DB2 Universal Database

Net Search Extender Administration and Programming Guide

Version 7

SC27-0818-04

DB2 Universal Database

Net Search Extender Administration and Programming Guide

Version 7

SC27-0818-04

Note!
Before using this information and the product it supports, be sure to read the general information under "Notices" on page 91.

Fifth Edition, June 2001

This edition applies to DB2 Net Search Extender Version 7.2, program number 5648-D66. This edition also applies to all subsequent releases and modifications until otherwise indicated in new editions.

This edition replaces SC27-0818-03.

Contents

About this book

DB2 Net Search Extender contains a DB2® stored procedure that adds the power of fast full-text retrieval to Net.Data®, Java, or DB2 CLI applications. It offers application programmers a variety of search functions, such as fuzzy search, stemming, Boolean operators, and section search.

Searching using Net Search Extender can be particularly advantageous in the Internet when performance is an important factor.

Who should use this book

This book is intended for:

- Administrators who prepare databases for text search by creating Net Search Extender indexes, and who update and maintain these indexes.
- Database application programmers who implement text search for end users.

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.

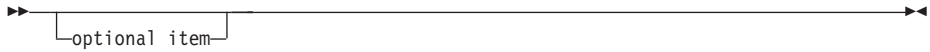
  The —► symbol indicates that the statement syntax is continued on the next line.

  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).

```
►►—required item————————————————————————————————————►◄
```

- 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.

- 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.

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 any of the following methods to provide comments:

- Send your comments from the Web. Visit the Web site at:

  http://www.ibm.com/software/data/db2/extenders

  The Web site has a feedback page that you can use to enter and send comments.

- Send your comments by e-mail to swsdid@de.ibm.com. Be sure to include the name of the book, the order 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).
- Fill out a Readers' Comments form at the back of this book and return it by mail, by fax, or by giving it to an IBM representative. The mailing address is on the back of the form. The fax number is +49-(0)7031-16-4892.

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.

Part 1. Guide

Chapter 1. Introduction

DB2 Net Search Extender adds the power of a fast full-text retrieval engine to Net.Data®, Java, or DB2 CLI applications integrated into DB2 Universal Database.

Optimized for performance, the benefits of using Net Search Extender are:

- Fast indexing of very large data volumes
- Searching at high speed with a large number of concurrent users
- Storing presorted table columns in main memory at indexing time to avoid expensive database access and paging at search time

Features

Net Search Extender provides complementary functionality to that provided by DB2 Text Extender and DB2 Text Information Extender, the other full-text retrieval search engines integrated into DB2. The key features of Net Search Extender are:

- Indexing
  - One very fast index type (Ngram)
  - Multiple indexes on the same text column are possible
  - Indexing proceeds without locking data
  - Dynamic updating of indexes to reflect changes in the database
- Search
  - Provides a stored procedure on the server instead of UDFs
  - Allows word, phrase, stemmed, or fuzzy search
  - Identifies and restricts searching to sections within documents that have been marked by special tags
  - Offers numeric search on a range of values
  - Supports Boolean and wildcard operations
- Search results
  - Lets you specify how the search results are sorted at indexing time, or uses relevance rank values for sorting
  - Lets you specify search result subsets when large data volumes are searched and large result lists are expected
  - Lets you set a limit on search terms with a high hit count
  - Allows positioning (cursor-setting) access on search results

These features enable Net Search Extender to provide solutions for the Internet, intranet, and other DB2–based applications.

However, if your solution requires any of the following features, you should consider using DB2 Text Extender or DB2 Text Information Extender, as they are not available in the Net Search Extender solution.

- Integrating SQL functionality into your text search
- Extensive linguistic search capabilities (tokenization, sentence recognition, term normalization, base form term reduction, stop-word filtering, and decomposition)
- Additional index types (linguistic and precise)
- Access of the optimizer to cost functions for the textSearch part of a query

Chapter 2. Installation

This chapter describes how to install DB2 Net Search Extender on UNIX and Windows NT.

System requirements

Net Search Extender runs on one of the following operating systems:

- AIX® Versions 4.2.1 or higher
- HP-UX V11
- Solaris V2.6 (SunOS V5.6) or higher
- Windows NT with Service Pack 5, or Windows 2000
- Linux (minimum kernel version 2.2.15, minimum glibc version 2.1.3 release 15). No distribution prerequisites necessary.

Net Search Extender requires that DB2 EE V7.1 is installed with at least Fixpack 2 applied.

If you want to use Net.Data to access the stored procedure of Net Search Extender, a minimum of Net.Data V6.1 needs to be installed.

If you want to use Java to access the stored procedure, a minimum of JDK 1.1.6 needs to be installed.

There is a minimum memory requirement of 256 MB RAM.

The overall memory requirement depends on the number of documents to be indexed, and on the performance optimization of your indexes. See point 3 on page 23, ″Calculate the amount of resources needed″ for further information.

Unix installation

Net Search Extender can be installed on the following platforms:

- AIX
- Solaris
- HP-UX
- Linux

To install Net Search Extender and create Net Search Extender instances on Unix systems, you must be logged in as a user with 'root' authority.

Depending on the platform, the files are installed in the following directory:

- For AIX: `/usr/lpp/db2nx_07_01`
- For Solaris and HP-UX: `/opt/IBMdb2nx/V7.1`
- For Linux: `/usr/IBMdb2nx/V7.1`

Net Search Extender files for the AIX platform are listed below. The listing is similar for HP-UX, Linux and Solaris platforms.

./bin/db2nx	Administration command file
./bin/nxservice	Diagnostic tool for collecting service information
./bin/nxupd72	Net Search Extender migration script
./bnd/desfpssp.bnd	Bind file for the stored procedure
./bnd/desfpiiu.bnd	Bind file for db2nx
./lib/desfpssp	Search library for the stored procedure
./lib/libdesfpg4.a	Search library
./lib/libicui18n.a	Codepage conversion library
./lib/libicuuc.a	Codepage conversion library
./data/char.brk	Codepage conversion data files
./data/icudt17b.dat	Codepage conversion data files
./data/line.brk	Codepage conversion data files
./data/line_th.brk	Codepage conversion data files
./data/sent.brk	Codepage conversion data files
./data/thaidict.brk	Codepage conversion data files
./data/word.brk	Codepage conversion data files
./data/word_th.brk	Codepage conversion data files
./instance/nxicrt	Instance creation and update program
./instance/nxidrop	Instance deletion program
./samples/nxsample	Sample database setup program
./samples/NXSample.java	Java sample program
./samples/NXSampleRank.java	Java sample with 'rank' support
./samples/nxsample.d2w	net.data sample program
./samples/nxsample.imp	Data for sample database
./ddl/nxcproc.ddl	Data definition file for migration

./license/LA_de, LE_de	The license files for different languages (these files are for German)
./lic/db2nse.lvl	Net Search Extender version and level information
./lic/db2nse.lic	License key (absent in the Try & Buy version)
./nls/De_DE/desfpapi.dat	Message files for different languages (this one is for German)
./nls/De_DE/desfpins.cat	Installation message files for different languages (this one is for German)
release.txt	Latest changes
readme.txt	Contents list of the Net Search Extender CD
copyright	Copyright information
nsdoc.pdf	This document in PDF format

Begin by installing the product components on the target machine and then create a DB2 Net Search Extender instance.

Install the product components

To install the product components:

1. Log on to the target machine as the root user.
2. Install the product components.
   - For AIX:

     Use SMIT (System Management Interface Tool) for a user-prompted interface or enter the `installp` command.

     To run SMIT:

     a. Enter `smit install_latest` to open **The Software Installation and Maintenance Tool** menu.

     b. Enter the directory of the install file in the **Input device / directory for software** field. This is /<cdrom-dir>/aix where <cdrom-dir> is your cdrom drive path.

     c. Click the DO button or press ENTER the installation directory.

     d. Select the component you want to install in the **Software to install** field.

     e. Click the DO button or press ENTER.

     f. Press ENTER to confirm the installation parameters.

     g. If the registration screen does not appear during install, set the DISPLAY variable and call `/usr/lpp/db2nx_07_01/register`.
   - For Solaris:

     Use the `pkgadd` command as follows:

pkgadd -d /<cdrom-dir>/sun/ibmnets.pkg where <cdrom-dir> is your cdrom drive path. Select the IBMNETS package from the list of available packages displayed.

- For HP-UX:

  Use the swinstall command as follows:

  ```
  /usr/bin/swinstall -s <yourhostname>:<full pathname of ibmnets.pkg> DB2NSE
  ```

  for example,

  ```
  /usr/sbin/swinstall -s myhost.mydomain.com:/<cdrom-dir>/hp/ibmnets.pkg DB2NSE
  ```

- For Linux:

  Use the rpm command as follows:

  ```
  rpm -i db2nse-7.2.i386.rpm
  ```

3. Logout.

Create a Net Search Extender instance

After installation, a DB2 Net Search Extender instance must be created for each DB2 user using Net Search Extender. Note that this instance is different from the DB2 instance. To create an instance:

1. Ensure you are active as the root user.
2. Move to the installation directory.
3. Run ./instance/nxicrt *instance-name* where *instance-name* is an existing DB2 7.1 instance.

   Running nxicrt creates a directory db2nx under the DB2 instance home directory. Do not add files or directories here, as these are lost when the instance is deleted.

   db2iupdt is called during nxicrt to ensure that the license file is tranferred.
4. Logout.
5. To drop a Net Search Extender instance, run

   ./instance/nxidrop *instance-name* from the installation directory.

Windows NT installation

For Windows NT, Net Search Extender files are installed in the directory that you specified during the installation setup.

Net Search Extender files are:

.\BIN\db2nx.exe	Administration command file
.\BIN\desfpdem.exe	Net Search Extender background daemon
.\BIN\desfpssp.dll	Search library for the stored procedure

.\BIN\desfpgtr.dll	Search library
.\BIN\desfpc.dll	Runtime library
.\BIN\nxstart.bat	Net Search Extender start program
.\BIN\nxstop.bat	Program to stop Net Search Extender
.\BIN\nxupd720.exe	NetSearch migration script
.\BIN\desfp60.dll	Net Search Extender runtime library
.\BIN\desfpsvc.exe	Net Search Extender Windows NT service program
.\BIN\icudt17l.dll	Codepage conversion data files
.\BIN\icuin17.dll	Codepage conversion data files
.\BIN\icuio17.dll	Codepage conversion data files
.\BIN\icuuc17.dll	Codepage conversion data files
.\BIN\nxclean.exe	Net Search Extender service tool
.\BIN\nxservice.bat	Diagnostic tool for collecting service information
.\SAMPLES\nxsample.bat	Sample database setup program
.\SAMPLES\NXSample.java	Java sample program
.\SAMPLES\NXSampleRank.java	Java sample with 'rank' support
.\SAMPLES\nxsample.d2w	net.data sample program
.\SAMPLES\nxsample.ddl	Data definition file for table creation
.\SAMPLES\nxsample.imp	Data for sample database
.\bnd\desfpssp.bnd	Bind file for the stored procedure
.\bnd\desfpiiu.bnd	Bind file for the db2nx.exe
.\ddl\nxcproc.ddl	Data definition file for migration
.\msg\desfpapi.dat	Message catalog file
.\cfg\db2nse.lvl	Net Search Extender version and level information
.\DATA\char.brk	Net Search Extender data files
.\DATA\line.brk	Net Search Extender data files
.\DATA\line_th.brk	Net Search Extender data files
.\DATA\sent.brk	Net Search Extender data files
.\DATA\thaidict.brk	Net Search Extender data files

.\DATA\word.brk	Net Search Extender data files
.\DATA\word_th.brk	Net Search Extender data files
.\license\LA_it	The license files for different languages (this one is for Italian)
.\prt\pribmita.dll	The registration files for different languages (this one is for Italian)
.\release.txt	Latest changes
.\readme.txt	Contents list of the Net Search Extender CD
.\nsdoc.pdf	This document in PDF format Adobe Acrobat Reader is needed to display the PDF documentation.

Run `setup.exe` to transfer the files from the installation package to the target machine. Restart the Windows NT machine to activate the system settings.

If the daemon process is not available, no queries can be performed. The background daemon is now implemented as a Windows service. By default, this service is activated when Net Search Extender is installed and automatically starts when the system restarts. If you prefer to have control over this service, use the Windows settings to change the default behavior.

To manually start or stop the service, use:

```
NET START "DB2 Net Search Extender"
```

or

```
NET STOP "DB2 Net Search Extender"
```

If the daemon has been stopped and restarted (manually or automatically), all indexes are deactivated. You will have to reactivate the indexes that you want to use for searching.

Migration from version 7.1

If DB2 Net Search Extender V7.1 is already installed, you can migrate your indexes to work with 7.2, the latest version.

Before installing DB2 Net Search Extender 7.2, deactivate all the indexes that you want to integrate and delete all the indexes that you do not want to use further, then:

1. Call `db2stop` to stop the database.
2. (Windows only) Call `nxstop` to stop the daemon.

3. Install DB2 Net Search Extender V7.2 (Windows NT only: this automatically starts the background daemon)
4. Run `db2start` to start the DB2 instance.
5. Call DB2 Net Search Extender V7.2 migration tool `nxupd72` to migrate a specified database. `nxupd72.exe` (`nxupd72` on non Windows platforms) checks the existing text column indexes and updates their Superblock- and Inode-shared memory if they are using Fast Recovery. If no Fast Recovery is used, nothing has to be done, because the information in shared memory is built from scratch when reactivating the text column index.

After migration, no system restart is necessary if the path settings are the same as in the previous installation.

The text column indexes can then be activated and used again.

Note

If you are migrating from DB2 Net Search Extender 7.1.2 or earlier, you can use the following migration information:

- If your indexes were created with an ORDER BY statement, migration is simply done by using the DB2NX ACTIVATE INDEX command.
- If your indexes did not use a pre-sorted order of the in-memory table, it is necessary that you run the DB2NX UPDATE INDEX command for each index that you want to keep using. This is necessary, since DB2 Net Search Extender now uses an implicit ORDER BY <key column> if no ORDER BY is used during ENABLE TEXT COLUMN. Otherwise, the correctness of search results involving numeric indexes cannot be guaranteed for large volumes of data, as the text index and item index may not be in sync.

Before executing queries, you need to run the DB2 bind command `db2 bind desfpssp.bnd`. This bind file is located in the directory `$DB2NX_INSTOWNERHOMEDIR/db2nx/bnd` for AIX, HP-UX, and Solaris. For Windows NT, the directory is `<inst.dir>\bnd`

Migrating to a new version of DB2

When installing a new version of DB2 you have to rebind the bind file against your database using the following commands:

1. `db2 connect to <your database>`
2. `db2 bind desfpssp.bnd`
3. `db2 bind desfpiiu.bnd`

4. Both .bnd files are located in the <instancehome>/db2nx/bnd (UNIX), or in the <DB2NXPATH>/bnd (Windows NT) directory.

Chapter 3. Running the sample programs

A set of sample programs is provided with Net Search Extender. These programs are located in the `db2nx/samples` directory. The sample programs are a means by which you can verify your installation.

The sample programs can only be run after Net Search Extender has been installed and the relevant DB2 and Net Search Extender instances have been created (UNIX only).

The sample provides a database setup program and three search examples, two that run in Java and one that runs in Net.Data.

The sample program `nxsample` creates a table in a sample database, fills this table with sample data, and indexes some of the data in the table.

You can then use the Net.Data and Java samples to run searches against that data.

These are the sample programs:

`nxsample`
: A database setup program for UNIX and Windows NT that creates a sample table containing sample data and three sample indexes.

  You must run this program before running the Net.Data sample `nxsample.d2w` and the two Java samples `NXSample.java` and `NXSampleRank.java`.

`nxsample.d2w`
: A Net.Data macro that displays a Web page that allows searching on the indexes created by the sample database setup program `nxsample`.

`NXSample.java`
: A Java program that allows searching without ranking on the search results from indexes created by the sample database setup program `nxsample`.

`NXSampleRank.java`
: A Java program that allows searching with ranking on the search results from indexes created by the sample database setup program `nxsample`.

To prepare your own database and run searches against your own data, see "Chapter 4. Using Net Search Extender" on page 21.

Running the sample database setup program

To run the sample database setup program:

On UNIX, from the $DB2NX_INSTOWNERHOMEDIR directory, change the directory to ./db2nx/samples/and then run ./nxsample [*database-name*]

On Windows NT, from the db2cmd command prompt, change the directory to <installdir>/samples/ where <installdir> is the installation directory of Net Search Extender, and run nxsample database-name

If you don't want to use an existing database, create a new one.

On Unix systems, if you don't provide a database name, the default database specified in the environment variable DB2DBDFT is used. It is not possible to create a database by just specifying the name; it must already exist.

On Windows NT, you must specify a database name; the database is automatically created if it does not already exist.

The nxsample program performs the following tasks:

- Creates a table db2nx.sample in the specified database
- Puts sample data into that table
- On Windows NT, starts the background daemon
- Runs db2nx enable database to enable the database for use by Net Search Extender
- Creates three sample indexes to be searched by the Net.Data sample macro nxsample.d2w or the Java sample programs NXSample.java and NXSampleRank.java

Monitoring the Net Search Extender environment

After running nxsample, two commands can be used to monitor the Net Search Extender environment:

get status

Issuing the db2nx get status command provides the following information about the status of the database:

- Whether it is enabled for Net Search Extender
- The tables that have text columns enabled, and the text column and index names

Example command:

```
db2nx get status database nxsample
```

Example command output:

```
Database is enabled for 'DB2 Net Search Extender'.

  Table DB2NX.SAMPLE is enabled
    IndexName   ColumnName
    ---------   ----------
    COMMENT     COMMENT
    TITLE       TITLE
    FPD         FORMAT_PUB_DATE

DES7800I  The command completed successfully.
```

get index status

Issuing the `db2nx get index status` command provides all the status information relating to the index/setup parameters of one of the indexes that has been created.

Example command:

```
db2nx get index status comment database nxsample
```

Example command output:

```
Settings of index 'COMMENT'

  Table                = 'DB2NX.SAMPLE'
  Column               = 'COMMENT'
  Index directory      = '/home/user1/db2nx/indices/'
  Temporary directory  = '/home/user1/db2nx/indices/'
  Key-column           = 'DOCID'
  Order by             = 'DOCID ASC '
  Optimize on:
    1.  = TITLE
    2.  = AUTHOR
    3.  = COMMENT
    4.  = FORMAT_PUB_DATE
    5.  = PRICE
    6.  = year ( last_updated )  as year_last_updated
  Numeric:
    1.  = PRICE
  Tags                 = no tags defined

  The Index is NOT enabled for Incremental Update

  Shared Memory Information :
    Internal Revision Number         : 72001
    (Re-)Creation Time               : Thu Mar 22 19:22:38 2001
    Existing Shared Memory Segments  : 3
    Size of SUPERBLOCK-Segment       : 8364 bytes
    Size of INODE-Segment            :  512 bytes
    Number of documents in SHM       : 8
    DATA-Segment 1 : allocated size  : 1290 bytes
```

Running the sample programs

```
In-Memory Table '/home/user1/db2nx/indices/COMMENT.SHM'
   ID    Address    Permissions    Creator  Group
m    393248 0x530dd80c --rw-r--r--    user1    staff      8364
m   1179689 0x490dd80c --rw-r--r--    user1    staff       512
m    262186 0x610dd80c --rw-r--r--    user1    staff      1290

DES7220I  Index is active.

DES7800I  The command completed successfully.
```

Running the Net.Data sample macro

Pre-requisites

The Net.Data sample macro can only be run if you have Net.Data installed to run with a Web server on your machine.

To run the Net.Data sample macro `nxsample.d2w`:

1. Copy `db2nx/samples/nxsample.d2w` to your Net.Data macro directory.
2. Open the macro for editing.
3. Update the values of the following variables:

```
DATABASE =  "your_database"
LOGIN    =  "your_user_ID"    Note: Remove this line, if already
                                    configured in Live connection
PASSWORD =  "your_password"   Note: Remove this line, if already
                                    configured in Live connection
```

4. Run the macro `http://your_hostname/cgi-bin/db2www/nxsample.d2w/main` in your Web browser.

After each Net.Data search, the connection from the DB2 client to the server is closed; DB2 reopens the connection for the next search. To enhance performance, you can use Net.Data's Live Connection to keep the connection open after each search.

See "Using your Net.Data search function" on page 28 for further information.

Running the Java sample programs

In the first example, the Java sample program `NXSample.java` is used. The second example then shows you how to run the Java sample program with ranking `NXSampleRank.java` on a client machine.

To run the Java sample program `NXSample.java` on the database server

1. Set up the environment to run DB2 Java programs according to the Building Java Applications and Applets section of the *DB2 Applications Development Guide.*
2. Copy `db2nx/samples/NXSample.java` to a work directory and ensure that the `CLASSPATH` environment variable points to that directory.
3. To compile the sample program, run:

   ```
   javac NXSample.java
   ```

4. To start the sample program, run:

   ```
   java NXSample yourDatabase "yourSearchTerm"
   ```

   Sample search calls

   ```
   java NXSample yourDatabase "\"kid\""
   java NXSample yourDatabase "(\"bestseller\" | \"pulitzer\") & \"book\""
   java NXSample yourDatabase "\"kid\" & not \"duck\""
   ```

   The index name `comment` is used by default. You can change the index name and the other search parameters in the variable-declaration section of the source file.

To run the Java sample program `NXSampleRank.java` on a DB2 client

1. Set up the environment to run DB2 Java programs according to the Building Java Applications and Applets section of the *DB2 Applications Development Guide* for the appropriate platform.
2. Copy the sample program to a working directory of the client machine to which the DB2 instance owner has write access. Ensure that the `CLASSPATH` environment variable points to that directory.
3. To compile the sample program, run:

   ```
   javac NXSampleRank.java
   ```

4. To start the sample program, run:

   ```
   java NXSampleRank yourDatabase yourSearchTerm
   ```

Note

Search terms need to be included in a single set of quotation marks (″ ″). See the previous Java sample `NXSample.java` for details. For the syntax of search terms, see “Chapter 6. Search syntax for the searchTerm parameter” on page 47.

Stored procedures for searching

Underlying the provided samples are calls to a DB2 stored procedure. The calls provide DB2 Net Search Extender with four different types of search functionality:

- A standard search (no ranking)
- A standard search with tracing (no ranking)
- A standard search with ranking
- A standard search with ranking and tracing

These stored procedures can then be used within Net.Data®, Java, or DB2 CLI applications integrated into DB2.

> **Note**
> The stored procedure is registered with the instance owner schema. When searching with a different user ID, it is necessary to qualify your stored procedure with the schema of the instance owner.

The signatures for the stored procedures are shown below.

Standard search without ranking

The following stored procedures are used:

- `db2nx.textsearch` Standard search
- `db2nx.textsearch_t` Standard search with enhanced tracing

On UNIX and Windows NT

```
textSearch (
   OUT totalDocs               INTEGER      ,
   IN  searchTerm              VARCHAR(500) ,
   IN  maxHitCount             INTEGER      ,
   IN  maxIntermediateHitCount INTEGER      ,
   IN  startRow                INTEGER      ,
   IN  maxRowsToReturn         INTEGER      ,
   IN  indexname               CHAR(50)     ,
   IN  indexDirectory          CHAR(100)    ,
   IN  tmpDirectory            CHAR(100)    ,
   IN  sqlStatement            VARCHAR(500) ,
   IN  dataSource              INTEGER      ,
   OUT wordCounts              CHAR(250)    ,
   )
```

Note that the signature for `textSearch_t` is the same.

The parameters for the procedure are explained in conjunction with using the Net.Data sample macro and the sample Java program. See "Using your Net.Data search function" on page 28 for further information.

Standard search with ranking

The following stored procedures are used:

- `db2nx.textsearch_r` Standard search with ranking
- `db2nx.textsearch_rt` Standard search with ranking and enhanced tracing

On UNIX and Windows NT

```
textSearch_r (
   OUT totalDocs               INTEGER      ,
   IN  searchTerm              VARCHAR(500) ,
   IN  maxHitCount             INTEGER      ,
   IN  maxIntermediateHitCount INTEGER      ,
   IN  startRow                INTEGER      ,
   IN  maxRowsToReturn         INTEGER      ,
   IN  indexname               CHAR(50)     ,
   IN  indexDirectory          CHAR(100)    ,
   IN  tmpDirectory            CHAR(100)    ,
   IN  sqlStatement            VARCHAR(500) ,
   IN  rankOper                INTEGER      ,
   IN  dataSource              INTEGER      ,
   OUT wordCounts              CHAR(250)    ,
   )
```

In this stored procedure, there is an additional parameter for ranking, `rankOper`. Note that the signature for `textSearch_rt` is identical.

The parameters for the procedure are explained in conjunction with using the Net.Data sample macro and the sample Java program. See "Using your Net.Data search function" on page 28 for further information.

Chapter 4. Using Net Search Extender

This chapter provides an overview of the tasks involved in using Net Search Extender on databases that you have created, or are about to create. This includes the following:

- Preparing a database for Net Search Extender search
- Enabling the database for Net Search Extender search (creating an index)
- Creating the Search function
- Running the Search function

All these tasks can only be carried out on a DB2 server. It is assumed that the administrator, logged in as a DB2 instance owner, has the following environment variables set:

```
DB2DBDFT
DB2NX_INSTOWNERHOMEDIR
```

DB2 requires you to specify a codepage for the database. Net Search Extender supports all codepages that DB2 supports on the respective platforms.

When issuing a `db2nx` command and specifying the optional database parameter, a database connection is made to that database. If the optional database parameter is not specified, a connection is made to $DB2DBDFT.

For information on monitoring and maintaining indexes you have created, and on administration-related tasks and database backup and retrieval, see "Chapter 5. Administration overview" on page 37.

Preparing a database for Net Search Extender search

Before preparing a database, check that the following steps have been completed:

- A Net Search Extender instance has been created.
- Net Search Extender and DB2 servers have been restarted with the new environment.

The sample program `nxsample` from "Chapter 3. Running the sample programs" on page 13 is used as an example in this section.

1. **Design, create (tablespace and tables) and load a database with text data.**

   The following table is created by `nxsample`:

```
db2 "create table $TABLE(\
         docid            INTEGER NOT NULL,\
         author           VARCHAR(50),\
         title            VARCHAR(50),\
         subject          VARCHAR(100),\
         comment          LONG VARCHAR,\
         format_pub_date  LONG VARCHAR,\
         price            DECIMAL(8,2),\
         last_updated     timestamp,\
         PRIMARY KEY (docid) )"
```

After creating the table, you can check the result with:

```
db2 describe table  db2nx.sample
```

This command returns:

```
Column                    Type      Type
name                      schema    name              Length   Scale Nulls
------------------------- --------- ---------------- -------- ----- -----
DOCID                     SYSIBM    INTEGER                 4     0 No
AUTHOR                    SYSIBM    VARCHAR                50     0 Yes
TITLE                     SYSIBM    VARCHAR                50     0 Yes
COMMENT                   SYSIBM    LONG VARCHAR        32700     0 Yes
FORMAT_PUB_DATE           SYSIBM    LONG VARCHAR        32700     0 Yes
PRICE                     SYSIBM    DECIMAL                 8     2 Yes
LAST_UPDATED              SYSIBM    TIMESTAMP              10     0 Yes

  7 record(s) selected.
```

2. **Identify the text data and related columns.**

   Identify the following information:

   - The table name and text column name where specific text data resides. The column type must be CHAR, VARCHAR, LONG VARCHAR, or CLOB.
   - The primary key of the table, and an index name of your choice.

   Optional information includes:

   - Tagging. Tags are used to identify sections of your document text that you want to limit searches to. Refer to "ENABLE TEXT COLUMN" on page 60 for details. A maximum of five tags can be specified. In `nxsample`, three tags are defined, `zzformat, zzpub, and zzdate`.
   - Optimized columns whose data is to be loaded into memory.
   - If you plan to index numeric data from your table, the data types must be INTEGER, DOUBLE, or DECIMAL. See "ENABLE TEXT COLUMN" on page 60 for further information.
   - Order-by columns, if the search results are always to be sorted in a predefined order.

- A location for the created index, if you do not want to store the index in the default directory. For different databases, use a different directory. This avoids a potential naming problem if you want to use the same index name for indexes of different databases.

3. **Calculate the amount of resources needed.**

   When an index is created, considerable amounts of data may be loaded into memory (actually, shared memory) of the machine on which the database server is running.

   The amount of shared memory required by an index depends on the following parameters:

   - The number of rows in the database table in which the indexed documents are stored
   - The width of the key column specified by the USING parameter of the ENABLE TEXT COLUMN command when the index was created
   - The width of the columns specified by the OPTIMIZE ON parameter of the ENABLE COLUMN command

   In addition to the amount of data put into shared memory, DB2 Net Search Extender allocates memory during the indexing process when retrieving data from DB2.

   To achieve higher performance, data is retrieved from DB2 in blocks of 100 rows at a time. Note that if these rows contain large CLOBs, the memory required is 100 times the size of each row. For example, rows containing CLOBs of 4 MB, will have a memory requirement — just for the retrieval — of at least 400 MB.

   To calculate the maximum amount of shared memory required by an index, use the following approximation formula:

```
max. memory    = total amount of optimized columns (including key column!)
                 + (100 * width of all indexed columns (text & numeric) )
```

   Because Net Search Extender optimizes the memory requirements by not saving trailing spaces, the effective amount of required memory can be much less than the amount calculated by the formula above.

   It is important to retain the shared memory of only those indexes that are currently needed for search. You can free the shared memory of an index that is currently not needed by issuing the DEACTIVATE INDEX command. Note that no search is possible on an index while it is deactivated. To get the index data back into shared memory and make it searchable again, run the ACTIVATE INDEX command.

   If your operating system has been restarted, all indexes are deactivated.

On Windows NT, if `nxstop` has been called, all indexes are deactivated.

On AIX, the amount of shared memory available on your database server depends on the version of AIX. Versions 4.2.1 to 4.3.1 allow 256 MB of shared memory to be allocated in one shared memory block. Version 4.3.2 and above allow up to 2 GB of shared memory per allocation.

Note

On Solaris and HP-UX, the maximum number of files that can be open is set at 64 by a system parameter default file. If you receive an open error message (rc 223) when indexing large amounts of data, increase the limit using the `ulimit -n` command.

Also note that other applications may be using system memory.

Tip

The AIX command `vmstat` reports virtual memory statistics. The Net Search Extender administrator may want to collect reports generated by the `vmstat` command to determine the system's paging activity.

Excessive paging can affect the Net Search Extender shared memory table performance. To reduce paging activities, add real memory to the system. The Net Search Extender administrator may also want to monitor the memory usage of various applications, and give advice on possible work changes to improve the paging situation.

In Windows NT, the amount of available shared memory depends on the size of the virtual memory of your system.

4. **Verify and set the environment variables.**

   Set the DB2DBDFT environment variable to the name of the database to be prepared for Net Search Extender search. See "Environment variables" on page 54 for other environment variables.

5. **(Windows NT only) Start the background daemon.**

   At the command prompt, enter: `nxstart`

6. **Enable the database.**

   At the command prompt, enter: `db2nx enable database` to prepare the database, which makes the stored procedure known to the database.

7. **Enable the text column - create a Net Search Extender index.**

   Create an index by entering the command:

   `db2nx enable text column` *`table text-column`* `index` *`index-name`* `using` *`key-column`*

Tip

The full list of options for the db2nx command is displayed when you enter the command db2nx help.

The command returns:

```
db2nx db2nx-command

List of db2nx commands
HELP
ENABLE DATABASE  [database [USER1 user1 USING password]]
                 [TABLESPACE tablespace]
DISABLE DATABASE [database [USER1 user1 USING password]]
GET STATUS       [DATABASE database [USER1 user1 USING password]]
ENABLE TEXT
COLUMN           table text-column INDEX index USING key-column
                 [TAGS ( tag [{,tag} ...] ) ]
                 [OPTIMIZE ON ( opt-column [ {,opt-column} ...] ) ]
                 [NUMERIC ( num-column [ {,num-column} ...] ) ]
                 [ORDER BY column {ASC|DESC} [{,column {ASC|DESC}} ... ]]
                 [DIRECTORY directory [TEMP DIRECTORY temp-directory ]]
                 [FASTRECOVERY]
                 [DATABASE database [USER1 user1 USING password]]
                 [INCREMENTAL [TABLESPACE tablespace]]
DISABLE TEXT
COLUMN           INDEX index [DATABASE database [USER user USING
                 password]]
UPDATE INDEX     index [DATABASE database [USER1 user1 USING password]]
                 [INCREMENTAL [COMMITCOUNT commitcount]]|[REORG]
GET INDEX STATUS index [DATABASE database [USER1 user1 USING password]]
ACTIVATE INDEX   index [DATABASE database [USER1 user1 USING password]]
DEACTIVATE INDEX index [DATABASE database [USER1 user1 USING password]]
```

All the options are explained in "ENABLE TEXT COLUMN" on page 60.

Examples:

```
db2nx enable text column DB2NX.SAMPLE comment index comment using DOCID \
  optimize on \(title,author,comment,format_pub_date,price, \
  {year\(last_updated\) \as year_last_updated}\) \ numeric \(price\) \
  order by DOCID ASC DATABASE NXSAMPLE

db2nx enable text column DB2NX.SAMPLE title index title using DOCID
  optimize on \(title,author,comment,format_pub_date,price, \
 {year\(last_updated\) \as year_last_updated}\) numeric \(price\) \
 order by DOCID ASC \DATABASE NXSAMPLE

db2nx enable text column DB2NX.SAMPLE format_pub_date index fpd using DOCID \
```

```
TAGS \ (zzformat,zzpub,zzdate\) \
optimize on (title,author,comment,format_pub_date,price, \
{year\(last_updated\) \as year_last_updated}\) \ numeric \(price\)
order by DATABASE ASC DATABASE NXSAMPLE
```

where `comment, title and format_pub_date` are the columns containing the text data to be indexed for a later Net Search Extender search, and `docid` is the primary key of table `db2nx.sample`

If data is expected to change frequently, you may need to use the `INCREMENTAL` keyword; see "Incremental Index Update" on page 40 for further information.

For more details about the parameters, see the description of the "ENABLE TEXT COLUMN" on page 60.

Note

All keywords used on the command line, such as database name, table name, column name, index name, tag name, directory, and so on, must be specified using SBCS (Single Byte Character Set) characters.

The indexes can now be searched.

Searching an index

You search a Net Search Extender index by calling one of the stored procedures `textSearch, textsearch_t, textsearch_r`, or `textsearch_rt` in one of the following ways:

- The most basic way of searching is using the DB2 CALL command to directly call the stored procedure. To call the stored procedure, specify the parameters, see "Stored procedures for searching" on page 18 for information; quote input parameters with CHAR values and, use question marks ('?') in all output parameter positions. In the following example, the first and last parameters, totaldocs and wordcounts, are output parameters.

  ```
  db2 "call textSearch(?,'\"paperback\"',100,100,0,100,'comment',\
                       '/home/user1/db2nx/indices',/tmp',' ',0,?)"
  ```

- Alternatively, use the CALL statement as a static SQL statement, or run it from a CLI program using functions like SQLPrepare() or SQLExecute().

However, more comfortable ways of interfacing to the stored procedure are provided by the following methods:

- Using a Net.Data macro
- Using a Java program

This section describes how to make a search using the last two methods to call a stored procedure. For each method, two examples are used, one with ranking and one without ranking.

Creating a Net.Data search function with ranking to search a database

> **Note**
> Before using this section, you should be familiar with Net.Data.

To make use of the stored procedure within a Net.Data macro, define a Net.Data function called `search` with the following parameter layout:

```
%FUNCTION(DTW_SQL) search (OUT INTEGER totalDocs,
                           IN VARCHAR(500) searchTerm,
                           IN INTEGER maxHitCount,
                           IN INTEGER maxIntermediateHitCount,
                           IN INTEGER startRow,
                           IN INTEGER maxRowsToReturn,
                           IN CHAR(50) indexname,
                           IN CHAR(100) indexDirectory,
                           IN CHAR(100) tmpDirectory,
                           IN VARCHAR(500) sqlStatement,
                           IN INTEGER rankOper
                           IN INTEGER dataSource,
                          OUT CHAR(250) wordCounts,
                          OUT outTable)

                    returns (RESULT) {

  Call desfpssp!textSearch_r(outTable)

  %REPORT(outTable){
  <center> The first column is $(V1). The second column is $(V2). </center>
  %}
%}
```

On UNIX and Windows NT, the line `Call desfpssp!textSearch_r()` runs the stored procedure with ranking. To run the stored procedure without ranking, use `Call desfpssp!textSearch()`.

Note

In the parameter layout displayed above, an additional line has been added for ranking:

On UNIX and Windows NT: `IN INTEGER rankOper`

If ranking is not required, this line **must not** be present, as the sequence and number of parameters is important for DB2 to be able to correctly handle the stored procedure.

The results are returned to the `%REPORT` section where you can format the display of the output or use Net.Data's default report feature.

Using your Net.Data search function

To use the search function that you have just created, place the Net.Data function call where you would like to issue the search in your Net.Data macro.

```
@search(totalDocs, searchTerm, maxHitCount, maxIntermediateHitCount,
        startRow, maxRowsToReturn, indexname, indexDirectory,
        tmpDirectory, sqlStatement, rankOper, dataSource, wordCounts, outTable)
```

Note

The length of the search argument is limited to 1000 bytes.

Below is a functional description of the parameters used in more detail.

Both types of search, with ranking and without, are covered in this section. However, the only parameters that are different for each type of search are:

```
sqlStatement
RankOper
```

totalDocs The total number of *hits* returned. *Hits* is the number of matches in the documents. You can have multiple matches per document or row.

The totalDocs variable does not always reflect the total number of documents returned by the query. If a query combines a textual search with an SQL statement, the SQL statement can increase or decrease the result set without reflecting this change in the totalDocs value.

searchTerm The word or words to be searched for. See "Chapter 6. Search syntax for the searchTerm parameter" on page 47.

maxHitCount The maximum number of hits to process. This is useful when one of the search terms has a high hit count and you want to improve performance by limiting the search result. Set to "0" or "" if you do not want the number of hits to be limited. If maxHitCount is less than the number of hits found, some of the search results will not be shown in your report. Note that the number of rows to be returned is set by maxRowsToReturn.

Using large datasets and high maxHitCount values can lead to an out-of-memory error during searches. To avoid this problem, reduce the maxHitCount value.

maxIntermediateHitCount
For searches with multiple search terms, the maximum hit count to be reached before retrieving hits for the next search term. This is useful when one of the search terms has a high hit count and you want to improve performance by limiting the result before it is joined with the result of the next search term. Set to "0" or "" if you do not want the number of intermediate hits to be limited. Use this parameter with caution: an intermediate result limited in this way can change the final search result report because the full intermediate queries may not be completed. This parameter is provided to help you improve performance, especially if the data is not well distributed, and if your system has limited memory resources for use during searching.

startRow The number of the row at which the result buffer is to start storing results. For example, if 1000 rows contain hits for a given search and you want results beginning at row 50, set startRow to 50.

maxRowsToReturn
The maximum number of rows to return beginning at startRow.

indexname The index to be used. This is the name of the index you specified in the `enable text column` command.

indexDirectory
The directory where the indexes are stored. If you do not specify this parameter, the default directory is used: `$DB2NX_INSTOWNERHOMEDIR/db2nx/indices`.

tmpDirectory The temporary directory to be used by the stored procedure. The database instance must have write access to this directory.

On Unix systems, the default value is /tmp. On Windows NT, the default value is the directory specified in the TEMP environment variable.

sqlStatement An SQL statement that specifies the columns and rows of the result table returned by the stored procedure. The statement for a search **without ranking** has the structure:

```
select ... from ... where ... <key-column> in (%s) ...
```

For example:

```
select DOCID,TITLE,AUTHOR from DB2NX.SAMPLE where DOCID in (%s)
```

A statement for a search **with ranking** has the structure:

```
select RSCORE, ... from ... %s <key-column> \
       where <key-column> in (%s) [order by RSCORE desc]
```

For example:

```
select RSCORE,DOCID,TITLE from DB2NX.SAMPLE %s DOCID \
       where DOCID (%s) ORDER BY RSCORE DESC
```

RSCORE is the calculated rank score of the documents and key-column is the unique key column specified in the USING clause of the ENABLE TEXT COLUMN command. The %s variable is expanded to a list of all key-column values relating to the documents found by the search.

Columns in the select statement are returned in the %RESULTS section of the Net.Data macro and accessed by referencing $(Vn), where n is the column number.

If dataSource is 0, this parameter is ignored.

rankOper A flag that specifies how to get the result set of documents scored. This can be seen in an example, based on the search term "Albert" & "Bernard" and the following document table:

Document Number	Document Content
1	Albert
2	Bernard
3	Albert and Bernard

0 no_ranking:

desfpssp!textSearch_r works in the same way as desfpssp!textSearch and only document 3, Albert and Bernard is returned.

1 fuzzy_ranking:

All the documents in the set that contain one or more of the search items are scored using the fuzzy operation. In this case, the logical operators are interpreted as fuzzy operators. This search returns all three documents, Albert, Bernard, and, Albert and Bernard, with the document 3 ranked highest.

Use this parameter value with care, especially when searching terms contain NOT clauses. The document containing the excluded term will still be in the result set, but with a low rank value.

2 strict_ranking:

The documents in the set obtained by the traditional strict logical operation are scored. This search returns only document 3, Albert and Bernard.

dataSource A flag that determines if row values come from DB2 or from memory.

0 Results come from the in-memory table only.

All columns specified in the OPTIMIZED ON section are returned together with the unique key.

1 Results are retrieved from DB2 according to the parameter `sqlStatement`.

wordCounts An output variable that returns the number of hits for each search term. These are listed in the order in which the search terms were entered. For example, if ″history Japanese″ was the search term, and wordCounts contains ″1000 210″ after the search, this means ″history″ was found 1000 times, and ″Japanese″ was found 210 times.

outTable An output variable that returns the results table.

This parameter is specific to Net.Data. If you call the stored procedure in a C program or in a Java program, do not use this parameter. See “Running the Java sample programs” on page 16.

Example of a Net.Data search function call without ranking

The Net.Data ranking search function call returns rows 0 to 50 of search results from search term `stemmed form of "test"` and returns the columns ProductId, Title, Author, Format, and Year.

The results come from DB2 (data source ″1″).

```
@search(outTotalDocs, "stemmed form of \"test\"", "32000", "32000", "0", "50",
            "title",   "/home/myuserid/indexes", "/home/myuserid/tmp",
            "select ProductId, Title, Author, Format,  year(PublicationDate)
            as Year from db2nx.sample
            where productid in (%s)
            order by Title, Author for read only",
            "1", outWordCounts, outTable)
```

Note

The search with ranking works in a similar way to the above example.

Creating a Java search function to search a database

Calling the stored procedure `textSearch` from a Java program is very similar to the way you call the stored procedure in Net.Data. Refer to the sample Java source code provided in the `samples` directory. If you look at the source code in the sample file `NXSample.java`, three sections of code relevant to the definition and calling of the stored precedure can be identified:

- Line 85 and following: here you can modify the default parameters that are passed to the stored procedure.
- Line 171 and following: here you can see the call statement for the stored procedure. The parameters set in the previous section are passed on.
- Line 203 and following: defines the signature of the stored procedure.

The parameter descriptions used in the Net.Data example also apply to the Java example. For further information, see “Using your Net.Data search function” on page 28.

The sample source file `NXSampleRank.java` has a similar structure, but uses the stored procedure for search with ranking instead.

For an example of a Java call, see “Running the Java sample programs” on page 16.

Solving textSearch problems

To solve textSearch problems:

- Check the error log.
- For more information, do a trace.
- Check the error messages.
- Check ownership and permissions.
- Check shared memory.

Checking the error log

If an error occurs, information is written to a log file in the directory specified by the `tmpDirectory` parameter of the stored procedure. The name of the log file is the same as the index name and has the extension `.LOG`

For example, on a Unix system, if the temporary directory is `/tmp` and the index name is `COMMENT`, the log file is `/tmp/COMMENT.LOG`

Tracing

To get more detailed information during search, use the tracing variants of the search functions `textSearch_t` (without ranking) and `textSearch_rt` (with ranking).

The search trace function adds additional information to the log file. Note that the search function with tracing runs slower than without.

Error messages

If you get one of the following message during search:

- `SQL0104N An unexpected token "VALUES) as " was found ...`, check that the startRow is not equal to the result size. The startRow default value is 0.
- `DES7302N index does not exist`, check whether you issued a numeric search, but did not enable a numeric index.

Ownership and permissions

One of the more common problems during search is the incorrect ownership and permission of the index files. This is the index files directory structure (relative to the specified index directory):

```
./<INDXNAME>.shm
./<INDXNAME>/Fullmain/gtr.KEY/
                      /gtr.POS/
                      /gtr.LEY/
                      /gtr.QOS
```

```
/ItemMain/gtr.MEY/
/gtr.ROS
/gtr.NEY/
/gtr.SOS
```

For FASTRECOVERY indexes there are also the following files:

```
<INDXNAME>.SHM73
<INDXNAME>.SHM83
<INDXNAME>.SHM97
<INDXNAME>.SHM98 (if INCREMENTAL UPDATE enabled)
...
```

For an example, see `$DB2NX_INSTOWNERHOMEDIR/db2nx/indices` after running `nxsample`.

All files in this directory structure must be readable by the DB2 instance owner. If a different user ID is used for running the stored procedure, these files must also be readable by the owner of `$DB2NX_INSTOWNERHOMEDIR/sqllib/adm/.fenced`

Shared memory monitoring

As Net Search Extender makes extensive use of shared memory, this resource needs to be closely monitored to avoid paging and negative effects on other applications:

- For UNIX and Linux, use `vmstat`.
- For Windows NT, use `Task Manager` or other monitoring tools.

Identify shared memory segments belonging to indexes that you want to keep. Delete those that you do not want to keep by deactivating the corresponding indexes, or if the indexes are no longer required, by disabling the corresponding text columns.

Query problems

To avoid query problems:

- Ensure that the index is activated.
- Ensure that the query statement is not too big (the limit of returned data for a query is 32700 bytes, the DB2 limit on the size of the VALUES statement). If it is, reduce the number of returned rows using `maxhitcount`, and reduce the number of `optimized for` columns (this requires re-enabling of the database).
- Ensure that the directory name specified with the directory parameter during enable text column is the same as the `indexDirectory` input parameter of the textSearch procedure call.

- After a system crash on Windows NT, run `nxclean` to delete the `<index_name.shm>` file from the index directory.

Chapter 5. Administration overview

This chapter provides an overview of administration-related tasks. These include:

- Database and index monitoring
- Index maintenance
- Database and index backup
- Optimizing search performance

Most tasks require input from the application programmers who develop the database applications that search for and retrieve data.

Except for backup, all Net Search Extender administration tasks are carried out by entering the `db2nx` command. The command can only be issued on a DB2 server. It is assumed that the administrator, logged in as a DB2 instance owner, has the following environment variables set:

```
DB2DBDFT
DB2NX_INSTOWNERHOMEDIR
```

When issuing a `db2nx` command and specifying the optional database parameter, a database connection is made to that database. If the optional database parameter is not specified, a connection is made to $DB2DBDFT.

The section on optimizing search performance provides guidelines for both administrators and application developers using Net Search Extender to prepare data (that is, to create an index) and to implement search functions.

Monitoring the Net Search Extender environment

Net Search Extender has two administration commands to provide you with status information:

- GET STATUS for information about the status of the database
- GET INDEX STATUS for information about the status of a Net Search Extender index

Displaying the status of the database

When you need information about the status of the database, enter:

```
db2nx GET STATUS
```

This displays a list of created indexes. It shows which index belongs to which table column.

Example command:

```
db2nx get status database nxsample
```

This command returns:

```
Database is enabled for 'DB2 Net Search Extender'.

  Table DB2NX.SAMPLE is enabled
    IndexName   ColumnName
    ---------   ----------
    COMMENT     COMMENT
    TITLE       TITLE
    FPD         FORMAT_PUB_DATE

DES7800I  The command completed successfully.
```

Displaying the settings and the status of an index

When you need to determine which parameters you used during `enable text column` and the current status of an index, enter:

`db2nx GET INDEX STATUS` *index-name*

Unix command example:

```
db2nx get index status comment database nxsample
```

This command returns:

```
Settings of index 'COMMENT'

  Table                = 'DB2NX.SAMPLE'
  Column               = 'COMMENT'
  Index directory      = '/home/user1/db2nx/indices/'
  Temporary directory  = '/home/user1/db2nx/indices/'
  Key-column           = 'DOCID'
  Order by             = 'DOCID ASC '
  Optimize on:
    1.  = TITLE
    2.  = AUTHOR
    3.  = COMMENT
    4.  = FORMAT_PUB_DATE
    5.  = PRICE
    6.  = year ( last_updated )  as year_last_updated
  Numeric:
    1.  = PRICE
  Tags                 = no tags defined

  The Index is NOT enabled for Incremental Update

  Shared Memory Information :
    Internal Revision Number         : 72001
    (Re-)Creation Time               : Thu Mar 22 19:22:38 2001
```

```
    Existing Shared Memory Segments  : 3
    Size of SUPERBLOCK-Segment       : 8364 bytes
    Size of INODE-Segment            :  512 bytes
    Number of documents in SHM       : 8
    DATA-Segment 1 : allocated size  : 1290 bytes

In-Memory Table '/home/user1/db2nx/indices/COMMENT.SHM'
   ID    Address    Permissions    Creator  Group
m    393248 0x530dd80c --rw-r--r--    user1    staff      8364
m   1179689 0x490dd80c --rw-r--r--    user1    staff       512
m    262186 0x610dd80c --rw-r--r--    user1    staff      1290

DES7220I  Index is active.

DES7800I  The command completed successfully.
```

Note

You will see a slightly different output for indexes that were created with the FASTRECOVERY or INCREMENTAL options.

Maintaining indexes

These are the administration tasks for maintaining existing indexes:

- **Activating or deactivating an index**

  An index must be activated before you can search on it. This is done automatically when you create an index using the ENABLE TEXT COLUMN command. However, in the following cases you must activate an index explicitly:

  – When the index has been deactivated explicitly by the DEACTIVATE INDEX command
  – When the system has been restarted
  – When Net Search Extender has been stopped by the `NXSTOP` command (Windows NT only). Reactivate the index using `NXSTART`.

  To activate an index, run:

  ```
  db2nx ACTIVATE INDEX myIndex
  ```

  To deactivate an index, run:

  ```
  db2nx DEACTIVATE INDEX myIndex
  ```

  When an index is activated, considerable amounts of data may be loaded into memory. To prevent the system from running out of resources, you should explicitly deactivate indexes that are not being used. For example, there may be some applications that require two indexes, but never both

indexes at the same time. In such cases, you should switch indexes by deactivating one index and activating the other.

For large indexes, use the FASTRECOVERY option to minimize the time taken to activate an index.

See point 3 on page 23, "Calculate the amount of resources needed" for further information.

- **Disabling a text column to delete an index that is no longer needed**

  To disable a text column to delete an index, run:

  ```
  db2nx disable text column index-name
  ```

  Note

  As you can have multiple indexes on a text column, the command uses *index-name* to determine, which one(s) are deleted.

- **Disabling a database when all the indexes are no longer needed**

  To disable a database, run:

  ```
  db2nx disable database database-name
  ```

- **Updating an index**

  Update an index whenever the content of its associated text column changes. Net Search Extender does **not** automatically update indexes.

  To update an index, run:

  ```
  db2nx UPDATE INDEX myIndex
  ```

  When you update an index, the in-memory table associated with that index is recreated. This means that you cannot search the index while it is being updated.

  However, if you want to frequently update an index containing large volumes of data, see the following section, "Incremental Index Update".

Incremental Index Update

As Net Search Extender always indexes all the data in the text column, the update process is expensive in system resources for large volumes of data. If this data changes frequently, then updating the index can cause operational problems.

To overcome this constraint, use incremental index update to reduce the time required to update an index. Note that, as the index needs to be enabled for incremental update when it is first built, you are unable to change an existing index into an incrementally updatable index; see "ENABLE TEXT COLUMN" on page 60.

Creating an index with incremental index update enabled introduces the following changes to the database:

- Triggers are introduced to keep track of the database changes.
- An additional table is added; This is used to log changes to the database.

As the updating of indexes is not automatic, use the UPDATE INDEX command with the `INCREMENTAL` keyword, see "UPDATE INDEX" on page 77.

This only processes data additions, changes and deletions from the selected columns in the index, which considerably reduces the time required to update the index and therefore, the time that the index is unavailable for searching.

Backing up databases and indexes

After you have set up the DB2 database for Net Search Extender, it is advisable to do the following:

1. Back up the DB2 database using DB2 commands.
2. Copy the Net Search Extender index files from their index directory to a backup directory from where they can be restored.

Optimizing search performance

To optimize the search performance on a Net Search Extender enabled database, consider the following aspects of index creation and of the search call:

- **Optimize performance on certain table columns**

  The stored procedure `textSearch` returns a result table which contains information about the documents found by a search. The columns of that result table are either retrieved from the DB2 table or from an in-memory table. This is triggered by the `dataSource` parameter value.

  Retrieving the result table from memory can be much faster than retrieving it from the original table. "ENABLE TEXT COLUMN" on page 60 provides an OPTIMIZE ON parameter that lets you specify a list of columns to be stored in memory, but you should use this feature carefully to avoid running out of system resources.

- **Avoid memory paging**

  The most significant advantage of Net Search Extender is that you can search without accessing the hard disk of the database server. This is possible by holding the required data in memory. However, if more memory is needed than is provided by the hardware, the operating system starts paging out parts of memory to disk. This negates the advantage in performance.

Keep in mind that other applications may be running on your server that also use considerable amounts of memory. See item 3 on page 23, ″Calculate the amount of resources needed″ for further information.

- **Limit the search result**

  Limit the search result by setting the parameters `maxHitCount` and `maxIntermediateHitCount` of the stored procedure `textSearch`. This is useful when one of the search terms has a large number of matches. See “Using your Net.Data search function” on page 28 for more information.

- **Use Live Connection**

  If you are using Net.Data, use the Live Connection feature to prevent a new connection to the DB2 server from being re-established for every search call.

- **Use the search trace function only when necessary**

  If you do not need trace information, do not use the search trace function textSearch_t; instead, use the stored procedure textSearch. This function can be very useful, but it degrades search performance. It is described in “Solving textSearch problems” on page 33.

- **Update DB2 parameters for using large databases**

  If you are using large databases on a high-end, heavily used server with a large amount of memory, you should change the settings of some DB2 parameters.

  Check that the following are set to the requirements of your environment.

  Stop and restart DB2 after changing these parameters.

Table 1. Database parameters

Parameter	Recommended setting
dbheap	1200
pckcachesz	–1
maxappls	500
catalogcache_sz	256

Table 2. Data manager parameters

Parameter	Recommended setting
maxdari	500
keepdari	YES

Table 3. Environment variables

Environment variable	Recommended setting
DB2_NO_PKG_LOCK	ON

- **Disk layout**

  For better performance during indexing and searching, it is advisable to distribute the DB2 tables, the DB2 indexes and the Net Search Extender indexes across separate physical disks.

Administration overview

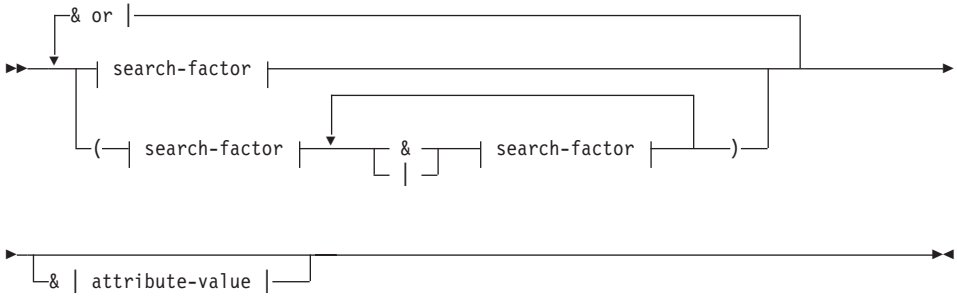
Part 2. Reference

Chapter 6. Search syntax for the searchTerm parameter

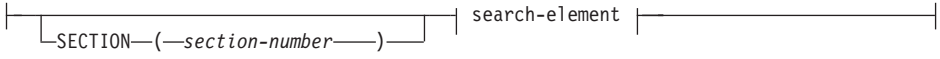
The search syntax described here is for the searchTerm parameter of the Net Search Extender stored-procedure call described in "Chapter 4. Using Net Search Extender" on page 21.

Search argument

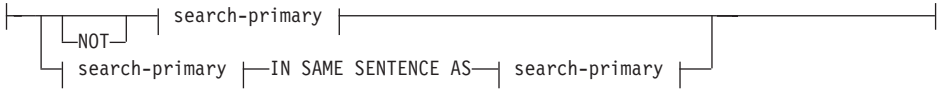
Command syntax



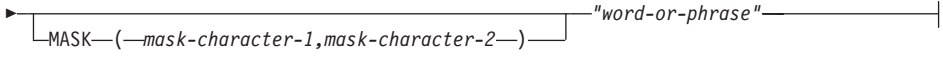
search-factor:



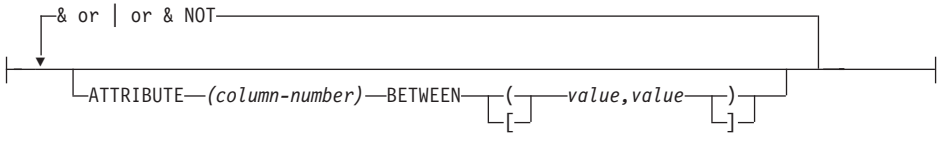
search-element:



search-primary:



attribute-value:



Command parameters

SECTION *(section-number)*

A keyword used to specify a section that the search is to be restricted to. To specify a section, use one of the section numbers displayed when running GET INDEX STATUS. The section IDs are determined by the the tag order during ENABLE TEXT COLUMN.

search-factor

An operand that can be combined with other operands to form a search argument. The evaluation order is from left to right. The logical AND (&) operator binds stronger than the logical OR (|) operator. Example:

```
"passenger" & "vehicle" | "transport" & "public"
```

is evaluated as:

```
("passenger" & "vehicle") | ("transport" & "public")
```

To search for:

```
"passenger" & ("vehicle" | "transport") & "public"
```

you must include the parentheses as shown.

NOT An operator that lets you exclude from your search text documents that contain a particular term.

IN SAME SENTENCE AS

A keyword that lets you search for a combination of terms occurring in the same sentence.

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.

FUZZY FORM OF *match-level*

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 `econony`.

The *match-level* is a value from 1 to 100, where 100 means least fuzzy (exact) and 1 means most fuzzy.

"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.

Masking characters (also known as ″wildcard″ characters)

A word can contain masking characters. You can use the `MASK` parameter to specify which characters are to be the masking characters. The default masking characters are:

% (percent)
Represents any number of arbitrary characters. If a word consists of a single %, then it represents an optional word of any length.

_ (underscore)
Represents any single arbitrary character.

A word cannot be composed exclusively of masking characters, except when a single % is used to represent an optional word.

MASK *(mask-character-1,mask-character-2)*
Masking characters that can be used in the search primary ″word or phrase″. For example: `MASK (*,?)`

mask-character-1 specifies the character that is to mask any number of arbitrary characters.

mask-character-2 specifies the character that is to mask any single arbitrary character.

ATTRIBUTE *(column-number)* **BETWEEN** *(value,value)*

You can add numeric search terms that limit the results of the query. However, numeric search terms can only be used in the following ways:

1. The numeric search terms must follow the last text search term in the searchTerm parameter.
2. The relational operators & (AND), | (OR), NOT can be used between numeric search terms. They must be separated from the numeric search terms by at least one blank space.

Note

The value of *column-number* corresponds to the position of the column name in the NUMERIC option during ENABLE TEXT COLUMN. If you used the following command for indexing, this can be seen below:

```
ENABLE TEXT COLUMN...NUMERIC (PRICE, WEIGHT, NUMBER_IN_STOCK)
```

you would search for items costing between $5 and $9, yet weighing less than 7kgs using

```
"... &ATTRIBUTE(1) BETWEEN [5.00,9.00] &ATTRIBUTE(2) BETWEEN [0,7]"
```

The brackets define the lower and upper limits of the interval. The left value defines the lower range value, the right the higher range value.

The brackets also define how the values inside are used. When searching for ranges of numbers, you can specify that the search **includes** the minimum or maximum values by using square brackets. To indicate exclusive matching of the minimum or maximum values (these values are **not included**), use parentheses.

- (means >x
- [means >=x
-) means <x
-] means <=x

The brackets can be paired in any combination ([with], [with), and (with], for example). Note that ranking is only applied to the results of the text portion of a given query.

'MIN' and 'MAX' can also be used as minimum or maximum values of a range to bound a search by the smallest or largest values.

When searching for exact matches against a numeric column, specify the desired number as both the minimum and maximum value, bounded by square brackets.

Below are some sample queries with further explanations of the numeric search function.

"book" & ATTRIBUTE (1) BETWEEN [1234,1234]

This search returns all the rows where the text column contains ″book″ and where the first numerically indexed column contains the value 1234 in that same row.

″book″ & ATTRIBUTE (2) BETWEEN [MIN,20.00)

This search returns all the rows where the text column contains ″book″ and where the second numerically indexed column contains values within the range MIN and 20.00, but **not equal to** 20.00.

Examples of query strings:

Search for documents containing `emergency` in section 5, and containing `security department` with 60% or more matching.

```
section (5) "emergency" & fuzzy form of 60 "security department"
```

Search for either `birds` or `nature`, and also `government`.

```
( "birds"  |  "nature" ) &  "government"
```

Search for inflectional endings of the word `shock`, such as `shocked`, `shocking`, and so on:

```
stemmed form of "shock"
```

Find two words in the same sentence (this assumes that the sentences are separated by periods):

```
"computer" in same sentence as "book"
```

Use masking characters to find documents that contain words that begin with `night` and also contain words like `dreams`, `dreamy`, and so on:

```
"night%" & "dream_"
```

Perform the same search, but change the masking character:

```
mask (*,?) "night*" & "dream?"
```

Note

Rows with NULL values in numeric columns cannot be searched for using the attribute value parameters.

Chapter 7. Administration commands

Command	Purpose	Page
ACTIVATE INDEX	Makes search possible after the index has been deactivated, or after the system has been rebooted	55
DEACTIVATE INDEX	Frees memory occupied by data associated with the index	56
DISABLE DATABASE	Deletes all indexes associated with a database; the database is no longer available for use by Net Search Extender	57
DISABLE TEXT COLUMN	Deactivates and deletes an index	58
ENABLE DATABASE	Prepares a database for use by Net Search Extender	59
ENABLE TEXT COLUMN	Prepares a text column for use by Net Search Extender and creates an individual text index for the column	60
GET INDEX STATUS	Displays the index settings and the status of an index	67
GET STATUS	Displays the status of a database	69
HELP	Displays help information	70
NXICRT (Unix only)	Creates a Net Search Extender instance	71
NXIDROP (Unix only)	Drops a Net Search Extender instance	72
NXSTART (Windows NT only)	Starts Net Search Extender	73
NXSERVICE	Copies DB2 and Net Search Extender and setup information to a file	74
NXSTOP (Windows NT only)	Stops Net Search Extender	76
UPDATE INDEX	Updates and activates an index	77

This chapter is a reference for the Net Search Extender administration commands. While "Chapter 5. Administration overview" on page 37 provides an overview of Net Search Extender administration, this chapter includes information about the syntax of each command and describes the related parameters.

As in DB2, you enter the Net Search Extender command at the command prompt. You prefix each command with `db2nx` except for `HELP`, `NXICRT`, `NXIDROP`, `NXSTART`,`NXSTOP` and `NXSERVICE`.

Except for the `HELP` command, each administration command always establishes a database connection to either the database specified as an optional parameter, or to the one specified by the environment variable `DB2DBDFT`.

Environment variables

The environment variables set the default values for the administration commands.

DB2DBDFT
Default database name. The name of the DB2 UDB database that is assumed if no database name is specified.

DB2NX_INSTOWNERHOMEDIR
Instance owner's home directory.

DB2NXPATH (Windows NT only)
Installation directory.

Tracing

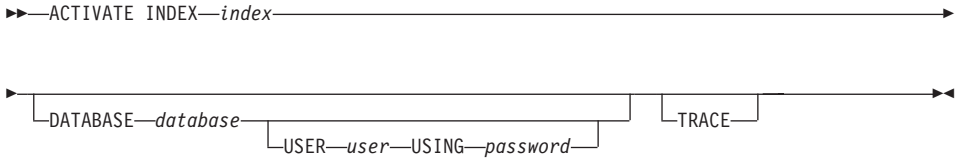
All `DB2NX` commands can be used in trace mode if the command is followed by an extra `trace` parameter. This parameter triggers the output of additional status and progress information. Note that using `trace` with the `DB2NX ENABLE TEXT COLUMN` command can produce potentially huge amounts of trace output, depending on the amount of data to be indexed.

ACTIVATE INDEX

This command makes search on an index possible after the index has been explicitly deactivated by the DEACTIVATE INDEX command, or after the system has been restarted.

The command reads the key column and all the OPTIMIZE ON columns from DB2 into the shared memory. If the index was created with the FASTRECOVERY option, this data is read from the FASTRECOVERY data files for faster activation, rather than from DB2.

Command syntax



Command parameters

index The index name specified when the index was created using the enable text column command.

DATABASE database
The name of the database you want to work with. If you do not specify this parameter, the value of the environment variable DB2DBDFT is used.

USER user
The user ID of the DB2 instance with DBADM authority for the database you want to work with.

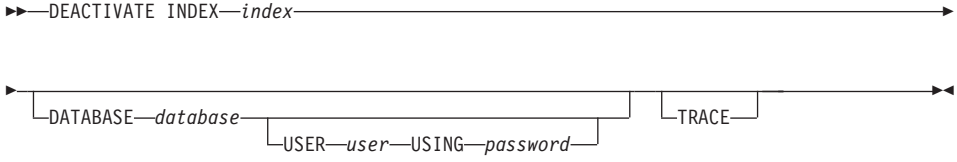
USING password
The password for the DB2 instance.

TRACE
Activates tracing.

DEACTIVATE INDEX

This command frees the shared memory occupied by an index. After running this command, you cannot search on the specified index until you run the ACTIVATE INDEX command. This command does not affect the index itself, but only the contents of the in-memory table.

Command syntax



Command parameters

index The index name specified when the index was created using the enable text column command.

DATABASE database
The name of the database you want to work with. If you do not specify this parameter, the value of the environment variable DB2DBDFT is used.

USER user
The user ID of the DB2 instance with DBADM authority for the database you want to work with.

USING password
The password for the DB2 instance.

TRACE
Activates tracing.

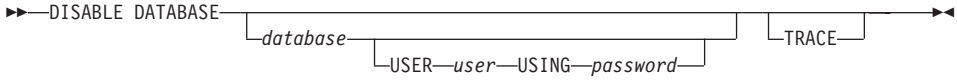
Usage notes

To prevent the system from running out of resources, you should explicitly deactivate indexes that are not currently being used, especially indexes with the `optimize on` parameter specified when the index was created. See point 3 on page 23, "Calculate the amount of resources needed" for further information.

DISABLE DATABASE

This command resets any preparation work done by Net Search Extender for a database and disables all its text columns for use by Net Search Extender. All index files for this database are removed from disk.

Command syntax



Command parameters

database
The name of the database you want to work with. If you do not specify this parameter, the value of the environment variable DB2DBDFT is used.

USER user
The user ID of the DB2 instance with DBADM authority for the database you are disabling.

USING password
The password for the DB2 instance.

TRACE
Activates tracing.

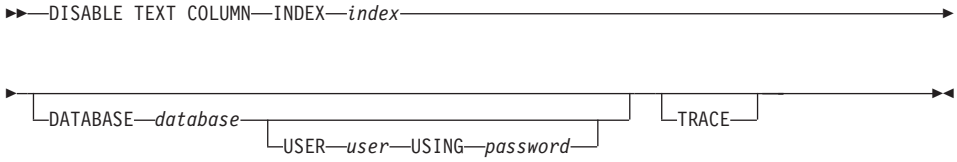
Usage notes

This command resets the connected database so that it can no longer be searched by Net Search Extender. All Net Search Extender index files are deleted. To re-enable the database for Net Search Extender searching, use the ENABLE DATABASE command.

DISABLE TEXT COLUMN

This command deletes a Net Search Extender index and frees memory occupied by the index.

Command syntax



Command parameters

INDEX index
The unique index name specified when the index was created using the enable text column command.

DATABASE database
The name of the database you want to work with. If you do not specify this parameter, the value of the environment variable DB2DBDFT is used.

USER user
The user ID of the DB2 instance with DBADM authority for the database you want to work with.

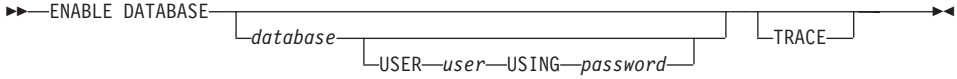
USING password
The password for the DB2 instance.

TRACE
Activates tracing.

ENABLE DATABASE

This command prepares a database to be used for text search by Net Search Extender.

Command syntax



Command parameters

database
The name of the database you want to work with. If you do not specify this parameter, the value of the environment variable DB2DBDFT is used.

USER user
The user ID of the DB2 instance with DBADM authority for the database you are enabling.

USING password
The password for the DB2 instance.

TRACE
Activates tracing.

Usage notes

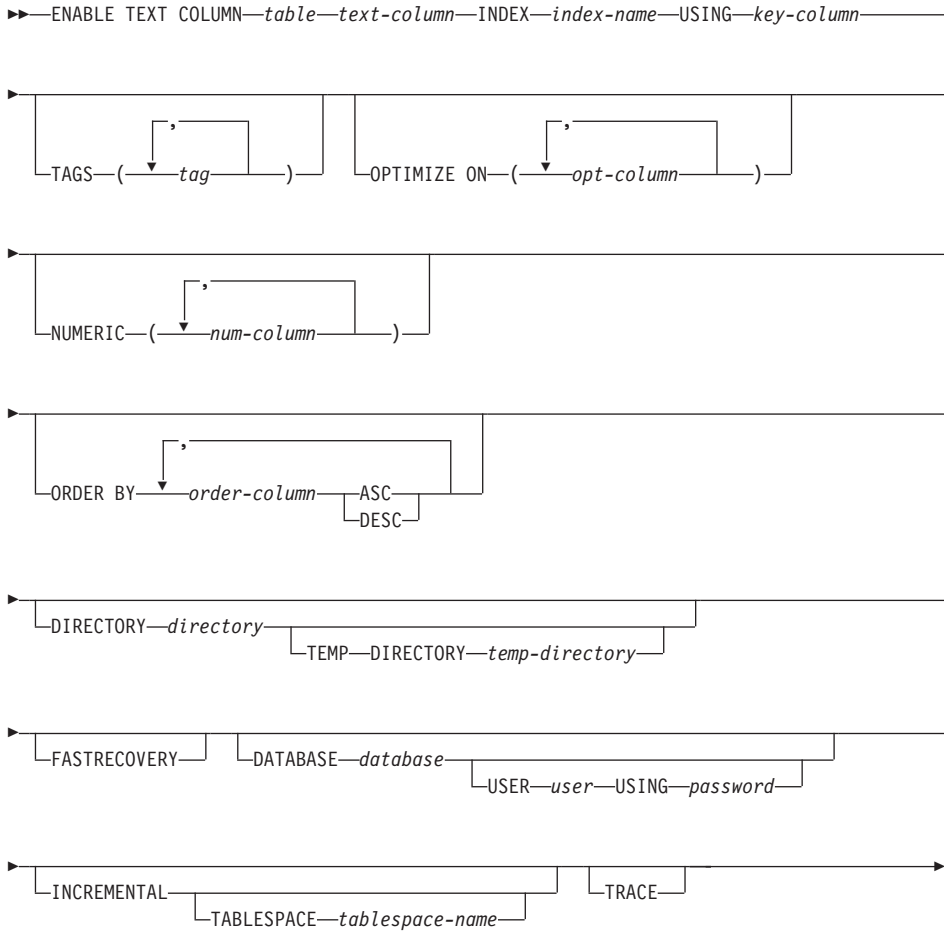
The `enable database` command creates the following tables:

- db2nx.fpcolumn
- db2nx.fptags
- db2nx.fpmemory
- db2nx.fpnumber

ENABLE TEXT COLUMN

This command creates an index for the specified text column.

Command syntax



Command parameters

table The name of the text table in the connected database that contains the column to be enabled. The name must include an explicit schema name (qualifier) unless the schema name is the same as your user ID.

text-column
The name of the column to be enabled for use by Net Search Extender. This column must be of the type CHAR, VARCHAR, LONG VARCHAR, or CLOB.

INDEX index-name
A name to be given to the index. The name must be unique in the specified directory and not longer than 8 characters.

USING key-column
The name of the key column used to establish a relation between the indexed documents and the database rows. In combination with the keyword INCREMENTAL, the key-column must be unique as it must be possible to join a record in shared memory with a record in the source table. If possible, use the primary key of the table.

The key column must be of type INT, SMALLINT, BIGINT, CHAR, VARCHAR, TIMESTAMP, DATE, or TIME.

Note that the contents of the *key-column* are held in an in-memory table while the index is activated. If the width of your key column is very large, you may run into system limitations. See point 3 on page 23, "Calculate the amount of resources needed" for further information.

TAGS tag

The names of up to five tags specified in the documents to support sections. If the format looks like:

```
xtitlex Document Title...

xbodyx Main text of document...

xfooterx Some footer information...
```

where there is a space after the tags `xtitlex`, `xbodyx`, and `xfooterx`, and there are two blank lines separating each section, then this is what the `TAGS` portion of the command would look like:

```
... TAGS (xtitlex, xbodyx, xfooterx)
```

Note
Tags that would appear as regular words in the documents should be avoided.

A TAG value must be specified using only SBCS characters.

The section number specified as the searchTerm parameter for the Net Search Extender stored procedure call corresponds to the order of the tags. In this example, `xtitlex` becomes 1, `xbodyx` becomes 2, and `xfooterx` becomes 3. See "Chapter 6. Search syntax for the searchTerm parameter" on page 47 for the use of the section number.

Use the `get index status command` to get a list of the tags you specified during the creation of an existing index.

OPTIMIZE ON opt-column

Up to 22 columns to be held in an in-memory table of the database server. This enables the stored procedure to retrieve them in a result table without accessing the original DB2 table. This feature is an important contributor to the high search performance of Net Search Extender.

To take advantage of this feature, set the parameter `dataSource` of the `textSearch` stored procedure to 0 when issuing the search. If you don't do this, the results are retrieved from DB2.

Note that the memory resources of your database server are limited, and that other applications may be running on your server that are using parts of memory. So it is important to estimate the amount of memory required for your indexes; see point 3 on page 23, "Calculate the amount of resources needed" for further information.

Instead of specifying an existing table column, you can instead specify an SQL expression. You can specify any expression allowed in the SELECT clause of an SQL query on the table containing the documents to be indexed.

Example: The following command creates an additional index for the sample table `db2nx.sample`, which is created by the sample program `nxsample`:

```
db2nx "ENABLE TEXT COLUMN db2nx.sample comment
                INDEX sample
                USING docid
                OPTIMIZE ON (title,{SUBSTR(comment,1,30)
                            as commentheader})"
```

Note that an SQL expression must be surrounded by braces { }.

NUMERIC num-column

A keyword to limit text search results to a range of values, or equal to a value. For example, search on the word `dog` but only return results between dates 19981201 and 20001201. Alternatively, search on `history` but only return books with price less than $50.00.

The following data types are supported:

- INTEGER
- DECIMAL
- NUMERIC
- SMALLINT
- DOUBLE

- CHAR
- VARCHAR

Note that with CHAR and VARCHAR data types, the contents of the column must be valid numeric strings. Also note that REAL and FLOAT are not supported.

For example, a database column (data type INTEGER), called **date** would be entered as:

```
...NUMERIC (date)
```

Note

Single or multiple numeric columns can be used in the command. However, ensure that their order corresponds to the first, second, and so on numeric fields of the search term.

The maximum number of numeric columns is 20.

Also note that the JULIAN_DAY scalar function can be applied to column type DATE or TIMESTAMP to convert the contents to INTEGER. See the *SQL Reference* documentation for further information.

ORDER BY order-column ASC or DESC

The columns used to specify a sequence during indexing, so that documents can be retrieved from the index in ascending or descending sorted order. When retrieving data from DB2 rather than from the in-memory table, this order must be used during a search to return the results in the requested order.

Note

The sort order cannot be preserved after an incremental update.

Also note that if order mask columns have NULL values, then according to SQL standards these always rank highest in any given sort order.

DIRECTORY directory

The directory where the index is to be stored. If specified, the directory must exist. If you don't specify a directory, the default directory `db2nx/indices` is used.

TEMP DIRECTORY temp-directory
The directory where temporary index files are to be stored. If specified, the directory must exist. In Unix systems, if you don't specify a directory, the default directory `/tmp` is used. If you don't specify a directory in Windows NT, the index directory is used.

FASTRECOVERY
A keyword to enable faster response times for the `DB2NX ACTIVATE INDEX` command after a shut down or server crash. When this keyword is set, Net Search Extender stores data in memory-mapped files and not directly in shared memory. Note that the same size calculations apply for additional disk space as when calculating the amount of shared memory required; See point 3 on page 23, "Calculate the amount of resources needed" for further information. Also note that indexing and search performance is slower when this keyword is set.

DATABASE database
The name of the database you want to work with. If you do not specify this parameter, the value of the environment variable `DB2DBDFT` is used.

USER user
The user ID of the DB2 instance with DBADM authority for the database that contains the table.

USING password
The password for the DB2 instance.

INCREMENTAL
A keyword to enable the Net Search Extender index for incremental update. Triggers and a log table are created to identify and protocol changes within the database for update, delete and change requests. The in-memory tables are organized differently for incremental update. When specifing this option, a subsequent UPDATE INDEX command only selects added, deleted or changed rows in the database for the indexing process. If INCREMENTAL is not specified when enabling a text column, UPDATE INDEX always indexes the entire text column.

> **Note**
>
> **Impact during Administration**: Due to additional processing in logging added, deleted, or changed rows in the database, all subsequent database inserts, deletes, or updates are expected to be slower. See also "Incremental Index Update" on page 40.
>
> **Impact during query**: After an update the query performance is lower for all subsequent queries expecting the correct sort order. See also "UPDATE INDEX" on page 77 for information on how to minimize this problem.

TABLESPACE tablespace-name

A keyword to define the DB2 tablespace where the log table will be created. If not defined, the log table will be created with DB2 defaults, which is the first tablespace created by the user. The tablespace must be created before the command is executed.

TRACE

Activates tracing.

Note

The ENABLE TEXT COLUMN command writes progress information to stdout indicating the amount of data indexed with time stamps to track performance information.

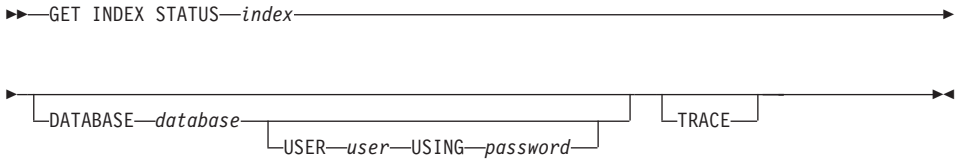
Example:

```
Fri Dec  8 11:56:51 2000 Full Start...
Fri Dec  8 11:56:51 2000 Item Start...
Calculating In-Memory Table size...
0

Reading data from DB2NX.SAMPLE and creating index...
0
Completed reading database
Fri Dec  8 11:56:52 2000 Full Flush <DOC :   6>< 1M>- Sort .
                                                    - Writing -  .
Fri Dec  8 11:56:52 2000 Full X_start...
Fri Dec  8 11:56:52 2000 Full Flush <PART :101>< 1M>- Sort .
                                                    - Writing -  .
Fri Dec  8 11:56:52 2000 Full End
-------------------------------------------------------------------
Sort1                           0 (sec)
Sort2                           0 (sec)
Write                           0 (sec)
Other(including caller's time)  1 (sec)
Total                           1 (sec)
```

GET INDEX STATUS

This command returns information about the index, including its current status.

Command syntax



Command parameters

index The index name specified when the index was created using the enable text column command.

DATABASE database
The name of the database you want to work with. If you do not specify this parameter, the value of the environment variable DB2DBDFT is used.

USER user
The user ID of the DB2 instance with DBADM authority for the database you want to work with.

USING password
The password for the DB2 instance.

TRACE
Activates tracing.

Examples

Example command:

```
db2nx get index status comment database nxsample
```

Example command output:

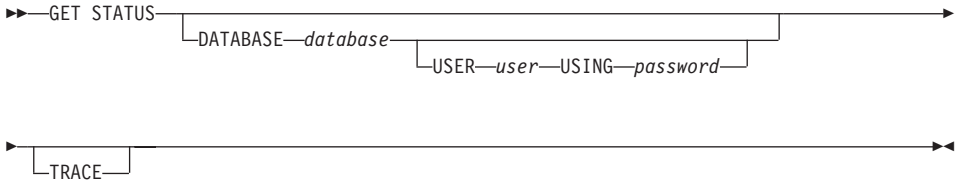
```
Settings of index 'COMMENT'

  Table                = 'DB2NX.SAMPLE'
  Column               = 'COMMENT'
  Index directory      = '/home/user1/db2nx/indices/'
  Temporary directory  = '/home/user1/db2nx/indices/'
  Key-column           = 'DOCID'
  Order by             = 'DOCID ASC '
  Optimize on:
    1.  = TITLE
```

```
    2.  = AUTHOR
    3.  = COMMENT
    4.  = FORMAT_PUB_DATE
    5.  = PRICE
    6.  = year ( last_updated )  as year_last_updated
  Numeric:
    1.  = PRICE
  Tags                 = no tags defined

  The Index is NOT enabled for Incremental Update

  Shared Memory Information :
    Internal Revision Number         : 72001
    (Re-)Creation Time               : Thu Mar 22 19:22:38 2001
    Existing Shared Memory Segments  : 3
    Size of SUPERBLOCK-Segment       : 8364 bytes
    Size of INODE-Segment            :  512 bytes
    Number of documents in SHM       : 8
    DATA-Segment 1 : allocated size  : 1290 bytes

In-Memory Table '/home/user1/db2nx/indices/COMMENT.SHM'
   ID    Address    Permissions    Creator  Group
m    393248 0x530dd80c --rw-r--r--    user1    staff      8364
m   1179689 0x490dd80c --rw-r--r--    user1    staff       512
m    262186 0x610dd80c --rw-r--r--    user1    staff      1290

DES7220I  Index is active.

DES7800I  The command completed successfully.
```

GET STATUS

This command displays information about the enabled status of a database, and shows a list of its indexes.

Command syntax



Command parameters

database
The name of the database you want to work with. If you do not specify this parameter, the value of the environment variable DB2DBDFT is used.

USER user
The user ID of the DB2 instance with DBADM authority for the database for which you are getting the status.

USING password
The password for the DB2 instance.

TRACE
Activates tracing.

Examples

Example command:

```
db2nx get status database nxsample
```

Example command output:

```
Database is enabled for 'DB2 Net Search Extender'.

  Table DB2NX.SAMPLE is enabled
    IndexName   ColumnName
    ---------   ----------
    COMMENT     COMMENT
    TITLE       TITLE
    FPD         FORMAT_PUB_DATE

DES7800I  The command completed successfully.
```

HELP

This command displays a list of the administration commands provided by Net Search Extender.

Command syntax

Command parameters

None.

Examples

The output of the `db2nx help` command looks like this:

```
List of db2nx commands
HELP
ENABLE DATABASE       [database [USER1 user1 USING password]]
                      [TABLESPACE tablespace]
DISABLE DATABASE      [database [USER1 user1 USING password]]
GET STATUS            [DATABASE database [USER1 user1 USING password]]
ENABLE TEXT COLUMN    table text-column INDEX index USING key-column
                      [TAGS ( tag [{,tag} ...] ) ]
                      [OPTIMIZE ON ( opt-column [ {,opt-column} ...] ) ]
                      [NUMERIC ( num-column [ {,num-column} ...] ) ]
                      [ORDER BY column {ASC|DESC} [{,column {ASC|DESC}} ... ]]
                      [DIRECTORY directory [TEMP DIRECTORY temp-directory ]]
                      [FASTRECOVERY]
                      [DATABASE database [USER1 user1 USING password]]
                      [INCREMENTAL [TABLESPACE tablespace]]
DISABLE TEXT COLUMN   INDEX index [DATABASE database [USER user USING password]]
UPDATE INDEX          index [DATABASE database [USER1 user1 USING password]]
                      [INCREMENTAL [COMMITCOUNT commitcount]]|[REORG]
GET INDEX STATUS      index [DATABASE database [USER1 user1 USING password]]
ACTIVATE INDEX        index [DATABASE database [USER1 user1 USING password]]
DEACTIVATE INDEX      index [DATABASE database [USER1 user1 USING password]]
```

NXICRT (Unix only)

This command creates a Net Search Extender instance.

Command syntax

Command parameters

instance-name
The user ID of the owner of the instance to be created.

Usage notes

Enabling text tables or columns is possible only on nodes where you have created a Net Search Extender instance.

This command must not be prefixed with `db2nx`.

Also note that after instance creation only the `.profile` will be updated. If the instance owner works with a UNIX shell not using `.profile`, the following environment variables should be set:

```
DB2NX_INSTOWNERHOMEDIR=/<your home directory>/
       PATH=$PATH:DB2NX_INSTOWNERHOMEDIR/DB2nx/bin

where $PATH = your existing PATH variable
```

NXIDROP (Unix only)

This command drops a Net Search Extender instance together with all its indexes created in the default directory. Note that if the indexes are not in the default directory, they will not be dropped.

Command syntax

Command parameters

instance-name
The name of the instance to be dropped.

Usage notes

Before dropping an instance, disable any databases that are enabled for it. Note that this also deletes index files that are not in the default directory.

This command must not be prefixed with `db2nx`.

Note
This command deletes **all** files in `$HOME/db2nx` including all the index files created there.

NXSTART (Windows NT only)

This command starts a background process to enable the use of shared memory in the stored procedure.

Command syntax

Command parameters

None.

Usage notes

If this process is stopped or not started, the indexes cannot be activated and no search is possible.

As `nxstop` automatically deactivates your indexes (by shutting down shared memory), you have to use `db2nx activate index` for each index after restarting. See "ACTIVATE INDEX" on page 55 for further information.

This command must not be prefixed with `db2nx`.

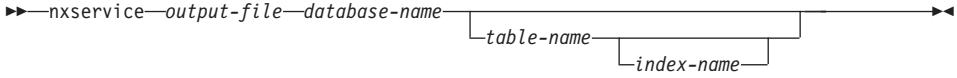
NXSERVICE

This command copies information about the current DB2 and Net Search Extender setup to a file, which can be provided to IBM support for problem analysis.

Authorization

DB2 instance owner

Command syntax



Command parameters

output-file
The path and file name of the output file to be provided to IBM support.

database-name
The name of the database in which the problem occurs.

table-name
The name of the table that is related to, or causes the problem. If the problem occurs during the creation of the Net Search Extender instance, or when enabling your database for Net Search Extender, you do not need this parameter because the problem is not related to a particular table.

index-name
The name of the index where the problem occurs. This parameter should be included with the table-name, if the problems are related to one or more text indexes. If the command returns an error message stating that the index-name does not exist, run the command again without the index-name parameter.

Examples

When performing a search on index `COMMENT`, on table `DB2NX.SAMPLE` in database `SAMPLE`, you get the error message DES7525N, "In-memory table `DB2NX.SAMPLE` could not be attached". Issue the following command:

```
nxservice service.out SAMPLE DB2NX.SAMPLE COMMENT
```

Then provide the file `service.out`, the command that caused the error, and the error message to IBM support.

Usage notes

The command does not apply any changes to your system. The following information is stored in the output file:

- Information about the installed levels of DB2 and Net Search Extender.
- Details on the installed Net Search Extender files.
- Contents of the internal Net Search Extender administration tables.
- A list of all text indexes created for the current database.
- Database and database manager configurations.
- The session settings for all environment variables.
- The structure of the database table provided with the table parameter.
- The status and settings of the text index provided.

The command does not save any information stored in the tables you created text indexes for. The output file is a simple text file.

NXSTOP (Windows NT only)

This command stops the background process started by the NXSTART command.

Command syntax

Command parameters

None.

Usage notes

After this command, the indexes cannot be activated and no search is possible.

As `nxstop` automatically deactivates your indexes (by shutting down shared memory), you have to use `db2nx activate index` for each index after restarting. See "ACTIVATE INDEX" on page 55 for further information.

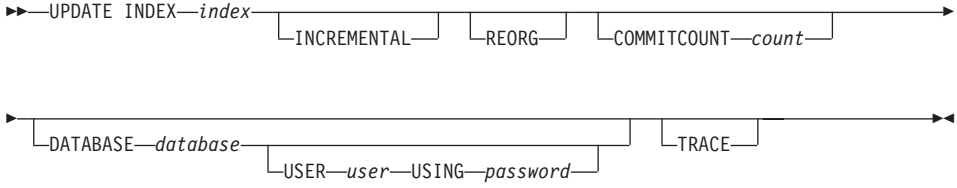
This command must not be prefixed with `db2nx`.

UPDATE INDEX

This command starts indexing using the same settings defined during ENABLE TEXT COLUMN. Using this command without any specific options creates the index files from scratch.

Performance consideration for incremental index update: use this command for a complete reorganization of the in-memory and index-data for keeping high query performance whenever the ORDER BY option is used during ENABLE TEXT COLUMN.

Command syntax



Command parameters

index The index name specified when the index was created using the enable text column command.

INCREMENTAL
A keyword to specify that only added, deleted or changed rows in the database are used for the in-memory and index update. Additional shared memory areas are used for in-memory usage and additional index files are used for the text-search engine. Depending on the amount of data and number of incremental index update commands, the internal system must be reorganized to reduce memory usage and to maintain high indexing speed and query performance.

A REORG is undertaken whenever internal thresholds are exceeded. See also the REORG keyword.

The physical sort order specified by the ORDER BY option of the ENABLE TEXT COLUMN is not retained by the INCREMENTAL index update. As the correct sort order is calculated during search, perform a complete index update to avoid any performance degradation.

REORG
A keyword to reorganize the in-memory data and index-data. Shared memory is optimized and the additional index files are merged with

the main index files. To be effective, the INCREMENTAL keyword must have been specified during ENABLE TEXT COLUMN, otherwise this keyword is ignored.

The correct sort order must still be calculated during search. A REORG is also triggered by an INCREMENTAL index update whenever internal thresholds are exceeded. See also INCREMENTAL index update.

COMMITCOUNT count
A keyword to define the number of log records to be deleted from the log table after command completion and before a COMMIT command is performed. If no commitcount is set, all the records are deleted. If there are too many records in the log table, this may cause a LOG-FILE-FULL condition. If this occurs, specify a lower count value and execute the command again.

DATABASE database
The name of the database you want to work with. If you do not specify this parameter, the value of the environment variable DB2DBDFT is used.

USER user
The user ID of the DB2 instance with DBADM authority for the database you want to work with.

USING password
The password for the DB2 instance.

TRACE
Activates tracing.

As with the `ENABLE TEXT COLUMN` command, using `TRACE` can produce huge amounts of trace output depending on the amount of changed data to be indexed.

Usage notes

Because updating an index makes it unavailable for search, your application is responsible for not allowing searches during updates. For example, if your application provides the results of text queries over the Web, then the search pages of the Web site should not be available during the update period.

If permanent availability of the search function is an issue, create two indexes and switch between them during UPDATE INDEX.

Chapter 8. Messages

Each message has a message identifier that consists of a prefix (DES), the message number, and a suffix letter.

I Information message

N Error (or "negative") message

Note that the message portion of the sqlca structure, which is used to pass the error message back to the client, holds a maximum of 70 characters, so some messages might be truncated. However, the full error message gets logged to the log file.

DES0249N A DB2 Net Search Extender internal error has occurred. Line number: '*nnn*'.

Explanation: An unexpected internal error occurred in the Net Search Extender demon.

What to do:

1. Check your system resources and retry the last command.
2. Try using `nxstop` to stop Net Search Extender, then reactivate all indexes again.

DES0250N A DB2 Net Search Extender internal error has occurred. Reason code: '*rc* '.

Explanation: An unexpected internal error occurred.

What to do:

1. Check your system resources and retry the last command.
2. Use the operating system reason code to solve the problem.

DES0251N The syntax of the command is: DESFPDEM {START | STOP} DESFPDEM {START | STOP | POST}

Explanation: The command syntax is incorrect.

What to do: Change the command syntax and try again.

DES0255N DB2 Net Search Extender has already been started.

Explanation: The Net Search Extender is already running.

What to do: No action required

DES0256N No system resources. Reason code: '*rc*'.

Explanation: No more system resources are available.

What to do: Use the operating system return code to solve the problem.

DES0257N Memory allocation failed. Line number: '*nnn* '.

Explanation: A system call to get memory failed.

What to do: Check the system resources and retry the last command.

DES0259N DB2 Net Search Extender has not been started.

Explanation: To work with Net Search Extender you first have to start it.

What to do: Call `nxstart` to start Net Search Extender.

DES0261N The environment variable '*variablename*' is not set. See DB2 Net Search Extender documentation for more information.

Explanation: The specified environment variable has not been set.

What to do: Check the system settings. You may need to restart the operating system.

DES0263N DB2 Net Search Extender demon has abnormally terminated.

Explanation: Caused by an unexpected error.

What to do: Check the system settings. You may need to restart the operating system.

DES0264N Function *functionname* ended with return code *rc*.

Explanation: A function ended with the specified return code.

What to do: Use the return code of the function to solve the problem.

DES7000N Failed on database open.

Explanation: An attempt to connect to a database failed.

What to do: Change the specified database name or check the environment DB2DBDFT variable.

DES7001N Failed on disable database.

Explanation: A problem occurred with the administration tables while disabling a database.

What to do: Check that all the administration tables are available and that the entries can be read by this user ID.

DES7101N Memory could not be allocated.

Explanation: No storage could be reserved for the application.

What to do: Increase the paging space.

DES7103N In-memory table *table* could not be allocated.

Explanation: No shared memory could be reserved for the application.

What to do: Decrease the number of `optimize on` columns or call DEACTIVATE INDEX for indexes that are no longer required.

DES7104N In-Memory Table init error.

Explanation: The In-Memory Table could not be initialized.

What to do: Check if the system limits for shared resources are reached and, if necessary, free additional resources.

DES7105I No data found in table.

Explanation: No data was found in the specified table. No table rows have been indexed.

What to do: Check the table to ensure that it contains data.

DES7106N The column *columnname* is optimized more than once. Remove this column from the optimized section.

Explanation: Column *columnname* occurs more than once in the optimized section.

What to do: Remove one of the occurrences of

the column from the optimized section.

DES7108N No column was found to join with the index.

Explanation: The key column following the USING tag is invalid or could not be used.

What to do: Check the key column following the USING tag, and, if necessary, change it.

DES7109N Invalid row count *rowcount* **returned for DB2 table** *schema.tablename***.**

Explanation: The row count is invalid. It is either zero or greater than 539 251 480.

What to do: Use a different table or change the number of rows to be in the above range.

DES7110N Data Cache Overflow.

Explanation: DB2 Net Search Extender has not allocated enough memory for the user data.

What to do: Try to reduce the number of columns specified in the 'optimize on' brackets.

DES7111N Inode block Overflow.

Explanation: DB2 Net Search Extender has not allocated enough memory for the user data.

What to do: Try to reduce the number of columns specified in the 'optimize on' brackets.

DES7201N *token* **is unexpected. Check the command syntax.**

Explanation: An unexpected token was found.

What to do: Check the command syntax.

DES7202N Parameter *parameter* **is too long.**

Explanation: The specified parameter is out of range.

What to do: Specify the parameter using a valid length.

DES7203N You reached the maximum number of OPTIMIZE ON columns.

Explanation: You specified more columns than allowed by the OPTIMIZE ON parameter.

What to do: Decrease the number of OPTIMIZE ON columns.

DES7204N You reached the maximum number of tags.

Explanation: You specified more tags than allowed by the TAG parameter.

What to do: Decrease the number of tags.

DES7205N Column *column* **does not exist in table** *schema.table***.**

Explanation: You are trying to enable a text column that does not exist.

What to do: Change the table name or the column name, then try again.

DES7206N Please specify schema owner for table: OWNER.TABLENAME.

Explanation: There is an error with the table name.

What to do: Check the name of the table and try to specify the owner.

DES7207N Index *indexname* **does not exist.**

Explanation: You are trying to specify an index that does not exist.

What to do: Change the index name, then try again.

DES7208N Indexname *indexname* **already exists.**

Explanation: You are trying to specify an index name that already exists.

What to do: Change the index name, then try again.

DES7209N Database is not enabled.

Explanation: You are trying to use a database that was not enabled.

What to do: Enable the database and try again.

DES7210N Database is already enabled.

Explanation: You are trying to enable a database which was already enabled

What to do: Check that the database name is correct.

DES7211N You reached the maximum number of NUMERIC columns.

Explanation: The maximum number of NUMERIC columns has been reached.

What to do: Reduce the number of NUMERIC columns.

DES7212N Installation problem! Check *environmentvariable*

Explanation: You are trying to use Net Search Extender using an incorrect environment variable.

What to do: Check the specified environment variable.

DES7213N No database specified.

Explanation: There is no database information specified.

What to do: Either set the `DB2DBDFT` variable or specify the database in the command.

DES7215N Load of administration tables failed.

Explanation: There seems to be an inconsistency in the administration tables.

What to do: Check if all administration tables are available. See the table layout in "ENABLE DATABASE" on page 59.

DES7216N Directory *directory* **does not exist.**

Explanation: There was a directory specified which does not exist.

What to do: Check the directory name.

DES7218N File could not be created.

Explanation: A file could not be created on the file system.

What to do: Check that there is enough disk space available.

DES7219I Index *index* **is already active.**

Explanation: You are trying to activate an index that is already active.

DES7220I Index is active.

Explanation: You are trying to activate an index that is already active.

DES7221N Index is not active. No search is available.

Explanation: You are trying to use an index that is not active.

DES7230W DB2 Net Search Extender will expire in *nn* **days.**

Explanation: You are using a Try & Buy version of Net Search Extender.

What to do: You can continue to use this version until the expiry date.

DES7300N Index has not been activated.

Explanation: The stored procedure could not access the in-memory table associated with the index name passed to the stored procedure.

What to do: Initialize the in-memory table for the index by issuing the ACTIVATE INDEX command. If the command still fails, UPDATE the index or create it again (DISABLE and ENABLE the column). Verify that the directory

you have specified contains the index files.

DES7301N SQL statement too long; please reduce the result set size.

Explanation: You have just requested a stored procedure to return rows. The stored procedure builds an SQL statement to either pull the rows from DB2 or to return the appropriate values from the in-memory table. However, you have requested so many rows that the SQL statement being built exceeds the maximum length of 32 KB. This limit is reached sooner for the in-memory table than for DB2.

What to do: Reduce the number of requested rows by using the maxRowsToReturn parameter for the stored procedure. You can step through the result set by calling the stored procedure multiple times with increasing value of the startRow parameter.

If the data source is the in-memory table associated with the index, you can also recreate the index with fewer (or smaller) optimization columns.

DES7302N retcode = *code*; *message*

Explanation: The search failed with return code *code* and message *message*.

What to do: Here are some of the possible causes:

- Improper search term syntax
- Illegal characters in the search string
- Null string passed as search term
- Bad index name

Check the search term syntax, index name, and so on.

DES7303N Bad value for parameter *maxRowsToReturn*

Explanation: The value of the parameter passed to the stored procedure was invalid.

What to do: Check the value of the parameter and make sure it is valid. For example, maxRowsToReturn must be a positive integer.

DES7304N Key column must be a unique key.

Explanation: The column specified in the ENABLE TEXT COLUMN command as the key column must be a unique key if the stored procedure is going to be called with dataSource=1, that is, if DB2 is the data source.

What to do: Use a key column that is a unique key if you want to use dataSource=1.

DES7305N Error parsing numeric search specification

Explanation: Syntax of the input string is invalid.

What to do: Check the command syntax.

DES7310N An unexpected token was found at position *position* rc *rc*.

Explanation: The syntax of the input string is invalid.

What to do: Correct the input string and try again.

DES7311N Invalid masking character usage.

Explanation: The characters you have used to specify masking are not valid.

What to do: Change the characters used for the mask token.

DES7400N Internal error occurred in *filename* at line *linenumber*.

Explanation: An internal function has produced an internal program error.

What to do: Make a note of the file name and line number, then contact your IBM representive.

DES7401N Environment variable *variablename* not found.

Explanation: The variable *variablename* was not set in the current environment.

What to do: Check that you are using the correct user ID, and that the user environment has been changed.

DES7402N **Environment variable** *variablename* **value length** *length* **is invalid.**

Explanation: The environment variable value exceeds the maximum size of an internal buffer.

What to do: Change the value of the variable to a smaller value.

DES7403N **Message file path length** *directory and filename* **is invalid.**

Explanation: The path length exceeds the maximum size of an internal buffer.

What to do: Change the value of the path to a smaller value.

DES7404N **Message file** *messagefile* **not found.**

Explanation: The file *messagefile* could not be found.

What to do: Check that the installation is correct and ensure that no files have been removed.

DES7405N **Data type of numeric column** *col-name* **not supported**

Explanation: The data type of the numeric column is not supported.

What to do: Use one of the supported data types.

DES7450N **Unable to convert data of codepage** *codepage***. Reason:** *rc*

Explanation: There is a conversion problem for the data of the specified codepage.

What to do: Take action on the reason code displayed with the message.

DES7451N **Unable to convert data to UTF-8. Reason:** *rc*

Explanation: There is a problem to convert data to UTF-8.

What to do: Take action on the reason code displayed with the message.

DES7452N **Data conversion to unicode failed. Reason:** *rc*

Explanation: There is a problem to convert data to unicode.

What to do: Take action on the reason code displayed with the message.

DES7453N **Unable to open converter** *convertername***. Reason:** *rc*

Explanation: There is a problem to open the specified converter.

What to do: Take action on the reason code displayed with the message.

DES7454N **Unable to open converter for CCSID** *ccsid***. Reason:** *rc*

Explanation: There is a problem to open a converter related to the specified CCSID.

What to do: Take action on the reason code displayed with the message.

DES7455N **Unable to convert data from unicode to** *format***. Reason:** *rc*

Explanation: There is a problem to convert data from unicode to the specified data format.

What to do: Take action on the reason code displayed with the message.

DES7456N **Unable to convert** *format* **query string to UTF-8. Reason:** *rc*

Explanation: Unable to convert the query string data of the specified format to UTF-8 format.

What to do: Take action on the reason code

displayed with the message.

DES7520N In-Memory Table status not available.

Explanation: Could not read information from the shared memory.

What to do: Deactivate the index and activate it again.

DES7521N Request failed to delete In-Memory Table *table*

Explanation: Shared memory could not be deleted.

What to do: Check the current status using `ipcs -s` and do a cleanup manually.

DES7523N The requested In-Memory Table would require *nnn* **bytes which exceeds system limits.**

Explanation: The shared memory could not be allocated.

What to do: Check the system resources and limits.

DES7525N In-Memory Table *table* **could not be attached.**

Explanation: The shared memory could not be attached.

What to do: For Unix systems, check the SharedMemory ID using `ipcs -s` and try to clean up.

DES7526N MapViewOfFile failed with return code *rc***.**

Explanation: The shared memory could not be accessed.

What to do: Check the return code to isolate the problem.

DES7527N In-Memory Table SHM file could not be accessed.

Explanation: The SHM file of the index could not be accessed.

What to do: Check if the SHM file in the index directory exists and could be read.

DES7600I Index *index* **is not enabled for Incremental Update.**

Explanation: The index was not enabled using the keyword INCREMENTAL. No update is possible.

What to do: If the index should be updated, it must be recreated using the INCREMENTAL keyword.

DES7601I No Update needed.

Explanation: There have been changes since the last index update, but they do not need to be added to the index. For example, a record was added, but has subsequently been deleted since the last update run.

DES7602I No Update record found in Log-Table.

Explanation: There have been no changes since the last index update.

DES7605N No Unique Key for Source Table given.

Explanation: The column used as key column for the index is not unique. The index cannot be enabled for incremental update.

What to do: Use a unique column as the key column.

DES7606N Name of Shared Memory Directory not found.

Explanation: The name of the shared memory directory could not be found in the database. There must be a problem with the Net Search Extender administration tables.

What to do: Try to disable the index and then run the enable index again.

DES7607N Name of Temp-Shared Memory Directory not found.

Explanation: The name of the temp-shared memory directory could not be found in the database. There must be a problem with the Net Search Extender administration tables.

What to do: Try to disable the index and then enable the index again.

DES7608N No Space left in Shared Memory.

Explanation: The shared memory resources have been exhausted. There is no space left.

What to do: Try a REORG. Alternatively, delete some data.

DES7610N DATA-Shared Memory Segment not found.

Explanation: A shared memory segment is missing, the index has been corrupted.

What to do: The index must be recreated.

DES7613N Can't allocate more Memory.

Explanation: The program requires more RAM than is currently available.

What to do: Free some memory by terminating unnecessary processes. Add swop space.

DES7614N Error when accessing TextSearch-Engine.

Explanation: There is a problem accessing the underlying search engine.

What to do: Check the installation directory against the installation file inventory.

DES7615N Error when accessing Database. \nSQLCODE: *sqlcode***nMessage:** *message***nSQLSTATE:***sqlstate*

Explanation: The database returned the described SQLCODE and message.

What to do: Refer to the *DB2 User Guide* to solve the problem.

DES7616N Can't open Shared Memory segment.

Explanation: The shared memory cannot be opened with write access.

What to do: Check the rights of the current user. Retry the operation with a user who has write access to the shared memory.

DES7800I The command completed successfully.

Explanation: The specified command completed successfully.

DES7801N The command failed.

Explanation: The command was not executed sucessfully.

What to do: Check the error message to get more information about the problem.

DES7802N The DB2 Net Search Extender license is invalid or has been expired.

ES7803N A DB2 Net Search Extender Try & Buy license found.

Explanation: You are currently working with a Try & Buy license.

What to do: If you intend to use Net Search Extender after the expiry date, upgrade to the normal version.

DES7804I **Starting reorganization of index** *index*

Explanation: The reorganization of the index has started.

DES7805I **Reorganization of index** *index* **has been ended successfully.**

Explanation: The index has been reorganized.

DES7806I **Reorganization of index** *index* **has ended with errors**

Explanation: The index has not been reorganized because an error has occurred.

What to do: Check the output for the errors. Fix the error and retry.

DES7807I **Starting Incremental Update of Index** *index*

Explanation: Starting incremental update for index *index*.

DES7808I **Incremental Update of Index** *index* **raised SHM-full-condition - starting implicit reorg.**

Explanation: The shared memory ran out of space while updates where being applied. The shared memory will be reorganized and the update will continue after the reorganization has completed.

DES7809I **Incremental Update of Index** *index* **has ended successfully**

Explanation: The index has been updated.

DES7810I **Incremental Update of Index** *index* **has ended with errors.**

Explanation: An error occured while the index was being updated.

What to do: Check the output for the error. Fix the error and retry.

DES7999N **Internal error: Invalid functionid.**

Explanation: The command parser found a function that is not valid.

What to do: Check the command syntax and try again.

DES9997N **An SQL error occurred. SqlState:** *state* **SQL Error code:** *code* **SqlErrorMessage:** *message*

Explanation: An SQL error occurred.

What to do: Take action on the SQL error message that is displayed with the message.

DES9998N **An SQL error occurred. No further information is available.**

Explanation: An internal processing error occurred.

What to do: Collect the available information and report it to your IBM service representative.

Part 3. Appendixes

Notices

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

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not 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.

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

IBM Corporation
J74/G4
555 Bailey Avenue
P.O. Box 49023
San Jose, CA 95161-9023
U.S.A.

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

The licensed program described in this 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.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

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

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which 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:

(c) (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. (c) Copyright IBM Corp. _enter the year or years_. All rights reserved.

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

Trademarks

The following terms are trademarks of the IBM Corporation in the United States, other countries, or both:

AIX	DB2 Universal Database	Net.Data
DB2	IBM	

Java and all Java-based trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

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

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

Intel is a registered trademark of Intel.

Linux is a registered trademark of Linus Torvalds.

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

Index

J

M

N

O

P

R

S

T

U

V

W

Readers' Comments — We'd Like to Hear from You

DB2 Universal Database
Net Search Extender Administration and Programming Guide
Version 7

Publication No. SC27-0818-04

Overall, how satisfied are you with the information in this book?

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Overall satisfaction	☐	☐	☐	☐	☐

How satisfied are you that the information in this book is:

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
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. _____

Readers' Comments — We'd Like to Hear from You
SC27-0818-04

Cut or Fold
Along Line

Fold and Tape **Please do not staple** Fold and Tape

PLACE
POSTAGE
STAMP
HERE

IBM Deutschland Entwicklung GmbH
Information Development, Dept. 0446
Schoenaicher Strasse 220
71032 Boeblingen
Germany

Fold and Tape **Please do not staple** Fold and Tape

SC27-0818-04

Cut or Fold
Along Line

Program Number: 5648-D66

SC27-0818-04

Spine information:

DB2 Universal Database

Net Search Extender Administration and Programming

Version 7