: Change the search string to correctly use the escape character.

CTE0177 The number value "%1" is invalid.

Explanation: The specified number in the search argument is not valid.
What to do: Check the documentation about the valid range. Update the value in the search argument.

CTE0178 Mask characters in fuzzy phrase must be preceded by an escape character.

Explanation: Masking together with fuzzy search is not allowed.
What to do: Update the search string with an escape character.

CTE0179 Thesaurus name "%1" exceeds allowed length of "%2".

Explanation: Primary keys longer than 60 bytes are not supported.
What to do: Change the layout of the table before creating the index again.

CTE0180 Thesaurus "%1" can not be found.

Explanation: The thesaurus specified cannot be found.
What to do: Check that the thesaurus files are located in the thesaurus directory or fully qualified.
### Error messages

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Explanation</th>
<th>What to do</th>
</tr>
</thead>
<tbody>
<tr>
<td>CTE0181</td>
<td>Library &quot;%1&quot; cannot be loaded.</td>
<td>A library cannot be found.</td>
<td>Check that the library is located in the library path and available. Start and stop DB2 to ensure that the current settings are used.</td>
</tr>
<tr>
<td>CTE0182</td>
<td>Function &quot;%1&quot; cannot be loaded from library &quot;%2&quot;.</td>
<td>A library entry point cannot be loaded.</td>
<td>The library accessed seems to be invalid. Check that the library is specified only once.</td>
</tr>
<tr>
<td>CTE0183</td>
<td>Error occurred using shared system resources.</td>
<td>A request to shared system resources like shared memory or semaphores cannot be fulfilled.</td>
<td>Check the current system status and configuration. On UNIX use the <code>ipcs</code> command to check the resources. Stop all applications, such as DB2 and DB2 Net Search Extender. If further resources are listed, clean them up using <code>ipcrm</code>.</td>
</tr>
<tr>
<td>CTE0184</td>
<td>No db2text start command was issued.</td>
<td>A command was called which requires the locking and update services.</td>
<td>Start the update and locking services with <code>db2text start</code>.</td>
</tr>
<tr>
<td>CTE0185</td>
<td>The update and locking services are already active.</td>
<td>A <code>db2text</code> start is issued but the update and locking services are already running.</td>
<td>No further action.</td>
</tr>
<tr>
<td>CTE0186</td>
<td>Update and locking service error occurred, check <code>db2diag.log</code> for details.</td>
<td>An update and locking service error occurred.</td>
<td>Check the <code>db2diag.log</code> for further information, or clean up your shared resources. See also CTE0183.</td>
</tr>
<tr>
<td>CTE0187</td>
<td>Update and locking services are still active, use FORCE option to stop the services.</td>
<td>The <code>db2text stop</code> command has not stopped the locking services, there are still processes running.</td>
<td>Check with <code>db2text control</code> which processes are running and wait for those to finish. If you need to stop them, use the FORCE option.</td>
</tr>
<tr>
<td>CTE0188</td>
<td>There is a temporary problem using update and locking services. Please try again.</td>
<td>The <code>db2text stop</code> command has not stopped the locking services. Programs are still running or an inconsistent situation is found.</td>
<td>Check with <code>db2text control</code> which processes are running and wait for those to finish. To stop them, use the FORCE option.</td>
</tr>
<tr>
<td>CTE0189</td>
<td>The executable program &quot;%1&quot; cannot be found.</td>
<td>The program file cannot be located or accessed.</td>
<td>Check if the program file is located in the <code>bin</code> or <code>adm</code> directory of the DB2 server. The installation is corrupt if the file cannot be found.</td>
</tr>
<tr>
<td>CTE0190</td>
<td>The executable program &quot;%1&quot; cannot be started.</td>
<td>The program cannot be started.</td>
<td>Check if the program is located in the <code>bin</code> or <code>adm</code> directory of the DB2 server.</td>
</tr>
</tbody>
</table>
Error messages

the bin or adm directory of the DB2 server and that the appropriated libraries are installed. For further information, call the program manually on the server.

CTE0191 The drop index operation is incomplete. Check db2diag.log for details.

Explanation: The drop index operation is incomplete, possibly caused by the FORCE option.

What to do: Using the FORCE option drops everything regardless of any errors. Check the index directory for pending files and remove these manually.

CTE0192 Errors occurred in an update index operation. Check event table "%1". "%2" and db2diag.log for details.

Explanation: During the index update process, any document errors are written to the event table.

What to do: Check the event table for more information about the document errors. Clean up the event log after the problems have been fixed.

CTE0194 The type "%1" of column "%2" is not supported.

Explanation: You used a column that is not in the list of the supported ones.

What to do: Check create index for a list of valid columns for Keys and Indexing. Make the appropriate changes to the command and try again.

CTE0195 "%1" is not an absolute path.

Explanation: An absolute path on the server is required.

What to do: Check the path and write an absolute path in the command.

CTE0198 No corresponding text index.

Explanation: There is no text index on the column.

What to do: Check if the text index still exists.

CTE0199 There is no text index corresponding to column "%1" of table "%2".

Explanation: You tried to search on a column without a text index.

What to do: Check the column you are searching on, or create a text index on the column.

CTE0200 At least one command option must be specified.

Explanation: The ALTER INDEX command changes the characteristics of an index, such as the update and storage options. None of the characteristics to be changed was specified.

What to do: Specify at least one command option. Refer to the command syntax for all possible options.

CTE0201 There is a conflict with an existing text index on the same column.

Explanation: A text index defined on the same column was created with different parameters from this create index command.

What to do: Correct the parameter values in the create index command. Make sure that following parameters have the same value for the existing index and the index to be created: ccsid, language, format, document model, index configuration, column function, and attributes.

CTE0202 The object "%1","%2" must be a view when key columns are specified.

Explanation: The specified object is not a view. The KEY COLUMNS FOR INDEX ON VIEW
clause is only allowed when indexing a column of a view.

What to do: Remove the KEY COLUMNS FOR INDEX ON VIEW(SQL-columnname-list) clause.

CTE0203 The text index "%1"."%2" was not created with the CACHE TABLE option. This is required for command execution.

Explanation: This command can only be executed if the specified index was created with the CACHE TABLE option.

What to do: Create an index with CACHE TABLE option. Refer to the documentation for the command syntax.

CTE0204 An attribute name is missing. Add "AS <attribute name>" to the attribute expression.

Explanation: Whenever a column expression is used in the attribute expression, an attribute name must be supplied. For example: (C1+C2 AS myname).

What to do: Add "AS <attribute name>" to the attribute expression.

CTE0205 CACHE TABLE expressions are not valid.

Explanation: The column list in the cache table expression is not valid.

What to do: Correct the cache table column list in the create index command. Make sure the columns exist in the specified table. If a function is applied on a column, verify that it is used correctly.

CTE0206 ATTRIBUTE expressions are not valid.

Explanation: The column list in the attribute expression is not valid.

What to do: Correct the attribute column list in the create index command. Make sure the columns exist in the specified table. If a function is applied on a column, verify that it is used correctly.

CTE0207 KEY COLUMNS FOR INDEX ON VIEW not specified for index on view "%1"."%2".

Explanation: If indexes on views are created, the KEY COLUMNS FOR INDEX ON VIEW(SQL-columnname-list) clause must be specified. The list of column names specifies the columns that UNIQUELY identify a row in the view.

What to do: Include the KEY COLUMNS FOR INDEX ON VIEW(SQL-columnname-list) clause in the create index command.

CTE0208 INITIAL SEARCH RESULT ORDER columns are not valid.

Explanation: The column list in the INITIAL SEARCH RESULT ORDER(SQL-order-by list) expression is not valid.

What to do: Correct the order by column list in the create index command. Check if the syntax is correct and the columns exist in the specified table. If a function is applied on a column, verify that it is used correctly.

CTE0209 The type "%1" of attribute column "%2" is not supported, type DOUBLE is required.

Explanation: For attribute columns, the only supported data type is DOUBLE.

What to do: Make sure the attribute columns of the table with the text column to be indexed are of type DOUBLE. It may be possible to use cast operators in attribute column expressions. Refer to the SQL Reference for data types which can be casted to double.

CTE0210 The value "%1" for index configuration parameter "%2" is not valid. A valid value is "%3".

Explanation: The specified value for the configuration parameter is incorrect. For valid
values of the parameters refer to the command syntax.

What to do: Correct the index configuration parameter value in the create index command.

CTE0211 "%1" is not a valid index configuration parameter.

Explanation: The index configuration option is not known.

What to do: Check the create index command syntax. Valid index configuration options are TreatNumbersAsWords and IndexStopWords. These have to be comma separated: index configuration(treatnumbersaswords 1, indexstopwords 1).

CTE0212 Internal index configuration file "%1" could not be saved.

Explanation: The internal configuration file for the index could not be saved.

What to do: Make sure the instance owner has write permissions to the directory the file should be saved in. If a file with the same name already exists, make sure that it is writable for the instance owner.

CTE0213 Internal index configuration file template "%1" could not be loaded.

Explanation: The internal index configuration file template could not be read.

What to do: Make sure the file exists in the correct location and is readable.

CTE0214 Internal error when setting new entry "[%1],%2=%3" for index configuration file.

Explanation: Internal error while writing an internal configuration file for the index.

What to do: If the file exists, check if it is readable and writable for the instance owner. Check that there is enough space on the device where the file is located.

CTE0215 Index creation on alias "%1","%2" is not supported. Use base table "%3","%4" instead.

Explanation: The index cannot be created on the alias.

What to do: Type in the create index command with the base table.

CTE0217 The schedule service is already active.

Explanation: The service is already active, you do not need to start it.

What to do: No action required.

CTE0218 Function "%1" failed with error code "%2".

Explanation: A Windows function failed with the specified error code which does not allow further processing.

What to do: Use the specified Windows system error code to get detailed error information.

CTE0219 The service "%1" could not be opened. Error code "%2".

Explanation: The specified service cannot be found on the Windows system.

What to do: Check if the specified service is installed on the Windows system. Use the specified Windows system error code to get detailed error information.

CTE0220 The DB2 instance profile path could not be found.

Explanation: Internal DB2 function to obtain the DB2 instance profile path failed.

What to do: Create a DB2 instance without specifying the instance profile path information and retry the command.
Error messages

CTE0221  UpdateFrequency "%1" is incorrectly specified.
Explanation: The syntax for the update frequency statement is not correct.
What to do: Correct the update frequency statement according to the syntax specification.

CTE0222  The schedule service input file "%1" is corrupted.
Explanation: The scheduler file containing index update information is corrupted.
What to do: Use your system editor and try to correct the problem. Maybe an entry has been truncated, or the ending line character has been deleted. If this does not restore the file content, try the following:
• Call command db2text stop to stop the scheduler.
• Delete the scheduler service file.
• Call command db2text start to start the scheduler.
• Use command db2text alter index ... to recreate the update frequency entries for all concerned indexes.

CTE0223  File "%1" could not be closed.
Explanation: The file specified cannot be closed.
What to do: Verify that the file is specified correctly.

CTE0224  File "%1" could not be copied to "%2".
Explanation: The first file cannot be copied to the second file.
What to do: Verify that the files are correctly specified. Check if the second file already exists and is read only. Also check if there is enough free space on the system.

CTE0225  File "%1" could not be removed.
Explanation: The file specified cannot be removed from the system.

What to do: Verify that the file is specified correctly and check the file access rights.

CTE0227  A write operation on file "%1" failed.
Explanation: The file specified is not writable.
What to do: Verify that the file is correctly specified and check the file access rights. Also check if there is enough free space on the system.

CTE0231  "%1" is not defined in same nodedgroup ("%4") as the tablespace of "%2", "%3".
Explanation: The tablespace of the administration tables is required to be distributed over different nodes in exactly the same way as the table containing the text column to be indexed. To enforce this, it is checked whether the specified tablespace is defined in the same nodedgroup.
What to do: Specify a tablespace that is defined in the same nodedgroup as the table containing the text column to be indexed.

CTE0232  The specified or default tablespace "%1" is not single-noded. This is necessary for an index on a view, or when the CACHE TABLE option is specified.
Explanation: An index on a view or with the CACHE TABLE option enabled is only supported for tables on a single node.
What to do: Put the table in a single-noded tablespace if the default tablespace caused this error. Alternatively, specify another single-noded tablespace, if you specified a multi-noded tablespace.

CTE0233  There is a conflicting administration command running. Please retry this command later.
Explanation: Another administration command is still running or terminated abnormally without
Error messages

releasing the command lock.

What to do: Check with CONTROL LIST which locks are still active. If there is an active lock but no command running, clear the lock manually using the CONTROL CLEAR command. Be aware that someone else may be running the administration command holding the lock.

CTE0234 There is a conflicting administration command running on a text index. Please retry this command later, or specify the FORCE option of a DISABLE DATABASE command.

Explanation: Another administration command is still running or terminated abnormally without releasing the command lock.

What to do: Check with CONTROL LIST which locks are still active. If there is an active lock but no command running, clear the lock manually using the CONTROL CLEAR command. Be aware that someone else may be running the administration command holding the lock. For a DISABLE DATABASE command you may specify the FORCE option which stops all other commands on that database.

CTE0235 No valid license found for DB2 Net Search Extender.

Explanation: There was no valid license found for DB2 Net Search Extender.

What to do: Check whether the license was correctly installed with db2lic. Make sure existing instances are updated after the product install.

CTE0236 Only Node0 is supported on MPP instances.

Explanation: Text Indexes can only be created on MPP instances, if the table with the text column to be indexed resides on Node0.

What to do: Check the node group of the table space in which the table is defined.

CTE0237 Internal error: log table “%1”. “%2” contains an invalid operation “%3”.

Explanation: The log table keeps track of operations executed on the table containing the indexed text column. This table might be corrupted, as it contains an entry not written by DB2 Net Search Extender.

What to do: Check the log table and delete the corrupted entry.

CTE0238 Internal error: table “%1”. “%2” contains an incorrect syntax expression in column “%3”.

Explanation: There is an error in the expression list in the specified text column.

What to do: Check the delimiter Begin and End pairs.

CTE0239 Internal error: total length of index properties “%1” exceeds maximum “%2”.

Explanation: The maximum size of the index properties (1016 bytes) is exceeded. The properties contain the instance, index, and work directory as well as other information.

What to do: Make sure these path names are not too long.

CTE0240 Internal error: setting environment variable “%1” failed.

Explanation: Setting the specified environment variable failed. There may be a problem with the environment setup.

What to do: Check your OS specific guidelines.

CTE0241 Internal error: datalink UDF “%1”. “%2” returns type “%3”. The expected type is: “%4”.

Explanation: The datalink UDF used to fetch the content of a datalink reference is defined with an unexpected datatype.
**What to do:** Extract the definition of the datalink UDF from SYSCAT.FUNCTIONS and report the error to IBM services.

**CTE0242**  Value "%1" for parameter "%2" is invalid.

**Explanation:** The search stored procedure or the table valued function DB2EXT.TEXTSEARCH was called with invalid parameters.

**What to do:** Correct the parameter values of the search stored procedure or table valued function. For valid parameters refer to the documentation.

**CTE0243**  The cache for text index "%1"."%2" has not been activated.

**Explanation:** A Net Search Extender operation requires an activated cache. The cache is currently not activated. These are the possible reasons:

- The cache has never been activated after the last DB2TEXT START command.
- The cache has been explicitly deactivated with the DB2TEXT DEACTIVATE CACHE command.

**What to do:** Perform a DB2TEXT ACTIVATE CACHE command for the index and rerun the Net Search Extender operation.

**CTE0244**  Internal error: call to "%1" returns rc="%2", SQLCODE="%3".

**Explanation:** An internal processing error occurred when calling an internal function.

**What to do:** If the error persists, start a trace and check the db2diag.1og. Report the error.

**CTE0245**  The requested cache size exceeds the available cache size. Increase the maximum cache size to a value > "%1" or decrease the pctfree value.

**Explanation:** The cache size necessary to load all data exceeds the MAXIMUM CACHE SIZE value for an index. This can be detected during activation of the cache (the DB2TEXT ACTIVATE command), or by an index update operation while the cache is activated.

**What to do:** If the error was reported in a DB2EXT ACTIVATE command, recalculate the maximum cache size using the DB2EXT.MAXIMUM_CACHE_SIZE function and alter the MAXIMUM CACHE SIZE setting for the index. Eventually decrease the PCTFREE value. If the maximum number of documents is exceeded during incremental update, rebuild the cache with the commands db2 deactivate cache and db2text activate cache recreate.

**CTE0246**  File "%1" is empty.

**Explanation:** A DB2TEXT CREATE INDEX command failed because the document model file specified in the command is empty.

**What to do:** Specify a valid document model file in the command.

**CTE0247**  A DB2 Net Search Extender stored procedure could not be created.

**Explanation:** A DB2TEXT ENABLE DATABASE command failed to create the internal stored procedure DB2EXT.CTESRVSP.

**What to do:** Check the additional DB2 error message associated with a CREATE PROCEDURE statement for details. If the error cannot be corrected by removing an existing stored procedure with an identical name, start a trace and report the error.

**CTE0248**  The generated search string is too long. Reduce the complexity of search query.

**Explanation:** A Net Search Extender query is too long or too complex to be processed by the base search engine. The complexity is affected by thesaurus expansions, FUZZY FORM OF expressions, and masking characters.

**What to do:** Reduce complexity or length of the query.
## Error messages

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
</table>
| CTE0249    | Executable program "%1" terminated abnormally.  
**Explanation:** When executing a Net Search Extender command, the executable "%1" was called, but terminated abnormally.  
**What to do:** Verify, that the executable was not terminated explicitly by user interaction, for example, a signal. If not, start a trace, rerun the command, and report the error. |
| CTE0250    | The return type "%1" of column type transformation function "%2","%3" is not supported.  
**Explanation:** In a DB2TEXT CREATE INDEX command a column type transformation was specified that returns an unsupported datatype. Supported datatypes are: CHARACTER, VARCHAR, LONG VARCHAR, CLOB, GRAPHIC, VARGRAPHIC, LONG VARGRAPHIC, DBCLOB, BLOB, and DATALINK.  
**What to do:** Choose a different column type transformation function. |
| CTE0251    | Internal error: the column type "%1" is not supported.  
**Explanation:** A column type is used that is not in the list of supported types.  
**What to do:** Check create index for a list of valid columns for Keys and Indexing. Make the appropriate changes to the command and try again. If the error persists, start a trace and also check the db2diag.log. Report the error to IBM Services. |
| CTE0252    | The parameter "%1" is missing.  
**Explanation:** Internal error - when executing a Net Search Extender command, an administration executable program was called with a missing parameter "%1".  
**What to do:** Try to change Net Search Extender parameter commands to avoid the problem. If the error persists, switch on the trace function and report the error to IBM Services. |
| CTE0253    | The document listed in the log view was not found.  
**Explanation:** The contents of a text document that is listed in the log view has changed and could not be accessed.  
**What to do:** Check that the document exists and the read/access permissions of the text documents to be included in the index. |
| CTE0254    | The cache for index "%1" is already activated.  
**Explanation:** The index has already been activated with the ACTIVATE CACHE command.  
**What to do:** Check the specified index name and the database that you are using. |
| CTE0255    | A column name for a cache result column expression is missing. Add "AS <cache column name>" to the expression.  
**Explanation:** A cache result column expression must be named. For example: 'C1+C2 AS myresult'.  
**What to do:** Add "AS <cache column name>" to the expression. |
| CTE0256    | The query necessary to select data for indexing failed. Reduce the complexity of the attribute, cache table, or the initial search result order expressions.  
**Explanation:** Net Search Extender creates a query from the expressions in your command to select data for indexing from the database. The query failed because it was too complex.  
**What to do:** Reduce the complexity of attribute, cache table, or initial search result order expressions. |
| CTE0257    | Error creating shared memory.  
**Explanation:** The shared memory resource could not be created due to a previous error or permission problem. |
What to do: Check db2diag.log for further information, or clean up your shared resources. See also error CTE0183.

CTE0258  |  Shared memory version error.
Explanation: The shared memory resource could not be accessed because it is corrupted or there is a version conflict.
What to do: Check db2diag.log for further information. Disable and re-enable the database and then try again.

CTE0259  |  Cannot insert entry in global shared memory. Entry already exists.
Explanation: An entry to be inserted in global shared memory already exists because of a previous error.
What to do: Check db2diag.log for further information. Restart the update and locking services using the commands db2text stop and db2text start.

CTE0260  |  Cannot access entry in global shared memory. Entry not found.
Explanation: An entry to be removed from global shared memory does not exist because of a previous error.
What to do: Check db2diag.log for further information. Try to restart the update and locking services using the commands db2text stop and db2text start.

CTE0261  |  There is at least one cache activated for a text index in this instance. Deactivate the cache for any activated index using the DEACTIVATE CACHE command, or use the FORCE option to stop.
Explanation: The db2text stop command can only be used if you run a DEACTIVATE CACHE command for all text indexes that have been activated with the ACTIVATE CACHE command.
What to do: Deactivate the cache for any

CTE0262  |  The value for parameter "%1" is too long.
Explanation: The value exceeds the maximum allowable size.
What to do: Check the maximum size.

CTE0263  |  The text index "%1","%2" was created with the RECREATE INDEX ON UPDATE option. In this context, the UPDATE MINIMUM or COMMITCOUNT FOR UPDATE may not be specified.
Explanation: Update minimum and commitcount for update are only effective if the index is updated incrementally.
What to do: If you want to recreate the index each time an update is performed, remove the UPDATE MINIMUM and COMMITCOUNT FOR UPDATE settings. If you want to use UPDATE MINIMUM and COMMITCOUNT FOR UPDATE, do not specify RECREATE INDEX ON UPDATE.

CTE0264  |  Errors occurred in an activate index operation. Check event view "%1","%2" and the db2diag.log for details.
Explanation: During the index activate process, errors are written to the event table and the db2diag.log file.
What to do: Check the event table for more information about the document errors. Clean up the event log after the problems have been fixed.

CTE0265  |  The tablespace of a user table or administration tablespace ("%1") is not only defined on node 0.
Explanation: If text indexes are created on MPP instances, the tablespace of the user table must only reside on Node0.
Error messages

What to do: Use a table where the tablespace resides on Node0.

CTE0266  ValueFrom "%1" must be smaller than ValueTo "%2".

Explanation: The values specified in the attribute search are not valid. If the search syntax is 'BETWEEN ValueFrom AND ValueTo', the lower boundary (ValueFrom) must be smaller than upper boundary(ValueTo).

What to do: Change the boundaries in the 'BETWEEN ValueFrom AND ValueTo' clause.
Appendix G. Document model reference

DB2 Net Search Extender provides the following reference information for document models:

- The DTD for document models
- The semantics of locator (XPath) expressions
- Limitation for text fields and document attributes

DTD for document models

Here is a formal description of the syntax of document models in the form of a document type definition (DTD):

```xml
<!ELEMENT GPPModel (GPPFieldDefinition|GPPAttributeDefinition)+>
<!ELEMENT HTMLModel (HTMLFieldDefinition|HTMLAttributeDefinition)+>
<!ELEMENT XMLModel (XMLFieldDefinition|XMLAttributeDefinition)+>

<!ELEMENT GPPFieldDefinition EMPTY>
<!ATTLIST GPPFieldDefinition name CDATA #REQUIRED>
<!ATTLIST GPPFieldDefinition start CDATA #REQUIRED>
<!ATTLIST GPPFieldDefinition end CDATA #IMPLIED>
<!ATTLIST GPPFieldDefinition exclude (YES|NO) NO>

<!ELEMENT GPPAttributeDefinition EMPTY>
<!ATTLIST GPPAttributeDefinition name CDATA #REQUIRED>
<!ATTLIST GPPAttributeDefinition start CDATA #REQUIRED>
<!ATTLIST GPPAttributeDefinition end CDATA #REQUIRED>
<!ATTLIST GPPAttributeDefinition type NUMBER #REQUIRED>

<!ELEMENT HTMLFieldDefinition EMPTY>
<!ATTLIST HTMLFieldDefinition name CDATA #REQUIRED>
<!ATTLIST HTMLFieldDefinition tag CDATA #REQUIRED>
<!ATTLIST HTMLFieldDefinition meta-qualifier CDATA #IMPLIED>
<!ATTLIST HTMLFieldDefinition exclude (YES|NO) NO>

<!ELEMENT HTMLAttributeDefinition EMPTY>
<!ATTLIST HTMLAttributeDefinition name CDATA #REQUIRED>
<!ATTLIST HTMLAttributeDefinition tag CDATA #REQUIRED>
<!ATTLIST HTMLAttributeDefinition meta-qualifier CDATA #IMPLIED>
<!ATTLIST HTMLAttributeDefinition type NUMBER #REQUIRED>

<!ELEMENT XMLFieldDefinition EMPTY>
<!ATTLIST XMLFieldDefinition name CDATA #REQUIRED>
<!ATTLIST XMLFieldDefinition locator CDATA #REQUIRED>
<!ATTLIST XMLFieldDefinition ignore (YES|NO) NO>
<!ATTLIST XMLFieldDefinition priority CDATA #IMPLIED>
<!ATTLIST XMLFieldDefinition exclude (YES|NO) NO>

<!ELEMENT XMLAttributeDefinition EMPTY>
```
The semantics of locator (XPath) expressions

According to the XML data model, XML documents are viewed as trees containing these kinds of nodes:

- The root node
- Element nodes
- Text nodes
- Attribute nodes
- Namespace nodes
- Processing instruction nodes
- Comment nodes

The links between those nodes, in other words the tree-forming relationship, reflect the immediate containment relationship in the XML document.

The root node can appear only at the root and nowhere else in the tree. It contains, as its children, the document element and optional comments and processing instructions.

Element nodes can contain any kinds of nodes except for the root node. The other kinds of nodes are only allowed at terminal nodes of the tree.

There are three kinds of containment links: ‘child’, ‘attribute’, and ‘namespace’. The ‘attribute’ and ‘namespace’ containment links must lead to attribute and namespace nodes, respectively. In other words, to access the children of an element node (in terms of graph theory) you need to follow ‘attribute’ links to find all contained attributes, follow ‘namespace’ links to find all contained namespace declarations, and follow ‘child’ links to find contained elements, text nodes, processing instructions, and comments.

An XPath expression needs to be interpreted with respect to a context node, and denotes a set of nodes. When used as Net Search Extender selector patterns, the context node is free, that is, a relative path pattern p is interpreted as //p.

These are the Net Search Extender XPath selector patterns:
- Pattern `|` LocationPathPattern in context N denotes the union of the nodes matched by Pattern and LocationPathPattern, both in context N.
Languages

- `'/RelativePathPattern` in context N denotes whatever this RelativePathPattern denotes in the context of the root.
- `'//RelativePathPattern` in context N denotes the union of the denotations of this RelativePathPattern interpreted in any context that is a descendant (on the child axis) of the root.
- RelativePathPattern `'/' StepPattern` matches a node in context N, if and only if that node is matched by StepPattern in the context of its parent, and its parent node is matched by RelativePathPattern in context N.
- RelativePathPattern `'//' StepPattern` matches a node in context N, if and only if that node is matched by StepPattern in the context of its parent, and it has an ancestor node that is matched by RelativePathPattern in context N.
- `'child':NodeTest` (abbreviated syntax: NodeTest) in context N matches a node that is a child of N (on the child axis) and that satisfies NodeTest.
- `'attribute':NodeTest` (abbreviated syntax: @NodeTest) in context N matches a node that is an attribute of N and that satisfies NodeTest.
- `NodeType '{' '}'` is satisfied for a node if and only if it is of the specified type.
- `'processing-instruction' '{' 'Literal '}'` is satisfied for any processing-instruction-type node that has Literal as its name.
- `'*'` is satisfied for any element or attribute node (name mask for element name).
- `NCName ':' '*'` is satisfied for any element node that has NCName as its name prefix.
- QName is satisfied for any node with the specified name.

Note

A NodeTest of the form NameTest assumes the node to be of the principal type on the selected axis, which is attribute type on the attribute axis and child type on the child axis. Consequently, NameTest cannot be used to choose comments or processing instruction nodes, but only child and attribute nodes. Moreover, the patterns allow for the selection of any kind of node, except for namespace nodes, because the axis specifier 'namespace' is not allowed.

Examples of patterns:
- chapter | appendix denotes all chapter elements and appendix elements
- table denotes all table elements
- * denotes all elements (note this is the abbreviation of child::*
- ulist/item denotes all item elements that have a ulist parent

Appendix G. Document model reference 219
**Languages**

- `appendix//subsection` denotes all subsection elements with an appendix ancestor
- `.` denotes the singleton set containing just the root node
- `comment()` denotes all comment nodes
- `processing-instruction()` denotes all processing instructions
- `attribute::* (or @*)` denotes all attribute nodes

This is the syntax of the locator element:

```
Locator ::= LocationPathPattern
| Locator '|' LocationPathPattern
LocationPathPattern ::= '/' RelativePathPattern ?
| '//'? RelativePathPattern
RelativePathPattern ::= StepPattern
| RelativePathPattern '/' StepPattern
| RelativePathPattern '//' StepPattern
StepPattern ::= ChildOrAttributeAxisSpecifier NodeTest
ChildOrAttributeAxisSpecifier ::= ('child' | 'attribute') '::'
| '0'? NodeTest ::= NameTest
| 'processing-instruction' '(' Literal ')' NameType ::= '*' | NCName ':' '*' | QName
| 'processing-instruction'
```

NCName and QName are as defined in the XML Names Recommendation:

---

**Limitations for text fields and document attributes**

Here is a list of the limitations for text fields and document attributes:

- Maximum number of fields in an index: 32767
- Maximum number of values for one attribute of type STRING in one document: 1024
- Maximum number of attributes of type STRING: 253
- Number of characters in a STRING attribute value is truncated to 128
- Maximum number of attributes of types DATE and NUMBER: 32766
- Number of characters in a DATE or NUMBER attribute value is truncated to 128
- For NUMBER attributes, a double precision floating point number is accepted as a value.
- Maximum number of values that can be specified for one attribute of type DATE or NUMBER in one document: unlimited

These are the tags that can be included in an HTML document model:

- `<A>`
Languages

• <ADDRESS>
• <AU>
• <AUTHOR>
• <H1>
• <H2>, <H3>, <H4>, <H5>
• <H6>
• <TITLE>

Tags like <HEAD> and <BODY> that can contain other tags, cannot be specified in an HTML document model as a text field.
Appendix H. Text Search Engine

DB2 Net Search Extender provides the following Text Search Engine information:

- Tokenization
- Stopwords

Tokenization

During indexing, Net Search Extender processes document text in the following way, breaking the text up into tokens.

Words

All alphanumeric characters ("a".."z","A".."Z", "0".."9") are used to create the full-text index. Separation characters are blank characters and the characters described in the sentence recognition section below. Control characters, such as line feed (also known as a new line character) and blank characters, are interpreted as follows: Control characters (less than 0x20) in the middle of the line are regarded as blank characters. Blank characters and control characters before and after a line feed (0x0A) are ignored. Line feed before and after a 1-byte character are regarded as blank characters and 2-byte characters for the same character are always regarded as the same characters. Capital letters and small letters for the same character, for example, "A" and "a", are regarded as the same characters if nothing is specified during search, or as different characters if exact matching is required during search.

Sentences

Net Search Extender recognizes ".", "!", "?", "." followed by blank characters, and the Japanese and Chinese full-stop at the end of a line as the end of a sentence.

Paragraphs

Paragraph recognition is dependent on the document format. In Plain Text format, any two consecutive new line characters (possibly with an intervening carriage return) are recognized as a paragraph boundary. In HTML, the paragraph tag <p> is interpreted as paragraph boundary. The other document formats do not support paragraph recognition.

Stopwords

Stopwords are words with a high frequency and no relevant content for the text retrieval process. Usually, all function words (in a linguistic sense) are considered stopwords, for example "and", "or", and "in".
Text Search Engine

Net Search Extender provides stopword processing for a list of languages, where the stopwords are not indexed and therefore, cannot be searched on. However, the result of stopword processing is a smaller and faster text index.

Note that stopwords that are not indexed are processed the same way as normal words during search. However, if a stopword has been indexed, the stopword is ignored during the search process.

Languages supporting stopwords

The following languages provide stopword processing.

<table>
<thead>
<tr>
<th>Language</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AR-AA</td>
<td>Arabic as spoken in Arabic countries</td>
</tr>
<tr>
<td>CA_ES</td>
<td>Catalan as spoken in Spain</td>
</tr>
<tr>
<td>DA_DK</td>
<td>Danish as spoken in Denmark</td>
</tr>
<tr>
<td>DE_CH</td>
<td>German as spoken in Switzerland</td>
</tr>
<tr>
<td>DE_DE</td>
<td>German as spoken in Germany</td>
</tr>
<tr>
<td>EL_GR</td>
<td>Greek as spoken in Greece</td>
</tr>
<tr>
<td>EN_GB</td>
<td>English as spoken in the U.K.</td>
</tr>
<tr>
<td>EN_US</td>
<td>English as spoken in the U.S.</td>
</tr>
<tr>
<td>ES_ES</td>
<td>Spanish as spoken in Spain</td>
</tr>
<tr>
<td>FI_FI</td>
<td>Finnish as spoken in Finland</td>
</tr>
<tr>
<td>FR_CA</td>
<td>French as spoken in Canada</td>
</tr>
<tr>
<td>FR_FR</td>
<td>French as spoken in France</td>
</tr>
<tr>
<td>HE_IL</td>
<td>Hebrew as spoken in Israel</td>
</tr>
<tr>
<td>IS_IS</td>
<td>Icelandic as spoken in Iceland</td>
</tr>
<tr>
<td>IT_IT</td>
<td>Italian as spoken in Italy</td>
</tr>
<tr>
<td>IW_IL</td>
<td>Hebrew as spoken in Israel</td>
</tr>
<tr>
<td>NB_NO</td>
<td>Norwegian Bokmal as spoken in Norway</td>
</tr>
<tr>
<td>NL_BE</td>
<td>Dutch as spoken in Belgium</td>
</tr>
<tr>
<td>NN_NO</td>
<td>Norwegian Nynorsk as spoken in Norway</td>
</tr>
<tr>
<td>PT_BR</td>
<td>Portuguese as spoken in Brazil</td>
</tr>
<tr>
<td>PT_PT</td>
<td>Portuguese as spoken in Portugal</td>
</tr>
<tr>
<td>RU_RU</td>
<td>Russian as spoken in Russia</td>
</tr>
<tr>
<td>SV_SE</td>
<td>Swedish as spoken in Sweden</td>
</tr>
</tbody>
</table>
### Appendix I. Text Search Engine reason codes

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Operation performed successfully - no error occurred.</td>
</tr>
<tr>
<td>1</td>
<td>An invalid handle was passed to a function.</td>
</tr>
<tr>
<td>2</td>
<td>Function could not allocate enough memory.</td>
</tr>
<tr>
<td>3</td>
<td>Function could not perform due to access limitations or security restrictions.</td>
</tr>
<tr>
<td>4</td>
<td>The operation is not supported for this version of the Text Search Engine run time.</td>
</tr>
<tr>
<td>5</td>
<td>The operation is currently not enabled.</td>
</tr>
<tr>
<td>6</td>
<td>The application violated the Text Search Engine protocol by calling Text Search Engine functions in illegal order.</td>
</tr>
<tr>
<td>7</td>
<td>An unexpected error occurred. Please report this to your service representative.</td>
</tr>
<tr>
<td>8</td>
<td>An invalid language was specified.</td>
</tr>
<tr>
<td>9</td>
<td>The specified language is valid but not supported by the Text Search Engine run time.</td>
</tr>
<tr>
<td>10</td>
<td>An invalid CCSID was specified.</td>
</tr>
<tr>
<td>11</td>
<td>The specified CCSID is valid but not supported by the Text Search Engine run time.</td>
</tr>
<tr>
<td>12</td>
<td>An invalid document ID was specified.</td>
</tr>
<tr>
<td>13</td>
<td>The specified document format is valid but not supported by the Text Search Engine run time.</td>
</tr>
<tr>
<td>14</td>
<td>An invalid document format was specified.</td>
</tr>
<tr>
<td>15</td>
<td>The operation could not succeed due to access limitation during file input/output.</td>
</tr>
<tr>
<td>16</td>
<td>The operation could not succeed due to read errors during file input/output.</td>
</tr>
<tr>
<td>17</td>
<td>The operation could not succeed due to read errors during file input.</td>
</tr>
<tr>
<td>18</td>
<td>The operation could not succeed due to write errors during file output.</td>
</tr>
<tr>
<td>19</td>
<td>The operation could not succeed due to seek errors during file input/output.</td>
</tr>
</tbody>
</table>
Reason codes

20  The operation could not succeed due to tell errors during file input/output.
21  The operation could not succeed due to close errors during file input/output.
22  The operation could not succeed due to errors during rename operations.
23  The operation could not succeed due to errors during remove operations.
24  The operation could not succeed due to errors during mkdir operations.
25  One or more function arguments did have an invalid value (for example, an unexpected null pointer or an invalid enumeration type value).
26  The specified directory does not exist.
27  An unexpected Text Search Engine error occurred. Please examine the Text Search Engine error code in the error info object for further details.
28  An unexpected COS error occurred. Please report this error.
29  An attempt was made to update an empty document.
30  The specified argument is not supported for this operation.
31  The date attribute parser found an invalid value when trying to parse a date attribute.
32  The number attribute parser found an invalid value when trying to parse a number attribute.
33  Attribute name invalid, probably too long.
35  Reserve number for future use.
36  The input document contains an attribute (DATE, NUMBER, or STRING) that exceeds the length limit for attributes. The attribute text has been truncated to that limit.
38  The warning threshold as set by the user has been exceeded. As a consequence, this error has been generated.
39  The input document could not be indexed. It contains too many nested fields.
40  The limit of different attributes for one of the attribute types has been exceeded for this index.
### Reason codes

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>46</td>
<td>The iterator is not (or no longer) valid, because its list is empty or deleted.</td>
</tr>
<tr>
<td>47</td>
<td>The function is not supported for the passed kind of handle. This error occurs, for example, when trying to use itlQueryResultEntry ObtainData on a list iterator that does not represent a query result iterator.</td>
</tr>
<tr>
<td>48</td>
<td>This warning is issued if a stop word file cannot be found for the specified language and resource path.</td>
</tr>
<tr>
<td>49</td>
<td>This warning is issued if a stop word file does not contain any stop words.</td>
</tr>
<tr>
<td>50</td>
<td>This warning is issued if a stop word file does contain invalid data.</td>
</tr>
<tr>
<td>100</td>
<td>The index could not be opened because it does not exist with the specified name and/or directory.</td>
</tr>
<tr>
<td>101</td>
<td>The specified index name is not a valid index name.</td>
</tr>
<tr>
<td>102</td>
<td>The specified index directory is not a valid directory name.</td>
</tr>
<tr>
<td>103</td>
<td>The operation cannot be performed because the Text Search Engine detected a corruption of the index structure and/or index file sets.</td>
</tr>
<tr>
<td>104</td>
<td>The specified index cannot be created because it already exists with the given name and directory.</td>
</tr>
<tr>
<td>109</td>
<td>Before any other operation can be performed on this index, a rollback operation must be performed.</td>
</tr>
<tr>
<td>110</td>
<td>The index configuration file does not contain the mandatory section as specified in the error context.</td>
</tr>
<tr>
<td>111</td>
<td>The index configuration file does not contain the mandatory option as specified in the error context.</td>
</tr>
<tr>
<td>112</td>
<td>The index configuration file contains invalid data in the option as specified in the error context.</td>
</tr>
<tr>
<td>113</td>
<td>The index configuration file does not match the Text Search Engine version.</td>
</tr>
<tr>
<td>200</td>
<td>The specified document model name is not a valid model name.</td>
</tr>
<tr>
<td>201</td>
<td>The specified document model field name is not a valid field name.</td>
</tr>
<tr>
<td>202</td>
<td>The specified document model is not known.</td>
</tr>
</tbody>
</table>
Reason codes

203 The specified document model already exists and cannot be redefined.
204 Too many or too large document models have been added to the index.
205 The document model contains too many elements.
206 The document model element contains a parameter (XML attribute) that is not allowed for this type of element.
207 The document model element contains a parameter value that is not allowed for this type of parameter (XML attribute).
208 The document model element does not contain a required parameter (XML attribute), like "name".
209 The document model does not seem to be XML, or starts with an unexpected XML element.
210 The given XPath (locator value) contains an unexpected token.
211 The given XPath (locator value) contains an unexpected axis specifier (name followed by two colons).
212 The given XPath (locator value) contains an unexpected node test.
213 The document model directory file (extension .mdx) is corrupted.
214 The document model index file (extension .mox) is corrupted.
215 The document contains an XML element which is mapped to a document attribute and which contains another document attribute. The inner attribute is ignored.
216 The given parameter value is too long as a GPP or HTML tag.
217 The document model contains a duplicate field definition.
218 The document model contains a duplicate attribute definition.
300 The operation cannot be performed because the Text Search Engine detected a corruption in the index files used for document name mapping.
301 The operation cannot be performed because the Text Search Engine detected an invalid document number.
302 The operation cannot be performed because the Text Search Engine detected an invalid document identifier.
303 The operation cannot be performed because the Text Search Engine found no index entry for the document identifier.
Reason codes

304  The operation cannot be performed because the Text Search Engine found no index entry for the document number.

305  The operation cannot be performed because the Text Search Engine detected an overflow in used document numbers.

306  The document identifier that the application tried to index has appeared already in the list of documents. The Text Search Engine does not support duplicate document identifiers appearing in one indexing sequence, that is, before the update has been committed.

340  The term strength is not valid.

341  The relation number is not valid, must be in.

342  The relation type is invalid, use one of the defines described in API.

343  The phrase (term) is too long.

344  Unexpected end of file encountered while reading.

345  Version conflict detected when reading index/thesaurus files.

346  Overflow in thesaurus buffers.

347  Invalid name, probably too long a name, for file or directory.

348  Lookup did not find term (phrase) in dictionary or entry in definition file does not contain mandatory term.

349  Definition file is empty.

350  Thesaurus dictionary or definition file as specified via input parameter does not exist.

351  Syntax errors in definition file.

352  The Relationship was specified incorrectly.

352  The Relationship number was out of range.

360  An invalid single character masking was used.

361  An invalid multiple character masking was used.

362  Operator arity is smaller than number of operands given in query.

363  Operator value out of range defined by ItlEnOperator enumeration.

364  Value for rank formula out of enumeration range.

365  Number identifying proximity segment is out of range.
### Reason codes

366  Query is under construction and cannot be redefined or reset.

367  Scope given as previous search result denotes empty result.

368  Invalid call requesting to add field names before setting the first one.

369  Invalid search flag requesting an invalid comparison with index content is ignored. If, for example, a case-sensitive comparison was requested for an index that was build in a case insensitive manner, this reason code is shown in the error information.

370  Masking of strings is not supported for Thai or DBCS languages.

371  No valid query input. For example, the search terms is available.

372  Invalid comparison operations requested.

373  Invalid comparison operations requested.

374  Search index handle was requested for an empty index.

375  The combination of operator and requested operator mode is not supported.

380  Search result is incomplete, search was discontinued due to threshold.

381  Index lookup revealed that query contained stopwords.

401  The operation cannot be performed because the Text Search Engine detected a corruption in the index files used for field/attribute name mapping.

402  The operation cannot be performed because the Text Search Engine detected an invalid field or attribute name.

403  The operation cannot be performed because the given field or attribute name is unknown.

404  The limit of different attributes for one of the attribute types or of different fields has been exceeded for this index.

500  The document/data contains an invalid character sequence (in a UTF8, UTF16, or DBCS source).

501  The code page converter was in error.

502  The document/data contains an incomplete character sequence (in a UTF8, UTF16, or DBCS source).

503  The code page converter has an invalid descriptor.
<table>
<thead>
<tr>
<th>Reason Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>600</td>
<td>The XML document contains an asynchronous entity. For example, an unquoted XML attribute value.</td>
</tr>
<tr>
<td>602</td>
<td>Invalid character reference, (for example, or).</td>
</tr>
<tr>
<td>603</td>
<td>Invalid binary entity reference.</td>
</tr>
<tr>
<td>604</td>
<td>XML Parser Expat could not be created.</td>
</tr>
<tr>
<td>605</td>
<td>An attribute name in tag must be unique.</td>
</tr>
<tr>
<td>607</td>
<td>XML Parser found an invalid external entity reference.</td>
</tr>
<tr>
<td>608</td>
<td>Documents includes an incorrect token, like missing a &lt; or &gt;.</td>
</tr>
<tr>
<td>609</td>
<td>XML documents must have an enclosing tag, and after this enclosing end tag no text is allowed.</td>
</tr>
<tr>
<td>610</td>
<td>A processing instruction is not allowed at its position. For example, the first processing instruction is not the prolog &lt;?xml .. ?&gt;.</td>
</tr>
<tr>
<td>611</td>
<td>An element is a sequence of start tag, content, and end tag. This error occurred, for example, from the sequence &quot;&lt;s&gt; text /s&gt;&quot;, because the end tag is not correct.</td>
</tr>
<tr>
<td>612</td>
<td>Memory allocation failed in XML parser.</td>
</tr>
<tr>
<td>614</td>
<td>Invalid parameter entity reference.</td>
</tr>
<tr>
<td>615</td>
<td>A non-complete character, maybe only the first byte of a 2-byte UTF8 character.</td>
</tr>
<tr>
<td>616</td>
<td>Recursive entity reference.</td>
</tr>
<tr>
<td>617</td>
<td>XML Syntax error; for example text outside the enclosing start and end tag.</td>
</tr>
<tr>
<td>618</td>
<td>Every start tag needs a matching end tag.</td>
</tr>
<tr>
<td>619</td>
<td>Unclosed cdata section.</td>
</tr>
<tr>
<td>620</td>
<td>Unclosed token; for example text after the last token in a document.</td>
</tr>
<tr>
<td>621</td>
<td>There is an entity in the document that could not be resolved.</td>
</tr>
<tr>
<td>622</td>
<td>Unexpected error.</td>
</tr>
<tr>
<td>631</td>
<td>Could not parse field or attribute information in meta-tag. Tag must have the format &lt;meta name=&quot;abc&quot; content=&quot;xyz&quot;&gt;; maybe attributes name or content of the meta-tag not correct.</td>
</tr>
<tr>
<td>632</td>
<td>The entity could not be transformed to a character.</td>
</tr>
<tr>
<td>650</td>
<td>Different field definitions begin with the same start tag.</td>
</tr>
<tr>
<td>Reason code</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td>651</td>
<td>One start tag contains another, so the tags are ambiguous.</td>
</tr>
<tr>
<td>652</td>
<td>If a field and an attribute use the same start tag, then they must use the same end tag or both no end tag.</td>
</tr>
<tr>
<td>653</td>
<td>A field not yet closed if the document ends.</td>
</tr>
<tr>
<td>654</td>
<td>No document model is specified for the structured format. The document will be parsed as plain text document without field or attribute informations.</td>
</tr>
<tr>
<td>670</td>
<td>The operation could not be performed, because it requires the &quot;Outside In&quot; (TM) libraries, which could not be found.</td>
</tr>
<tr>
<td>671</td>
<td>The operation could not be performed, because a required procedure from the &quot;Outside In&quot; (TM) libraries could not be loaded. Probably the libraries are outdated or corrupted.</td>
</tr>
<tr>
<td>672</td>
<td>An error occurred while the document was processed with &quot;Outside In&quot;.</td>
</tr>
</tbody>
</table>
Appendix J. Troubleshooting

DB2 Net Search Extender provides the following information for tracing faults.

**Tracing faults**

If you need to report an error to an IBM representative, you may be asked to switch on tracing so that information can be written to a file that can be used for locating the error.

As system performance is affected when tracing is switched on, only use the trace facility when directed by an IBM Support Center representative, or by your technical support representative.

To turn tracing on, use the DB2 facility:

```
db2trc on
```

See the *DB2 UDB Command Reference* documentation for further information.

To receive information specific to Net Search Extender, a mask with component in 96 can be used:

```
db2trc on -m *.*.96.*.*
```

In the case of severe errors, it may also help to look in *db2diag.log*. 
Appendix K. Data Link messages

Errors from the Data Link will not lead to Net Search Extender errors and, therefore, a termination of the indexing process. Ensure to check the event log and take care of not indexed documents. You need to manually take care that those are reindexed if required.

Table 15. Data Link warning messages

<table>
<thead>
<tr>
<th>Number</th>
<th>Data Link Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>01H90=</td>
<td>CTEDL - Error setting the return Blob value.</td>
</tr>
<tr>
<td>01H91=</td>
<td>CTEDL - DataLink I/O Operation timed out.</td>
</tr>
<tr>
<td>01H92=</td>
<td>CTEDL - The character encoding is not supported.</td>
</tr>
<tr>
<td>01H93=</td>
<td>CTEDL - Unsupported DataLink scheme.</td>
</tr>
<tr>
<td>01H94=</td>
<td>CTEDL - Error creating instance of datatype BLOB.</td>
</tr>
<tr>
<td>01H95=</td>
<td>CTEDL - Error setting proxy information.</td>
</tr>
<tr>
<td>01H96=</td>
<td>CTEDL - UNC scheme only valid on Windows but the OS could not be determined.</td>
</tr>
<tr>
<td>01H97=</td>
<td>CTEDL - UNC scheme only valid on Windows OS.</td>
</tr>
<tr>
<td>01H98=</td>
<td>CTEDL - DFS scheme only valid on AIX.</td>
</tr>
<tr>
<td>01H99=</td>
<td>CTEDL - DFS scheme only valid on AIX but the OS could not be determined.</td>
</tr>
<tr>
<td>01H01=</td>
<td>CTEDL - Error determines the port number in the url string.</td>
</tr>
<tr>
<td>01H02=</td>
<td>CTEDL - Unknown Datalink scheme detected.</td>
</tr>
<tr>
<td>01H03=</td>
<td>CTEDL - Could not establish connection.</td>
</tr>
<tr>
<td>01H00=</td>
<td>CTEDL - Error during execution of DataLink UDF.</td>
</tr>
<tr>
<td>01H80=</td>
<td>CTEDL - The DataLink file &quot;[0]&quot; could not be found.</td>
</tr>
<tr>
<td>01H81=</td>
<td>CTEDL - Unauthorized DataLink file access to &quot;[0]&quot;.</td>
</tr>
<tr>
<td>01H82=</td>
<td>CTEDL - Unexpected end of file or end of stream reached for &quot;[0]&quot;.</td>
</tr>
<tr>
<td>01H83=</td>
<td>CTEDL - DataLink file &quot;[0]&quot; is not readable.</td>
</tr>
<tr>
<td>01H85=</td>
<td>CTEDL - DataLink URL scheme &quot;[0]&quot; requires file name.</td>
</tr>
<tr>
<td>01H86=</td>
<td>CTEDL - No connection to DataLink file server &quot;[0]&quot; established.</td>
</tr>
<tr>
<td>01H60=</td>
<td>CTEDL - Bad HTTP Request - malformed DataLink URL syntax.</td>
</tr>
<tr>
<td>01H61=</td>
<td>CTEDL - Unauthorized DataLink request - user authentication required.</td>
</tr>
<tr>
<td>01H62=</td>
<td>CTEDL - DataLink access requires payment.</td>
</tr>
<tr>
<td>01H63=</td>
<td>CTEDL - Forbidden access for DataLink URL.</td>
</tr>
</tbody>
</table>
## Data link messages

Table 15. *Data Link warning messages (continued)*

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>01H64=</td>
<td>CTEDL - File not found on the DataLink server.</td>
</tr>
<tr>
<td>01H65=</td>
<td>CTEDL - The requested method is not allowed for the DataLink resource.</td>
</tr>
<tr>
<td>01H66=</td>
<td>CTEDL - Request not acceptable.</td>
</tr>
<tr>
<td>01H67=</td>
<td>CTEDL - Proxy Authentication Required.</td>
</tr>
<tr>
<td>01H68=</td>
<td>CTEDL - Client request timeout.</td>
</tr>
<tr>
<td>01H69=</td>
<td>CTEDL - Conflict with the current state of the DataLink resource.</td>
</tr>
<tr>
<td>01H10=</td>
<td>CTEDL - The DataLink resource is no longer available at the server.</td>
</tr>
<tr>
<td>01H11=</td>
<td>CTEDL - A content length must be specified to accept the request.</td>
</tr>
<tr>
<td>01H12=</td>
<td>CTEDL - The precondition given in the header field evaluated to false.</td>
</tr>
<tr>
<td>01H13=</td>
<td>CTEDL - The requested DataLink entity is too large.</td>
</tr>
<tr>
<td>01H14=</td>
<td>CTEDL - The requested DataLink URL is too long.</td>
</tr>
<tr>
<td>01H15=</td>
<td>CTEDL - Unsupported MIME type.</td>
</tr>
<tr>
<td>01H16=</td>
<td>CTEDL - Range Request not satisfiable.</td>
</tr>
<tr>
<td>01H17=</td>
<td>CTEDL - Expectation failed.</td>
</tr>
<tr>
<td>01H18=</td>
<td>CTEDL - Request to DataLink URL gets no content length information.</td>
</tr>
<tr>
<td>01H20=</td>
<td>CTEDL - HTTP response not valid.</td>
</tr>
<tr>
<td>01H70=</td>
<td>CTEDL - Internal DataLink server error.</td>
</tr>
<tr>
<td>01H71=</td>
<td>CTEDL - Functionality not supported by the DataLink server.</td>
</tr>
<tr>
<td>01H72=</td>
<td>CTEDL - Bad Gateway.</td>
</tr>
<tr>
<td>01H73=</td>
<td>CTEDL - Service unavailable - DL server temporarily overloaded or maintained.</td>
</tr>
<tr>
<td>01H74=</td>
<td>CTEDL - Gateway timeout.</td>
</tr>
<tr>
<td>01H75=</td>
<td>CTEDL - HTTP version not supported.</td>
</tr>
<tr>
<td>01H30=</td>
<td>CTEDL - Try to establish a socket connection - error in underlying protocol.</td>
</tr>
<tr>
<td>01H31=</td>
<td>CTEDL - Could not establish a route to the DataLink Server [0].</td>
</tr>
<tr>
<td>01H32=</td>
<td>CTEDL - Could not connect the socket to the remote address [0].</td>
</tr>
<tr>
<td>01H33=</td>
<td>CTEDL - Could not bind the socket to the local address.</td>
</tr>
<tr>
<td>01H34=</td>
<td>CTEDL - IP address of DataLink server [0] could not be determined.</td>
</tr>
<tr>
<td>01H35=</td>
<td>CTEDL - Unknown service exception - no MIME type support.</td>
</tr>
<tr>
<td>01H36=</td>
<td>CTEDL - Malformed URL ‘[0]’ - no supported protocol or DL URL could not be parsed.</td>
</tr>
</tbody>
</table>
# Appendix L. Thesaurus supported CCSIDs

The following CCSIDs are supported by the thesaurus:

<table>
<thead>
<tr>
<th>CCSIDs</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>819</td>
<td>Latin 1</td>
</tr>
<tr>
<td>850</td>
<td>PC Data Latin 1</td>
</tr>
<tr>
<td>874</td>
<td>Thai</td>
</tr>
<tr>
<td>932</td>
<td>Combined Japanese</td>
</tr>
<tr>
<td>943</td>
<td>Combined Japanese</td>
</tr>
<tr>
<td>950</td>
<td>Combined Traditional Chinese</td>
</tr>
<tr>
<td>954</td>
<td>Japanese</td>
</tr>
<tr>
<td>970</td>
<td>Combined Korean</td>
</tr>
<tr>
<td>1208</td>
<td>UTF 8</td>
</tr>
<tr>
<td>1250</td>
<td>Latin 2</td>
</tr>
<tr>
<td>1252</td>
<td>Latin 1</td>
</tr>
<tr>
<td>1253</td>
<td>Czech</td>
</tr>
<tr>
<td>1254</td>
<td>Turkish</td>
</tr>
<tr>
<td>1255</td>
<td>Hebrew</td>
</tr>
<tr>
<td>1256</td>
<td>Arabic</td>
</tr>
<tr>
<td>1258</td>
<td>Vietnamese</td>
</tr>
<tr>
<td>1363</td>
<td>Combined Korean</td>
</tr>
<tr>
<td>1381</td>
<td>Combined Simplified Chinese</td>
</tr>
<tr>
<td>1383</td>
<td>Chinese (simplified), combined SBCS/DBCS</td>
</tr>
<tr>
<td>1386</td>
<td>Chinese (simplified), combined SBCS/DBCS</td>
</tr>
<tr>
<td>5039</td>
<td>Japanese (combined SBCS/DBCS)</td>
</tr>
</tbody>
</table>

To compile the thesaurus definition file, see "DB2EXTTH (Utility)" on page 130.
Appendix M. Messages returned by the thesaurus tools

ADM_MSG_INVALID_CCSID
Explanation: Invalid CCSID specified. The requested code page is not supported.

ITL.THES.MSG.DEFFILE_MISSING
Explanation: Parameter error file name. The thesaurus definition file does not exist.

ITL.THES.MSG.NONAME_ERROR
Explanation: Parameter error. No thesaurus definition file name specified.

ITL.THES.MSG.PATHLEN_ERROR
Explanation: Parameter error file name. The thesaurus definition file path is too long. The path length must not exceed the maximum length supported for directory names in the operating system.

ITL.THES_MSG_NAMELEN_ERROR
Explanation: Parameter error file name. The thesaurus definition file name is too long.

ITL.THES_MSG_NO_TARGET_DIR_ERROR
Explanation: Parameter error. No target directory specified.

ITL.THES_MSG_UNEXPECTED_ERROR
Explanation: Internal unexpected error.

ITL.THES_MSG_PARAMETER_ERROR
Explanation: Internal parameter error.

ITL.THES_MSG_FILE_OPEN_ERROR
Explanation: Could not open file file name.

ITL.THES_MSG_FILE_REACHED_END
Explanation: Unexpected end of file in thesaurus definition file. There is an error in the definition file.

ITL.THES_MSG_FILE_READ_ERROR
Explanation: Could not read file file name.

ITL.THES_MSG_FILE_WRITE_ERROR
Explanation: Could not write file file name.

ITL.THES_MSG_FILE_ACCESS_ERROR
Explanation: Could not access file file name.

ITL.THES_MSG_FILE_REMOVE_ERROR
Explanation: Could not remove file file name.

ITL.THES_MSG_FILE_RENAME_ERROR
Explanation: Could not rename file file name 1 to file name 2.

ITL.THES_MSG_FILE_CLOSE_ERROR
Explanation: Could not close file file name.

ITL.THES_MSG_FILE_EOF_ERROR

ITL.THES_MSG_MEMORY_ERROR
Explanation: Memory error.

ITL.THES_MSG_BUFFER_OVERFLOW
Explanation: Buffer overflow.
### Messages returned by the thesaurus tools

<table>
<thead>
<tr>
<th>Code</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>ITL_THES_MSG_LOCKING_ERROR</td>
<td>Could not lock dictionary file name.</td>
</tr>
<tr>
<td>ITL_THES_MSG_LOCKED</td>
<td>Thesaurus dictionary dictionary name is in use.</td>
</tr>
<tr>
<td>ITL_THES_MSG_OUTFILE_EXIST</td>
<td>Output file file name already exists.</td>
</tr>
<tr>
<td>ITL_THES_MSG_DICT_INTEGRITY_ERROR</td>
<td>Integrity of dictionary dictionary name is lost.</td>
</tr>
<tr>
<td></td>
<td>The thesaurus dictionary file is corrupted.</td>
</tr>
<tr>
<td>ITL_THES_MSG_DICT_VERSION_ERROR</td>
<td>Dictionary dictionary name version error.</td>
</tr>
<tr>
<td></td>
<td>The thesaurus dictionary was created with an incompatible earlier version.</td>
</tr>
<tr>
<td>ITL_THES_MSG_DICT_NOT_EXIST</td>
<td>Thesaurus dictionary dictionary name does not exist.</td>
</tr>
<tr>
<td>ITL_THES_MSG_DICT_EXIST</td>
<td>Thesaurus dictionary dictionary name already exists.</td>
</tr>
<tr>
<td></td>
<td>Cannot be overwritten.</td>
</tr>
<tr>
<td>ITL_THES_MSG_NORMALIZE_ERROR</td>
<td>Error in normalizing a term. Error in the thesaurus definition file name at line number.</td>
</tr>
<tr>
<td>ITL_THES_MSG_INPUT_ERROR</td>
<td>Error in the thesaurus definition file name at line number.</td>
</tr>
<tr>
<td>ITL_THES_MSG_ERROR_IN_FILE</td>
<td>Error in file file name.</td>
</tr>
<tr>
<td>ITL_THES_MSG_IE_EMPTY</td>
<td>The thesaurus definition file file name is empty.</td>
</tr>
<tr>
<td>ITL_THES_MSG_IE_BLOCK_START</td>
<td>No block starting line was found in file file name at line line number.</td>
</tr>
<tr>
<td>ITL_THES_MSG_IE_REL_SYNTAX</td>
<td>Relationship is specified incorrectly in file name at line line number.</td>
</tr>
<tr>
<td>ITL_THES_MSG_IE_USER_DEF</td>
<td>Relationship is specified incorrectly in file name at line line number.</td>
</tr>
<tr>
<td>ITL_THES_MSG_IE_USER_DEF_DOMAIN</td>
<td>A relationship number is out of range in file name at line line number.</td>
</tr>
<tr>
<td>ITL_THES_MSG_IE_NO_TERM</td>
<td>No terms are defined in file name at line line number.</td>
</tr>
<tr>
<td>ITL_THES_MSG_IE_TERM_LEN</td>
<td>A thesaurus term is longer than 64 characters.</td>
</tr>
<tr>
<td>ITL_THES_MSG_IE_STRENGTH_SYNTAX</td>
<td>A strength value is specified incorrectly.</td>
</tr>
</tbody>
</table>
Messages returned by the thesaurus tools


ITL_THES_MSG_IE_STRENGTH_DOMAIN

Explanation: Strength is out of range.
Valid values are 1 - 100; the default is 100.
Messages returned by the thesaurus tools
Appendix N. Notices

IBM may not offer the products, services, or features discussed in this document in all 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 give 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:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, 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 Canada Limited
Office of the Lab Director
1150 Eglinton Ave. East
North York, Ontario
M3C 1H7
CANADA

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 information 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 may contain 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 may contain sample application programs in source language, which illustrates 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.

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. _enter the year or years_. All rights reserved.

### Trademarks

The following terms are trademarks of International Business Machines Corporation in the United States, other countries, or both.

- AIX
- DB2 Universal Database
- DB2
- IBM
- DB2 Extenders

The following terms are trademarks or registered trademarks of other companies:

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

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.
Glossary

This glossary defines terms and abbreviations used in this manual. If you do not find the term you are looking for, refer to the index or to the *Dictionary of Computing*, New York: McGraw-Hill, 1994.

A

access function. A user-provided function that converts the data type of text stored in a column to a type that can be processed by DB2 Net Search Extender.

B

Boolean search. A search in which one or more search terms are combined using Boolean operators.

C

catalog view. A view of a system table created by DB2 Net Search Extender Text for administration purposes. A catalog view contains information about the tables and columns that have been enabled for use by DB2 Net Search Extender Text.

CCSID. Coded Character Set Identifier.

code page. An assignment of graphic characters and control function meanings to all code points. For example, assignment of characters and meanings to 256 code points for an 8-bit code.

command line processor. A program called db2text that:

- Allows you to enter DB2 Net Search Extender commands
- Processes the commands
- Displays the result

count. A keyword used to specify the number of levels (the depth) of terms in the thesaurus that are to be used to expand the search term for the given relation.

D

DBCS. Double-byte character set.

disable. To restore a database to its condition before it was enabled for DB2 Net Search Extender Text by removing the items created during the enabling process.

document. See text document.

document model. The definition of the structure of a document in terms of the sections that it contains. A document model makes DB2 Net Search Extender aware of the sections within documents when indexing. A document model lists the markup tags that identify the sections. For each tag you can specify a descriptive section name for use in queries against that section. You can specify one or more document models in a document models file.

E

enable. To prepare a database for use by DB2 Net Search Extender.

escape character. A character indicating that the subsequent character is not to be interpreted as a masking character.

expand. The action of adding to a search term additional terms derived from a thesaurus.

F

format. The type of a document, such as ASCII, or HTML.
free-text search. A search in which the search term is expressed as free-form text – a phrase or a sentence describing in natural language the subject to be searched for.

function. See access function.

fuzzy search. A search that can find words whose spelling is similar to that of the search term.

H

hybrid search. A combined Boolean search and free-text search.

I

index. To extract significant terms from text, and store them in a text index.

index characteristics. Properties of a text index determining:
- The frequency with which the index is updated
- When the first index update is to occur

L

log table. A table created by DB2 Net Search Extender containing information about which text documents are to be indexed. Triggers are used to store this information in a log table whenever a document in an enabled text column is added, changed, or deleted.

M

masking character. A character used to represent optional characters at the front, middle, and end of a search term. Masking characters are normally used for finding variations of a term in a precise index.

match. The occurrence of a search term in a text document.

P

periodic indexing. Indexing at predetermined time intervals, expressed in terms of the day, hour, and minute, and the minimum number of documents names that must be listed in the log table for indexing, before indexing can take place.

R

retrieve. To find a text document using a search argument in one of DB2 Net Search Extender’s search functions.

S

SBCS. Single-byte character set.

Score. An absolute value of type DOUBLE between 0 and 1 that indicates how well a document meets the search criteria relative to the other found documents. The value indicates the number of matches found in the document in relation to the document’s size.

search argument. The conditions specified when making a search, consisting of one or several search terms, and search parameters.

T

text column. A column containing text documents.

text document. Text of type CHAR, GRAPHIC, VARGRAPHIC, LONG VARGRAPHIC, DBCLOB, VARCHAR, LONG VARCHAR, or CLOB datatypes, stored in a DB2 table.

text index. A collection of significant terms extracted from text documents. Each term is associated with the document from which it was extracted. A significant improvement in search time is achieved by searching in the index rather than in the documents themselves.

tracing. The action of storing information in a file that can later be used in finding the cause of an error.
trigger. A mechanism that automatically adds information about documents that need to be indexed to a log table whenever a document is added, changed, or deleted from a text column.

U

UDF. User-defined function.

UDT. User-defined type.

update frequency. The frequency with which a text index is updated, expressed in terms of the day, hour, and minute, and the minimum number of document names that must be listed in the log table for indexing, before indexing can take place.

user-defined type (UDT). A data type created by a user of DB2, in contrast to a data type provided by DB2 such as LONG VARCHAR.

user-defined function (UDF). An SQL function created by a user of DB2, in contrast to an SQL function provided by DB2.

W

wildcard character. See masking character.
Datalink Manager (continued)
installing the jar file 38
DB2 Control Center
activate the cache dialog 72
administration 51
Alter Index dialog 68
Cache Table panel 63
creating a text index 55
deactivate the cached table
dialog 72
disabling a database 53
Drop Index dialog 70
enabling a database 53
Index Events dialog 71
Index Status dialog 72
maintaining text indexes 67
Name panel 55
starting and stopping DB2 Net
Search Extender 52
Summary panel 66
Target panel 56
Text Properties panel 59
Update Characteristics panel 61
Update Index dialog 71
using the wizard 55
db2ext.dbdefaults view 181
db2ext.indexconfiguration view 185
db2ext.proxyinformation view 183
db2ext.textindexformats view 186
DB2EXTDL command
syntax 104
using 130
DB2TX, command line processor
syntax 93, 99, 105
using 31
DEACTIVATE CACHE command
syntax 126
using 42
deactivate the cached table
dialog 72
default document model 159
depth of terms in a thesaurus,
specifying 144
directory and file names 15
DISABLE DATABASE command
syntax 102
using 33, 53
disk space for indexes 23
document
CCSID 23
converting data types 36
format, description 23
formats supported 23
indexing 3
structure 159
document data types
binary data types 36
converting unsupported data
types 36
DATALINK data types 37
document model
default 159
document type definition 217
limitations 220
document model reference 217
document models
attribute name in search
syntax 144
description 159
modifying 159
overview 83
SECTION keyword in search
syntax 141
document types 23
DROP INDEX command
syntax 128
using 48
Drop Index dialog 70
ENABLE DATABASE command
syntax 100
using 32, 53
environment, client/server 11
escape character
using 77
EXPAND keyword 144
expanding search terms
See thesaurus externally stored data 9
INDEX Events dialog 71
INDEX Status dialog 72
index update events
deleting 47
recording 35
information catalogs 181
installation 13
installation verification 15
installing Data Links jar file 38
instance services 9, 27, 93
K
key concepts 3
key features 10
key terms 3
L
languages supported 195
locking services
CONTROL command 94
using 27
viewing 29
log table
creating 35
description 5
log table view 188
M
masking characters in a search term 76
match
in a search result 74
NUMBEROFMATCHES function 149
memory amounts 179
migration 177
N
Name panel 55
Net Search Extender
activating the cache 72
altering text index settings 46, 68
backup and restore 49
clearing index events 47, 71
creating a cache for a stored procedure search 38
creating a text index 34, 55
deactivating the cache 72
disabling a database 33, 53
displaying index status 72
dropping text indexes 48, 70
enabling a database 32, 53
instance services 27
maintaining text indexes 44, 67
messages 199
starting and stopping 27, 52
update services 30
updating text indexes 45, 71
using DB2 Control Center 51
viewing text index status 48
Net Search Extender Information catalogs
See views 181
NUMBEROFMATCHES function
text examples 74
syntax 149
O
occurrences of a search term 149
OR Boolean operator 76
overview 3
overview of DB2 Net Search Extender 3
P
panels and dialogs
activate the cache dialog 72
Alter Index dialog 68
Cache Table panel 63
deaivate the cached table dialog 72
Drop Index dialog 70
Index Events dialog 71
Index Status dialog 72
Name panel 55
Summary panel 66
Target panel 56
Text Properties panel 59
Update Characteristics panel 61
Update Index dialog 71
performance considerations
for indexing 44
for searching 82
planning 23
PRECISE FORM OF keyword 142
primary key types 125
R
recognition
paragraph 223
sentence 223
stopwords 223
word 223
recreating an index 45
relation in a thesaurus 86
RESULT LIMIT keyword 140
S
sample functions
running 80
SCORE function
text example 75
syntax 150
search argument
attribute name 144
BOOLEAN operators 140
description 137
free-text search 79
fuzzy search 78, 142
search argument (continued)
numeric attribute search 79
search-primary operators 141
searching for parts of a term 76
searching for terms in a fixed sequence 77
searching for terms in any sequence 75
searching for terms in document sections 78
searching for terms in the same paragraph 77
searching for terms in the same sentence 77
searching with & and \l 76
searching with NOT 78
specifying 75
taxonomy 138
thesaurus search 78
using masking characters 76
using wildcard characters 76
search argument keywords
COUNT 144
EXPAND 144
FUZZY FORM OF 142
PRECISE FORM OF 142
RESULT LIMIT 140
SECTION 141
STEMMED FORM OF 142
STOP SEARCH AFTER number DOCUMENTS(S) 140
TERM OF 144
THESAURUS 143
search functions
CONTAINS 148
NUMBEROFMATCHES 149
SCORE 150
SQL table-valued 151
stored procedure 156
search term expansion
See thesaurus
search-primary operators 141
searching for text
getting the number of matches found 74
getting the score of a found document 75
making a query 74
overview 74
syntax 138
using a stored procedure 80
using a table-valued function 81
server
starting 96
stopping 97

Index 253
server (continued)
  tracing faults 233
space requirements for indexes 23
START command
  syntax 96
  using 27, 52
starting DB2 Net Search
  Extender 96
STEMMED FORM OF keyword 142
STOP command
  syntax 97
  using 27, 52
STOP SEARCH AFTER number
  DOCUMENTS(S) keyword 140
stopping DB2 Net Search
  Extender 97
stored procedure
  activating a text index 42
  deactivating a text index 42
  overview 38
  text indexes on view 42
  updating a text index 41
Stored Procedure function
  searching 80
  syntax 156
structured documents
  default document models 159
  enabling section support 159
  example 78
  overview 83
  search syntax 141
Summary panel 66
system requirements 13

T
Table-Valued function
  text indexes on view 42
Table-Valued Search Function
  searching 81
  syntax 151
tablespace 35, 56
Target panel 56
TERM OF keyword 144
text characteristics
  CCSID 23
  format 23
Text Properties panel 59
Text Search Engine
  languages supporting
    stopwords 224
    reason codes 225
    stopwords 223
    tokenization 223
  thesaurus
    compiling 88
  thesaurus (continued)
    concepts 85
    creating 88
    definition file 88
    messages 239
    structure 85
    supported CCSIDs 237
    thesaurus definition syntax 171
    thesaurus compile utility 130
    thesaurus search
      example 78
      syntax 143
    THESAURUS keyword 143
    tracing faults 233
    triggers
      creating 35
      description 5
      troubleshooting 233
U
UNIX installation 14
UNIX installation verification 16
Update Characteristics panel 61
update frequency 45
UPDATE INDEX command
  for a stored procedure 41
  RECREATE option 45
  syntax 132
  update frequency 45
  using 46
Update Index dialog 71
user roles
  database administrators 25
  DB2 instance owner 24
  text table owners 25
user scenarios
  SQL scalar search example 17
  SQL table-valued function
    example 20
  stored procedure search
    example 19
  using large amounts of
    memory 179
    for AIX 179
    for Sun Solaris 180
    for Windows 179
V
views
  created views 33
db2ext.dbdefaults 33, 181
db2ext.indexconfiguration 33, 185
db2ext.proxyinformation 33, 183
views (continued)
db2ext.textindexes 33, 48, 183
db2ext.textindexformats 33, 186
log view 187
overview 10
W
wildcard characters in a search term 76
Windows installation 15
Windows installation verification 15
X
XML documents
  default document model 159
  defining a document model 165
  document format 23
  document type definitions for
    document models 217
  limitations 220
  structured documents 159
  XPath expression semantics 218
XPath expression semantics 218

254  DB2 Net Search Extender Administration and User’s Guide
Readers’ Comments — We’d Like to Hear from You

IBM DB2 Universal Database
Net Search Extender
Administration and User’s Guide
Version 8.1

Publication No. SH12-6740-01

Overall, how satisfied are you with the information in this book?

<table>
<thead>
<tr>
<th>Very Satisfied</th>
<th>Satisfied</th>
<th>Neutral</th>
<th>Dissatisfied</th>
<th>Very Dissatisfied</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Overall satisfaction

How satisfied are you that the information in this book is:

<table>
<thead>
<tr>
<th>Very Satisfied</th>
<th>Satisfied</th>
<th>Neutral</th>
<th>Dissatisfied</th>
<th>Very Dissatisfied</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Accurate
Complete
Easy to find
Easy to understand
Well organized
Applicable to your tasks

Please tell us how we can improve this book:

Thank you for your responses. May we contact you?  Yes  No

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

Name

Address

Company or Organization

Phone No.