

IBM DB2 10.1  
for Linux, UNIX, and Windows

*Text Search Guide*





IBM DB2 10.1  
for Linux, UNIX, and Windows

*Text Search Guide*



**Note**

Before using this information and the product it supports, read the general information under Appendix E, "Notices," on page 245.

**Edition Notice**

This document contains proprietary information of IBM. It is provided under a license agreement and is protected by copyright law. The information contained in this publication does not include any product warranties, and any statements provided in this manual should not be interpreted as such.

You can order IBM publications online or through your local IBM representative.

- To order publications online, go to the IBM Publications Center at <http://www.ibm.com/shop/publications/order>
- To find your local IBM representative, go to the IBM Directory of Worldwide Contacts at <http://www.ibm.com/planetwide/>

To order DB2 publications from DB2 Marketing and Sales in the United States or Canada, call 1-800-IBM-4YOU (426-4968).

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.

© Copyright IBM Corporation 2008, 2012.

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

---

# Contents

## Chapter 1. DB2 Text Search . . . . . 1

### Chapter 2. DB2 Text Search key features and concepts . . . . . 3

DB2 Text Search server deployment scenarios . . . . .	4
Text search index creation, updates and property alterations . . . . .	6
DB2 Text Search in a partitioned database environment . . . . .	9
Incremental updates for DB2 Text Search indexes . . . . .	11
Linguistic processing for DB2 Text Search . . . . .	12
Scenario: Indexing and searching . . . . .	14
Rich text and proprietary format support . . . . .	16

### Chapter 3. Text search solution planning . . . . . 19

Document characteristics . . . . .	19
Document formats supported for DB2 Text Search . . . . .	19
Supported data types . . . . .	19
Conversion of unsupported formats and data types . . . . .	19
Supported languages and code pages . . . . .	20
Document size considerations . . . . .	20
DB2 Text Search security overview . . . . .	21
User roles . . . . .	22
Access policies and communication security . . . . .	24
DB2 Text Search capacity planning and optimization . . . . .	24
DB2 Text Search server configuration . . . . .	25
DB2 Text Search index planning and optimization . . . . .	29
DB2 Text Search system tuning . . . . .	33
DB2 Text Search query planning . . . . .	35
DB2 Text Search arguments . . . . .	35
DB2 Text Search multiple predicates . . . . .	35
DB2 Text Search locale and language . . . . .	36
DB2 Text Search SCORE function . . . . .	37
DB2 Text Search RESULTLIMIT function . . . . .	37
Parser configuration for DB2 Text Search . . . . .	37
DB2 Text Search XML namespaces . . . . .	38

### Chapter 4. Installing and configuring DB2 Text Search . . . . . 41

Hardware and software requirements for DB2 Text Search . . . . .	43
Installing DB2 Text Search with a default configuration . . . . .	44
Installing and configuring DB2 Text Search with the DB2 Setup Wizard . . . . .	44
Installing and configuring DB2 Text Search with a response file . . . . .	45
Installing DB2 Text Search using db2_install (Linux and UNIX) . . . . .	46
Installing DB2 Text Search without initial configuration . . . . .	46

Installing DB2 database servers using the DB2 Setup wizard (Windows) . . . . .	46
Installing DB2 servers using the DB2 Setup wizard (Linux and UNIX) . . . . .	49
Response file installation of DB2 overview (Windows). . . . .	52
Response file installation of DB2 overview (Linux and UNIX) . . . . .	53
Installing and configuring a stand-alone Text search server . . . . .	54
Installation space requirements for the stand-alone server . . . . .	54
Installing a stand-alone DB2 Text Search server . . . . .	54
Installing and configuring stand-alone server as a Windows service . . . . .	55
Uninstalling a stand-alone DB2 Text Search server . . . . .	55

### Chapter 5. Configuring DB2 Text Search . . . . . 57

Initial configuration of an integrated DB2 Text Search server . . . . .	59
Updating DB2 Text Search server information . . . . .	60
Configuring a stand-alone DB2 Text Search server . . . . .	61
Updating the services file on the server for TCP/IP communications . . . . .	63
Installing DB2 Accessories Suite for DB2 Text Search . . . . .	63
Uninstalling the DB2 Accessories Suite for DB2 Text Search . . . . .	64

### Chapter 6. Upgrading DB2 Text Search 67

Upgrading DB2 Text Search for administrator or root install. . . . .	67
Upgrading DB2 Text Search for non-root install (Linux and UNIX) . . . . .	70
Upgrading a multi-partition instance without DB2 Text Search . . . . .	71
Upgrading a stand-alone DB2 Text Search Server . . . . .	72

### Chapter 7. Configuring and administering text search indexes . . . 73

Command-line tools for DB2 Text Search . . . . .	73
Issuing text search commands . . . . .	73
Rich text and proprietary format support . . . . .	74
Enabling DB2 Text Search for rich text document support. . . . .	74
Disabling support for rich text and proprietary formats . . . . .	74
Starting the DB2 Text Search instance service . . . . .	75
Stopping the DB2 Text Search instance service . . . . .	75
Enabling a database for DB2 Text Search . . . . .	76
Disabling a database for DB2 Text Search . . . . .	77
Deleting orphaned DB2 Text Search collections . . . . .	78
Synonym dictionaries for DB2 Text Search . . . . .	79

Adding a synonym dictionary for DB2 Text Search . . . . .	80
Removing a synonym dictionary for DB2 Text Search . . . . .	81
Text search index creation . . . . .	81
Creating a text search index . . . . .	82
Text search index maintenance . . . . .	88
Updating a text search index . . . . .	90
Clearing text search index events . . . . .	93
Altering a text search index . . . . .	94
Viewing text search index status . . . . .	95
Changing the location of a DB2 Text Search collection . . . . .	95
Backing up and restoring text search indexes . . . . .	96
Dropping a text search index . . . . .	96
Sample: Scheduling a DB2 Text Search index update . . . . .	98

**Chapter 8. Searching with text search indexes . . . . . 99**

Search functions for DB2 Text Search . . . . .	99
Full-text search methods. . . . .	100
Basic search . . . . .	101
Fuzzy search . . . . .	101
Proximity search . . . . .	102
Searching for special characters . . . . .	103
Structural full-text search in XML documents . . . . .	106
Searching text search indexes using SCORE . . . . .	107
DB2 Text Search argument syntax . . . . .	108
Search syntax for XML documents . . . . .	112
Enhancing performance for full-text queries . . . . .	116

**Chapter 9. SQL and XML built-in search functions . . . . . 119**

CONTAINS function . . . . .	119
SCORE function . . . . .	121
xmlcolumn-contains function . . . . .	124

**Chapter 10. Administration commands for DB2 Text Search . . . . . 129**

db2ts ALTER INDEX . . . . .	130
db2ts CLEANUP FOR TEXT . . . . .	134
db2ts CLEAR COMMAND LOCKS . . . . .	135
db2ts CLEAR EVENTS FOR TEXT . . . . .	136
db2ts CREATE INDEX . . . . .	138
db2ts DISABLE DATABASE FOR TEXT . . . . .	145
db2ts DROP INDEX . . . . .	147
db2ts ENABLE DATABASE FOR TEXT . . . . .	148
db2ts HELP . . . . .	150
db2ts RESET PENDING command . . . . .	152
db2ts START FOR TEXT . . . . .	153
db2ts STOP FOR TEXT . . . . .	154
db2ts UPDATE INDEX . . . . .	155

**Chapter 11. DB2 Text Search stored procedures . . . . . 159**

SYSTS_ADMIN_CMD procedure - Run text search administration commands . . . . .	159
---	-----

SYSTS_ALTER procedure - Change the update characteristics of an index . . . . .	161
SYSTS_CLEANUP procedure - Remove invalid text search indexes . . . . .	166
SYSTS_CLEAR_COMMANDLOCKS procedure - Remove command locks for text search indexes . . . . .	167
SYSTS_CLEAR_EVENTS procedure - Delete indexing events from an index's event table . . . . .	169
SYSTS_CONFIGURE procedure - Configure current database for text search . . . . .	171
SYSTS_CREATE procedure - Create a text search index on a column . . . . .	174
SYSTS_DISABLE procedure - Disable current database for text search . . . . .	182
SYSTS_DROP procedure - Drop text search index . . . . .	185
SYSTS_ENABLE procedure - Enable current database for text search . . . . .	187
SYSTS_UPDATE procedure - Update the text search index. . . . .	189
SYSTS_UPGRADE_CATALOG procedure - Upgrade the text search catalog . . . . .	192
SYSTS_UPGRADE_INDEX - Upgrade text search indexes . . . . .	193

**Chapter 12. Text search administrative views . . . . . 197**

SYSIBMTS.TSDEFAULTS view. . . . .	197
SYSIBMTS.TSLOCKS view . . . . .	198
SYSIBMTS.TSSERVERS view . . . . .	199
SYSIBMTS.TSINDEXES view . . . . .	199
SYSIBMTS.TSCONFIGURATION view . . . . .	201
SYSIBMTS.TSCOLLECTIONNAMES view. . . . .	201
SYSIBMTS.TSEVENT view . . . . .	202
SYSIBMTS.TSSTAGING view . . . . .	203

**Appendix A. DB2 Text Search and Net Search Extender comparison . . . . . 205**

**Appendix B. Locales supported for DB2 Text Search . . . . . 211**

**Appendix C. DB2 commands. . . . . 213**

db2iupgrade - Upgrade instance . . . . .	213
db2icrt - Create instance. . . . .	216
db2idrop - Remove instance . . . . .	224
db2iupdt - Update instances . . . . .	226

**Appendix D. Overview of the DB2 technical information . . . . . 235**

DB2 technical library in hardcopy or PDF format . . . . .	235
Displaying SQL state help from the command line processor. . . . .	238
Accessing different versions of the DB2 Information Center . . . . .	238
Updating the DB2 Information Center installed on your computer or intranet server. . . . .	238
Manually updating the DB2 Information Center installed on your computer or intranet server . . . . .	240
DB2 tutorials . . . . .	241

DB2 troubleshooting information . . . . . 242  
Terms and conditions. . . . . 242

**Index . . . . . 249**

**Appendix E. Notices . . . . . 245**





---

## Chapter 1. DB2 Text Search

DB2® Text Search allows you to search text columns by issuing SQL and XQuery statements to perform text search queries on data stored in a DB2 database.

DB2 Text Search provides extensive capabilities for searching data in text columns stored in a DB2 table. The search system provides fast query response times and a consolidated, ranked result set that enables you to quickly and easily locate the information that you need. By incorporating the functions of DB2 Text Search in your SQL and XQuery statements, you can create powerful and versatile text-retrieval programs. Furthermore, the search engine uses linguistic analysis to ensure that it returns only relevant search query results. By enabling text search support, you can use the CONTAINS, SCORE, and xmlcolumn-contains functions, which are built into the DB2 engine, to search text search indexes based on the search arguments that you specify.

DB2 Text Search achieves high performance and scalability by employing data streaming to avoid high resource consumption during search.

You can install the DB2 Text Search server and DB2 database servers on the same system for an integrated text search server setup. You can also install DB2 Text Search server and the DB2 database server on different systems for a stand-alone setup. The DB2 Text Search server runs in its own Java Virtual Machine (JVM). You explicitly start and stop the DB2 Text Search services after starting the DB2 instance.

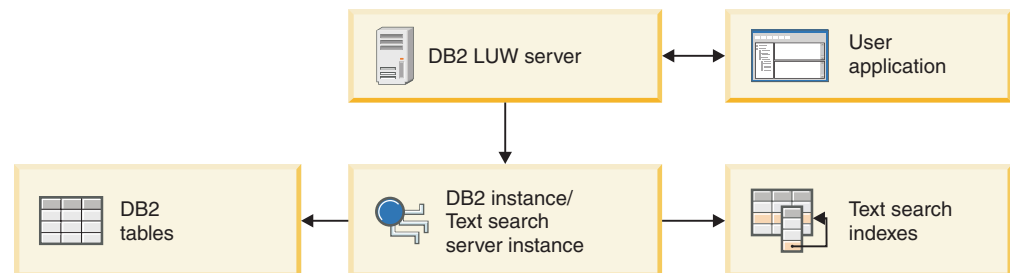


Figure 1. Deployment diagram for an integrated DB2 Text Search server

DB2 Text Search does not have a graphical user interface. Instead, command-line tools are available for tasks such as configuring and administering the DB2 Text Search server, creating a synonym dictionary for a collection, and diagnosing problems. In addition, you can use a stored-procedure interface for various common administrative tasks.

You can migrate from Net Search Extender to DB2 Text Search by creating and updating DB2 Text Search indexes and then toggling the index status when the indexes are ready for use. For details, see the topic about migration from Net Search Extender to DB2 Text Search.

DB2 Text Search includes the following key features:

### Tight integration with DB2 Database for Linux, UNIX, and Windows

- A stored procedure interface for administration commands
- Installation and configuration performed by the DB2 installer

- Invisible authentication
- SQL codes for error handling

#### **Document indexing**

- Fast indexing of large amounts of data
- pureXML<sup>®</sup> support
- Multiple document format support
- Incremental and asynchronous index updating

#### **Advanced search technology**

- SQL, SQL/XML, and XQuery support
- The CONTAINS and SCORE SQL functions
- Built-in SQL functionality combined with the DB2 Optimizer
- The xmlcolumn-contains XML function
- XML filtering
- Linguistic processing in all supported languages
- Weight, wildcard, and optional term support
- Synonym dictionary support

## Chapter 2. DB2 Text Search key features and concepts

DB2 Text Search offers you a fast and versatile method for searching text documents that are stored in a table column in DB2 databases. You can search the documents by using SQL queries or XQuery for searches on XML documents.

The text documents must be uniquely identifiable. DB2 Text Search uses the primary key of the table for this purpose.

Rather than searching text documents sequentially, DB2 Text Search searches using a *text search index*, which is a more efficient approach. A text search index consists of significant terms that are extracted from the text documents.

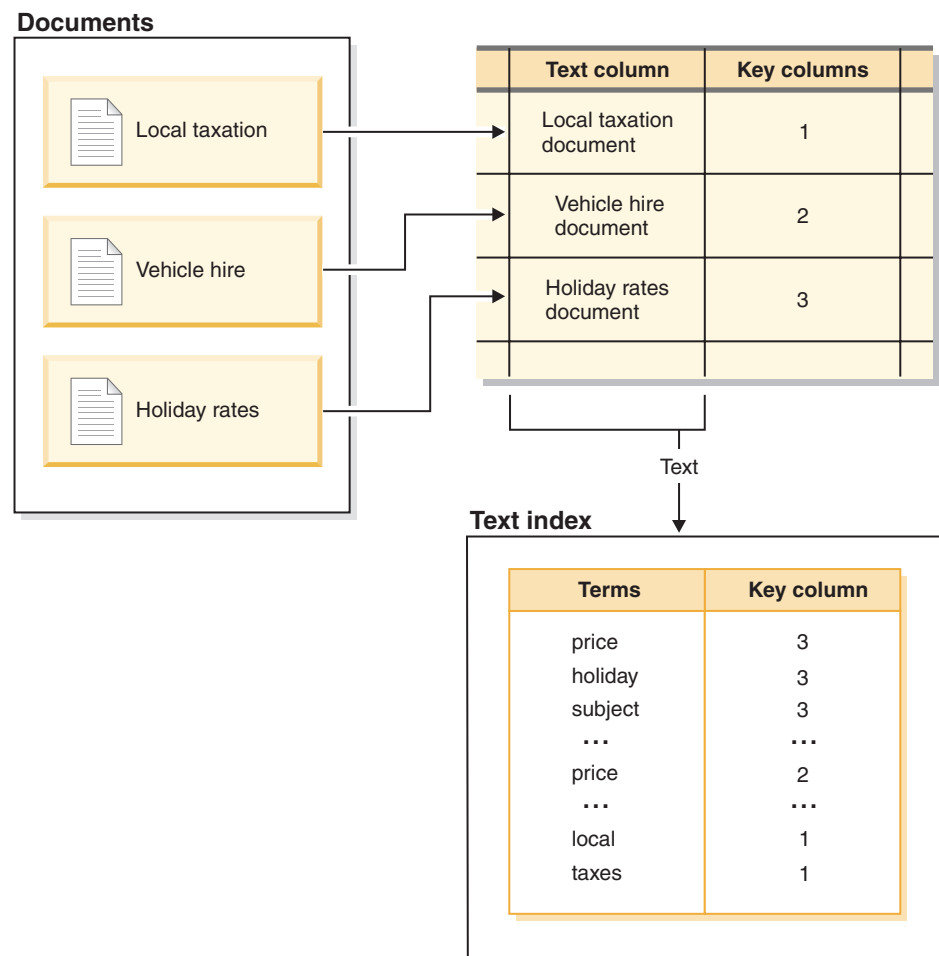


Figure 2. Creating a text search index

Creating a text search index defines the properties of the index, such as the update frequency. The text search index does not contain any data immediately after you create it. Updating the index adds data about the terms and the text documents to the text search index. The initial index update adds all text documents from a text column to the index. Subsequent updates are known as incremental updates and synchronize the data in the table and the data in the text search index. DB2 Text Search provides two methods for synchronizing a text search index with its table:

- The basic synchronization method uses triggers that automatically store information about new, changed, and deleted documents in a staging table.
- The extended synchronization method uses a trigger to store information about changed documents in a staging table but captures information about new and deleted documents through integrity processing and stores that information in an auxiliary staging table.

See the text search index creation, updates, and property alterations topic for details.

DB2 Text Search works by collecting data from diverse sources and indexing it for subsequent fast retrieval. DB2 Text Search uses linguistic analysis to improve search results and supports the following document formats:

- Unstructured plain text.
- Structured text such as that in HTML or XML documents
- Proprietary document formats such as PDF or Microsoft Office document formats.

For proprietary formats, you need filtering software that might require an additional download and setup step.

DB2 Text Search supports full-text search in a partitioned database environment. You can also create a text search index for range-partitioned tables or tables that use the multidimensional clustering feature in a single-partition or partitioned database environment. Text search indexes are supported for any partitioning feature combination. In a partitioned database environment, the text search index is partitioned according to the partitioning of the table across multiple database partitions. Other partitioning features, such as table partitioning or multidimensional clustering, do not affect the partitioning of the text search index.

DB2 Text Search also supports both an integrated or a stand-alone setup. A stand-alone DB2 Text Search server is preferred for partitioned environments, as it avoids resource contention with the database server. DB2 Text Search is not supported in a DB2 pureScale® environment.

---

## DB2 Text Search server deployment scenarios

DB2 Text Search supports an integrated installation of the text search server as well as a stand-alone one separate from the DB2 database product. A stand-alone text search server, also known as Enterprise Content Management (ECM) Text Search server, can be installed and administered on supported host platforms. DB2 Text Search is not supported with the High availability disaster recovery (HADR) feature.

The DB2 database instance uses TCP/IP to communicate with the stand-alone DB2 Text Search server. SSL or GSKit support are not available. However, encryption channels can be used through the **stunnel** program or SSH tunneling. Restrict access to your document repository and text search index files depending on your security requirements. The stand-alone text search server must be installed on computers with a secure network connection behind a firewall to prevent unauthorized access to the text search indexes. Setting up TCP/IP access restriction to the stand-alone text search server ensures that it can only be accessed by the host on which the database server is installed.

The following are high-level illustrations of DB2 Text Search server deployments, including integrated and stand-alone setups. You can set up and configure an

integrated DB2 Text Search server and switch to a stand-alone server later. However, there is no automated support to move text search indexes to a different text search server. Depending on the setup it might therefore be necessary to drop existing text indexes before assigning a new text search server to the database instance.

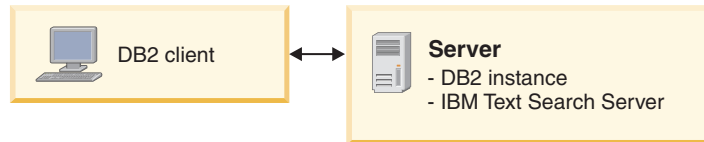


Figure 3. Integrated DB2 Text Search server setup

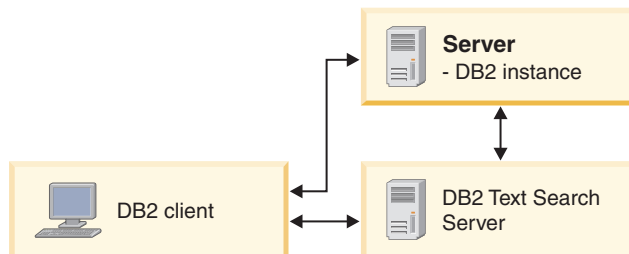


Figure 4. Stand-alone DB2 Text Search server setup

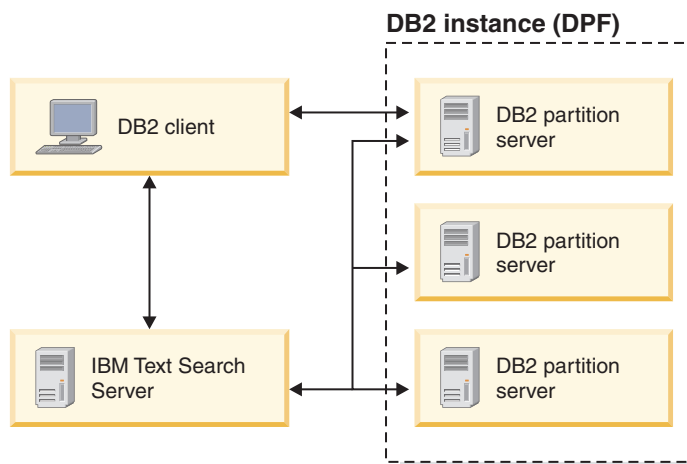


Figure 5. Stand-alone DB2 Text Search server setup in a partitioned environment

**Note:** The DB2 Text Search installation directory depends on the type of deployment.

- For an integrated server:
  - `<TS_HOME>` represents the `../sql1lib/db2tss` path on Windows, Linux or UNIX operating systems.
- For a stand-alone setup, `<ECMTS_HOME>` represents the install location of the text search server.
  - By default, `<ECMTS_HOME>` represents the `/opt/ibm/ECMTextSearch` path on Linux or UNIX systems.
  - By default, `<ECMTS_HOME>` represents the `C:\Program Files\IBM\ECMTextSearch` path on Windows systems.

Deployment of a stand-alone text search server should be considered for:

- security management: the stand-alone Text Search server allows to define a text server process owner other than the database instance owner.
- workload management: the stand-alone Text Search server separates the resource-intensive text search processing from database server tasks.

Each database instance is associated with a single Text Search server. In partitioned database environments involving multiple partition servers, a stand-alone setup avoids a concentration of resource-intensive processing on a single partition server.

The stand-alone and the integrated Text Search server only differ in the initial configuration, most notably, the stand-alone Text Search server is already configured for processing of rich text/proprietary format documents.

---

## Text search index creation, updates and property alterations

Text search index creation is the process of defining the properties of a text index. After you create a text search index, you must update it by adding data from the table that it is associated with. You can also alter some properties of the text search index later, such as the **UPDATE FREQUENCY** or **UPDATE MINIMUM** parameters.

You can use a text search index to search through the data in a text column using text search functions. A text search index consists of significant terms that are extracted from text documents. The primary key of the table row is used in the index to identify the source of the terms.

Immediately after its creation, a text search index contains no data. You add data to a text search index by using the **db2ts UPDATE INDEX** command or the **SYSTS\_UPDATE** administrative SQL routine. The first index update, also known as *initial update*, adds all text documents in a text column to the text search index. Subsequent updates, also known as *incremental updates*, synchronize the data in the base table with the text search index.

In the following example, a user creates a text search index called **MYSHEMA.PRODUCTINDEX** on the **PRODUCT** table in the **SAMPLE** database. Creating a text search index and then performing initial and incremental updates shows that the index is empty until the user performs an initial update and that as the user adds data to the table, an incremental update must be run to add the new data to the text search index.

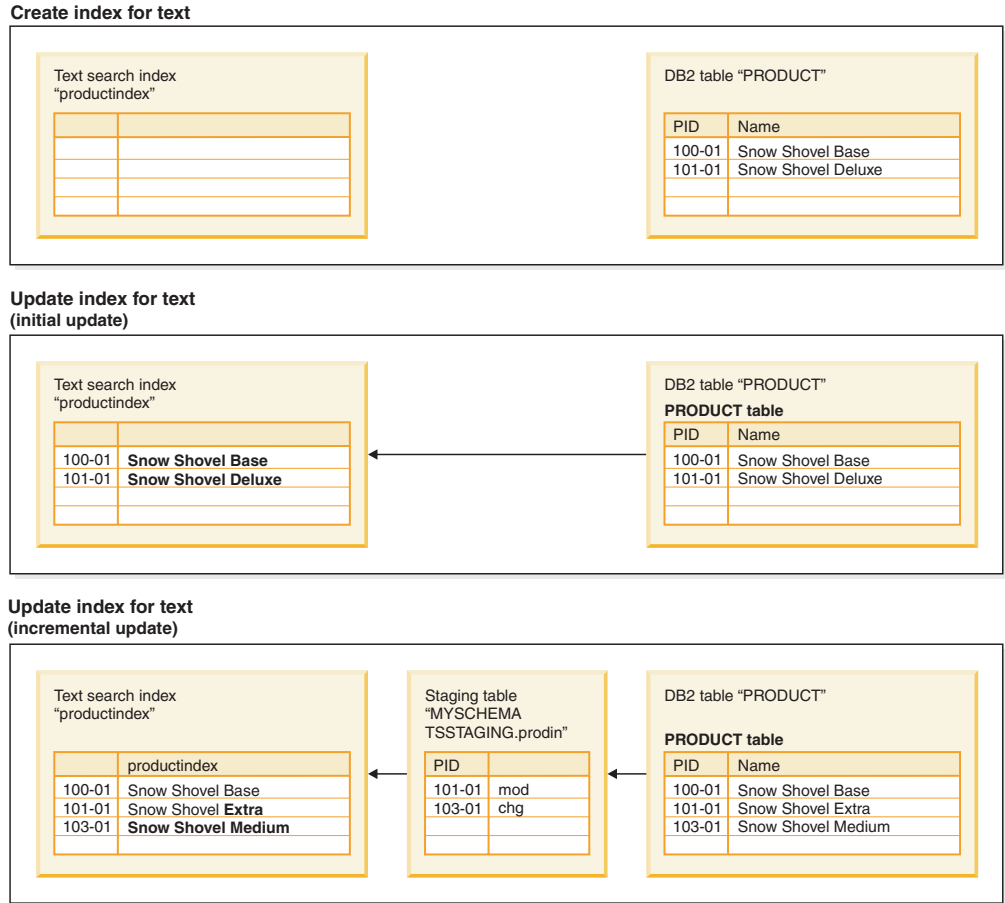


Figure 6. Creating a text search index and then performing initial and incremental updates

DB2 Text Search provides two methods for synchronizing a text index with its table:

- The basic synchronization method uses triggers that automatically store information about new, changed, and deleted documents in a staging table. There is one staging table for each text index.

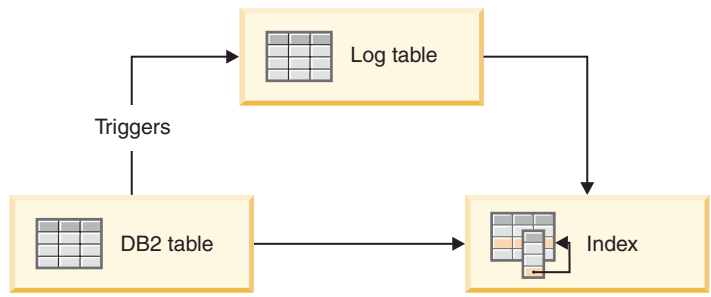


Figure 7. Incremental update with triggers

Because the basic method uses only triggers, updates that are not recognized by triggers are ignored, for example, loading data with the **LOAD** command and attaching or detaching the ranges of a range-partitioned table.

- The extended synchronization method uses a trigger to store information about changed documents in a staging table but captures information about new and deleted documents through integrity processing and stores that information in a

text-maintained auxiliary staging table. If you attach a partition or load data, you must then issue the **SET INTEGRITY** command on the base table to make data available in the auxiliary staging table. As for the case when a partition is detached, the staging table then requires another **SET INTEGRITY** command to make the data accessible for processing. Alternatively, a **RESET PENDING** command on the base table can be used to make the data accessible in all its auxiliary staging tables. The base table is fully accessible for read and write operations while the command is executing. If you detach a partition, you must issue the **RESET PENDING** command on the base table or the **SET INTEGRITY** command on each of the staging tables.

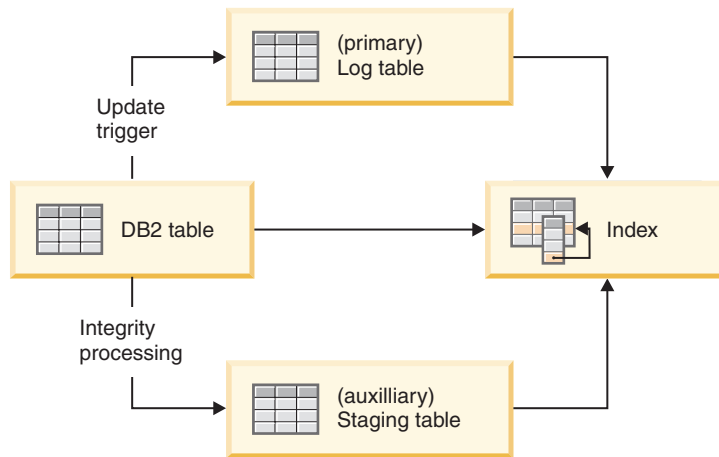


Figure 8. Incremental update with triggers and integrity processing

Some database operations implicitly or explicitly invalidate the text search index. An explicit invalidation will set the status of the text search index `INDSTATUS='INVALID'` in the `SYSIBMTS.TSINDEXES` administrative view, for example, the command `ALTER DATABASE PARTITION GROUP`. An implicit invalidation occurs when content changes bypass the staging mechanism, for example, if a `LOAD INSERT` is used without the extended staging infrastructure. An implicit invalidation will not mark the text search index as invalid.

You can update the text index by using a manual or automatic option. The automatic option uses an update schedule with specified days and times. You can manually update the text search index by issuing the **UPDATE INDEX FOR TEXT** command or the `SYSPROC.SYSTS_UPDATE` procedure. The text search index is updated asynchronously, that is, outside the transaction that inserts, updates, or deletes data in the database. Asynchronous text search index update processing improves throughput and concurrency because multiple updates can be batched and applied to a copy of the affected text index segments. The text search index is then only locked for read access for a short period of time while the updated index segments are put in place of the original.

Text search indexes are reorganized automatically as needed; in addition, you can explicitly trigger a reorganization with the adminTool or re-create an index with the `ALLROWS` option when you update it.



---

## DB2 Text Search in a partitioned database environment

DB2 Text Search supports full-text search in a partitioned database environment. Text search indexes are distributed in a pattern that matches the base tables on which they are created. For each database partition, a text index partition, also called a collection, is created. This pattern facilitates text search maintenance by allowing text search index updates with parallel execution on all index partitions.

The staging tables used for multi-collection text search index updates are per index rather than per collection and are distributed in a manner similar to the base table. Staging tables use the `DBPARTITIONNUM` scalar function to find relevant changes that need to be applied to each index partition per index refresh. Data from each database partition server is updated in the corresponding text index partition during the text search index update to enable a parallelization of the update operation.

Every text search index update may result in multiple collection updates and text search server capacity planning is required. For workload distribution, a stand-alone remote text search server setup is recommended in partitioned database environments.

A DB2 Text Search server setup that is installed and configured separately from the DB2 instance is referred to as a stand-alone setup. A remote stand-alone setup, that is, a setup on a separate host from the database server, can be used for non-partitioned, single-partition and multi-partitioned DB2 instances to remove the resource-intensive text search server workload from the database server host.

The configuration of the integrated Text Search server during the default instance creation of a partitioned database instance applies to the lowest numbered database partition server. It is not required to configure during installation, the administration and configuration of the Text Search server in an existing partitioned database environment can be managed by Text Search server tools.

The following diagram depicts a DB2 instance with four database partitions. They are located on two dedicated hosts, Machine1 and Machine2 with two logical partitions per host. All database partition servers are served by a single text search server.

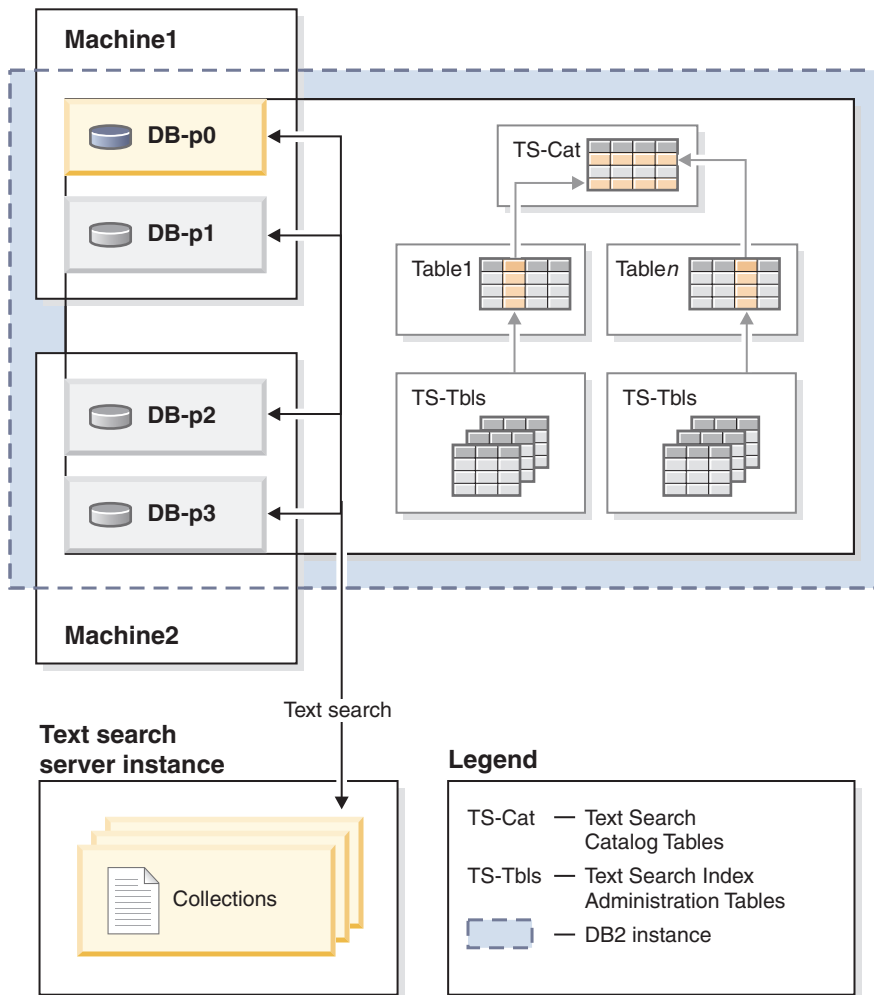


Figure 9. A DB2 Text Search server setup in a partitioned environment

Stand-alone setups are suggested to help achieve a balanced workload and avoid sharing resources by the text search server with a single database partition server.

In a partitioned database environment, the **db2ts START FOR TEXT** command with the **STATUS** and **VERIFY** parameters can be issued on any one of the partition server hosts. To start the instance services, you must run the **db2ts START FOR TEXT** command on the integrated text search server host machine. The integrated text search server host machine is the host of the lowest-numbered database partition server. If custom collection directories are used, ensure that no lower numbered partitions are created later. This restriction is especially relevant for Linux and UNIX platforms. If you configure DB2 Text Search when creating an instance, the configuration initially determines the integrated text search server host. That configuration must always remain the host of the lowest-numbered database partition server.

Database partitions in a partitioned instance can be added and deleted. This is generally followed by data redistribution, using the **REDISTRIBUTE DATABASE PARTITION GROUP** command to move and rebalance data in the tables. If a text search index is hosted by one of the affected tables, such a data redistribution requires a reshuffling of the text index partition content to align the text index partitions with the new set of relevant database partitions. Incremental updates of text search indexes are usually inadequate for this purpose, instead, the text search

index must be updated with the FOR DATA REDISTRIBUTION option. Note that this can result in significant downtimes for large workloads similar to initial updates.

When enabling and administering DB2 Text Search in a partitioned database environment, consider the following:

- Ensure that the DB2 setup is complete as described in the DB2 documentation. The NFS mount must be configured with root access and setuid.
- If startup fails, you need to check if DB2 Text Search has been configured correctly and then issue the **db2ts START** command a second time.
- Before inserting or deleting partition numbers from the `db2nodes.cfg` file, stop the DB2 Text Search instance services. This applies to any command that might result in changes to the `db2nodes.cfg` configuration file.
- On Windows platforms, while using DB2 Text Search in a partitioned database environment, the `db2nodes.cfg` file should not use IP addresses as well as host names for the same host.

You should be aware of the following considerations when conducting searches in a partitioned database environment:

- The RESULT LIMIT is evaluated on every partition during search. This means that if you specify a RESULT LIMIT of 3 and use 4 partitions, you will get up to 12 results.
- The SCORE value reflects the document's relevance when compared to the SCORE value of all documents from a single partition even if the query accesses multiple partitions.

---

## Incremental updates for DB2 Text Search indexes

Data synchronization in DB2 Text Search is based on triggers that update a staging table whenever the triggers capture information about new, changed, or deleted documents. There is one staging table for each text index. Applying the information in the staging table to the corresponding text index is referred to as performing an *incremental update*.

The AUXLOG option of the **INDEX CONFIGURATION** parameter of the DB2 Text Search **CREATE INDEX** operation controls whether a text-maintained staging table is used for a text search index. If you set the AUXLOG option to ON, information about new and deleted documents is captured through integrity processing in an auxiliary staging table maintained by DB2 Text Search, and information about changed documents is captured through triggers and stored in the staging table. By default, this configuration option is set to ON for range-partitioned tables and OFF for nonpartitioned tables.

Capturing changes for an incremental update of the text index through integrity processing might require you to perform additional administrative tasks. For example, you may have to issue a **RESET PENDING** command before text search index updates can be processed. The effect of the text-maintained staging infrastructure is similar to the effect of a materialized query table (MQT) with deferred refresh.

If you update tables using only commands that affect all rows in the tables, for example, using the **LOAD REPLACE** command, adding the extended staging infrastructure does not provide a benefit. Instead, it is recommended to re-create the index after updating a table.

To create a text index on the table and load data through initial and incremental updates see the following example.

1. Create a table and add data to it.

```
db2 "create table test.simple (pk integer not null primary key,
comment varchar(48))"
db2 "insert into test.simple values (1, 'blue and red')"
```

2. Create a text search index.

```
db2ts "create index test.simpleix for text on test.simple(comment)
index configuration(auxlog on) connect to mydb"
```

3. Update the index and load data.

```
db2ts "update index test.simpleix for text connect to mydb"
db2 "load from loaddata4.sql of del insert into test.simple"
```

4. After the load operation, the base table is locked. For example, a select operation results in SQL0668N Operation not allowed for reason code "1" on table "TEST.SIMPLE". SQLSTATE=57016. The staging table is accessible, but it does not yet contain the information about the new data.

5. Enable integrity processing.

```
db2 "set integrity for test.simple immediate checked"
```

The following message is returned:

```
SQL3601W The statement caused one or more tables to automatically
be placed in the Set Integrity Pending state.SQLSTATE=01586
```

6. At this point, the staging table is locked, and modifying operations for the base table are rejected. For example, the following statement fails:

```
"insert into test.simple values(15, 'green')"
```

The following message is returned:

```
DB21034E The command was processed as an SQL statement because
it was not a valid command line processor command. During SQL processing
it returned:
SQL0668N Operation not allowed for reason code "1" on
table "SYSIBMTS"."SYSTSAUXLOG_IX114555". SQLSTATE=57016
```

7. Reset the tables.

```
db2ts "reset pending for table test.simple for text connect to mydb"
```

After successfully issuing the **RESET PENDING** command, the staging table is unlocked and modifications on the base table are again possible. Unlock the staging table either by issuing **RESET PENDING** command on the base table to unlock all dependent text-maintained staging tables, or with a **SET INTEGRITY** command on the specific staging table.

8. The text-maintained staging table now contains the changes that must be applied to the text search index. Issue an update command for the index.

```
db2ts "update index test.simpleix for text connect to mydb"
```

---

## Linguistic processing for DB2 Text Search

DB2 Text Search provides dictionary packs to support the linguistic processing of documents and queries. In addition, n-gram segmentation is supported for languages such as Chinese, Japanese, and Korean. As an alternative to dictionary-based word segmentation, the search engine provides an option to select n-gram segmentation for languages such as Chinese, Japanese, and Korean.

If a text document is in one of the supported languages, linguistic processing is carried out during the tokenization stage, that is when then text is broken up into

individual words. For unsupported languages, the document is parsed using white space or n-gram segmentation. Lemmatization (like stemming, this means to find the normalized form of a word, but it also analyzes the word's part of speech) is not performed on unsupported languages.

When you search a text search index, a match is indicated if the indexed document contains the query terms or linguistic variations of the query terms. The variations of a word depend on the language of the query.

## **Linguistic processing for Chinese, Japanese, and Korean documents**

For a search engine, getting good search results depends in large part on the techniques that are used to process text. After the text is extracted from the document, the first step in text processing is to identify the individual words in the text. Identifying the individual words in the text is referred to as segmentation. For many languages, white space (blanks, the end of a line, and certain punctuation) can be used to recognize word boundaries. However, Chinese, Japanese, and Korean do not use white space between characters to separate words, so other techniques must be used.

DB2 Text Search provides two processing options for Chinese, Japanese, and Korean: a morphological segmentation option, also called dictionary-based word segmentation, and an n-gram segmentation option (the default setting).

Morphological segmentation uses a language-specific dictionary to identify words in the sequence of characters in the document. This technique provides precise search results, because the dictionaries are used to identify word boundaries.

N-gram segmentation avoids the problem of identifying word boundaries, and instead indexes overlapping pairs of characters. Because two characters are used, this technique is also called bi-gram segmentation. N-gram segmentation always returns all matching documents that contain the search terms. However, this technique can return documents that do not match the query.

### **Example**

To show how both types of linguistic processing work, examine the following text in a document: election for governor of Kanagawa prefecture. In Japanese, this text contains eight characters. For this example, the eight characters are represented as A B C D E F G H. A sample query that users might enter could be election for governor, which is four characters and are represented as E F G H. (The document text and the sample query share similar characters.)

- After the document is indexed using morphological segmentation, the search engine segments the text election for governor of Kanagawa prefecture into the following sets of characters: ABC DEF GH.

The sample query election for governor is segmented into the following sets of characters EF GH. The characters EF do not appear in the tokens of the document text. Even though the document does not have EF, it does have DEF.

Since the document text contains DEF, but the query contains only EF, the document is less likely to be found by using the sample query.

When you enable morphological segmentation, you will likely see more precise results, but possibly fewer results.

- After the document is indexed using n-gram segmentation , the search engine segments the text election for governor of Kanagawa prefecture into the following sets of characters: AB BC CD DE EF FG GH.

The sample query election for governor is segmented into the following sets of characters: DE EF FG GH. If you search with the sample query election for governor, the document will be found by the query because the tokens for both the document text and the query appear in the same order.

When you enable n-gram segmentation, you will likely see more results but possibly less precise results. For example, in Japanese, if you search with the query Kyoto and a document in your index contains the text City of Tokyo, the query Kyoto will return the document with the text City of Tokyo. The reason is that City of Tokyo and Kyoto share two of the same Japanese characters.

---

## Scenario: Indexing and searching

After you have installed and configured DB2 Text Search, there are four steps that you must take before performing searches.

1. Start the DB2 Text Search instance services.
2. Prepare the database for use by DB2 Text Search.  
Enable the database and use the configure procedure to complete the Text Search server association. You must enable the database only once for DB2 Text Search. The configure procedure is necessary in the following cases:

- enablement was incomplete
- for partitioned databases
- for stand-alone Text Search server setups.

Note that you cannot enable Net Search Extender for a database once it has been enabled for DB2 Text Search.

3. Create a text search index on a column that contains, or will contain, text that you want to search.
4. Populate the text search index. This adds data to the empty, newly created text search index.

To set up automatic updates for text search indexes according to specified update frequencies, see the topic about scheduling a DB2 Text Search index update.

After a text search index contains data, you can search the index using an SQL statement and can search with XQuery if the index contains XML data.

As Figure 10 on page 15 shows, you should update existing text search indexes, either manually or automatically, to reflect changes to the text column that the index is associated with.

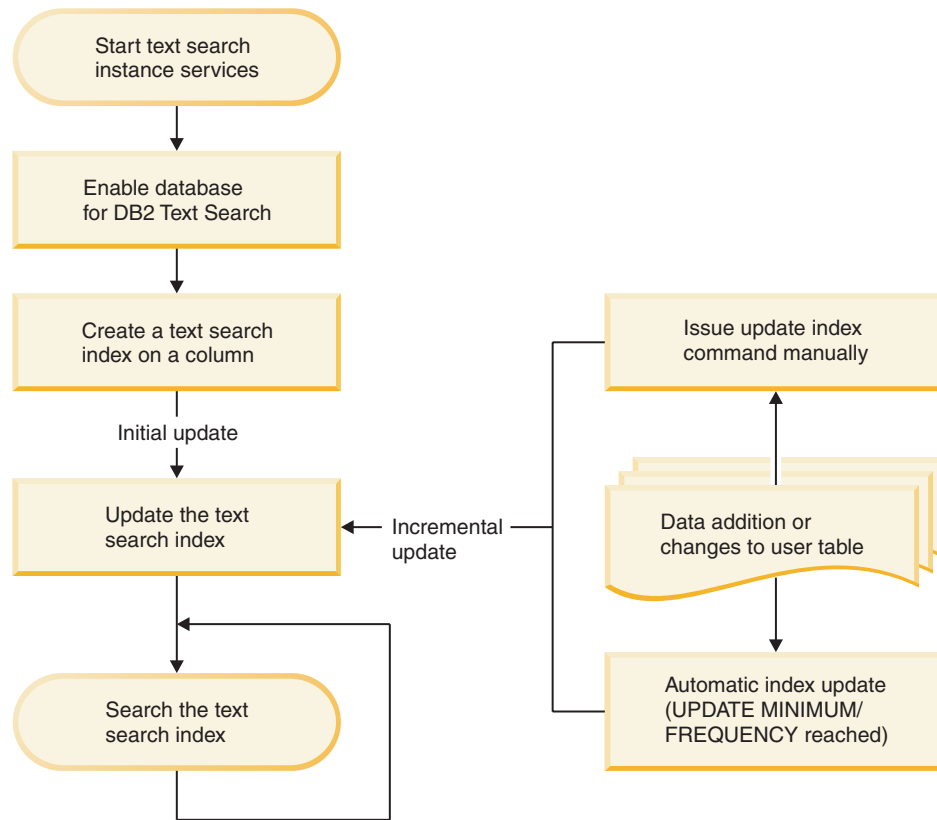


Figure 10. Setting up text search indexes for searching in a non-partitioned instance with an integrated Text Search server

## Basic scenario

Suppose that you want to make the products in the PRODUCT table in the SAMPLE database searchable by DB2 Text Search. Assuming that you already created the sample database (by running the `db2samp1` command) and that you set the `DB2DBDFT` environment variable to SAMPLE, you could issue the following commands:

```

db2ts START FOR TEXT
db2ts ENABLE DATABASE FOR TEXT
db2ts CREATE INDEX myschema.productindex FOR TEXT ON product(name)
db2ts UPDATE INDEX myschema.productindex FOR TEXT

```

The product names and descriptions contained in the NAME column of PRODUCT are now indexed and searchable. If you want to find the product IDs of all the snow shovels, you can issue the following search query:

```

db2 "SELECT pid FROM product WHERE CONTAINS (name, 'snow shovel') = 1"

```

## Coexistence scenario for DB2 Text Search and Net Search Extender

If a database is already enabled for Net Search Extender, and you want to use Text Search in that database, you can use the index coexistence feature to query the database.

Start the database for text search.

```
db2ts start for text
DB20000I The SQL command completed successfully.
```

Enable Text Search for a database where Net Search Extender indexes are already present.

```
db2ts enable database for text
CIE0000I Operation completed successfully
```

Create and update a DB2 Text Search index on a column which has an existing Net Search Extender index.

```
db2ts "CREATE INDEX db2ts.title_idx FOR TEXT ON books(title)"
CIE0000I Operation completed successfully.
```

```
db2ts "UPDATE INDEX db2ts.title_idx FOR TEXT"
CIE0000I Operation completed successfully.
```

Activate the new DB2 Text Search index to switch query processing from the NSE index to the new index.

```
db2ts "ALTER INDEX db2ts.title_idx FOR TEXT SET ACTIVE"
CIE0000I Operation completed successfully.
```

Issue a query to use the DB2 Text Search index.

```
db2 "select isbn, title from books where contains(title,'top')=1"
```

ISBN	TITLE
123-014014014	Climber's Mountain Tops
111-223334444	Top of the Mountain: Mountain Lore

2 record(s) selected.

Queries that attempt to use both types of text indexes are not supported. For example, here the title column has an active DB2 Text Search index, while the bookinfo column has an active Net Search Extender index. The search will return an error because all text indexes in one query must be of the same index type.

```
db2 "select isbn, title from books where contains(title, 'top')=1 and
contains(bookinfo, ' MOUNTAIN ')=1"
```

ISBN	TITLE
------	-------

```
SQL20425N Column "BOOKINFO" in table "BOOKS" was specified as an argument to
a text search function, but a text search index does not exist for the column.
SQLSTATE=38H12
```

To avoid this error, create a DB2 Text Search index on the bookinfo column and activate it.

```
db2ts "CREATE INDEX db2ts.bookinfo_idx FOR TEXT ON books( bookinfo )"
CIE0000I Operation completed successfully.
```

```
db2ts ALTER INDEX db2ts.bookinfo_idx FOR TEXT set active
CIE0000I Operation completed successfully.
```

---

## Rich text and proprietary format support

DB2 Text Search supports indexing and searching of documents in rich text format and proprietary formats within a properly configured DB2 Text Search instance.



DB2 Text Search supports TEXT, XML, and HTML text index formats to prepare indexes for full-text search on text data. In addition, the INSO format enables indexing and searching in documents with rich text or proprietary formatting:

- Rich text documents are documents that contain text as well as formatting instructions such as bold, italics, font types, font sizes, spacing, and more.
- Proprietary formats encompass a variety of common office products, such as, pdf, doc, ppt, ods.

For information about the enablement and configuration of the INSO format feature, see the topic about setting up DB2 Text Search for rich text and proprietary formats.



---

## Chapter 3. Text search solution planning

Understanding certain key concepts, such as supported document types and languages and user roles, will help you leverage the benefits of DB2 Text Search.

---

### Document characteristics

#### Document formats supported for DB2 Text Search

You must specify the format (or type) of text documents that you intend to search using DB2 Text Search. This information is necessary for indexing text documents.

The text column data can be plain text, HTML documents, XML documents, or documents with rich text or proprietary formatting. Documents are parsed to extract relevant parts for indexing, thus making them searchable. Some elements, for example, tags and metadata in an HTML document, are not indexed and thus not searchable.

#### Supported data types

The data types in the text columns that you want to index and search can be either binary or character.

DB2 Text Search supports the following data types:

- CHAR
- VARCHAR
- LONG VARCHAR
- CLOB
- DBCLOB
- BLOB
- GRAPHIC
- VARGRAPHIC
- LONG VARGRAPHIC
- XML

#### Conversion of unsupported formats and data types

You can use your own function to convert an unsupported format or data type into a supported format or data type.

By creating the text index using a user-defined function (UDF), you can convert an unsupported format to a supported format that can be processed during indexing by filtering the unsupported characters.

You can also use this approach for indexing documents that are stored in external unsupported data stores. In this case, where a DB2 column contains document references, you can use a UDF to return the content of documents that have the relevant document reference.

## Supported languages and code pages

You can specify that the text documents be parsed using a particular language when you first create a text search index. You can also specify that the query terms be interpreted in a particular language while searching. In addition, you can specify a code page when you create a text search index on a binary data type column.

### Language specification

A *locale* is a combination of language and territory (region or country) information and is represented by a five-character locale code. You define the message locale for a text search administration procedure by passing the procedure the locale code. Refinements of these locale codes are possible depending on the locales installed on the DB2 server.

There is an important difference between specifying a language when you create a text search index and specifying a language when you issue a search query:

- The locale that you specify in your **db2ts CREATE INDEX** command determines the language used to tokenize or analyze documents for indexing. If you know that all documents in the column to be indexed use a specific language, specify the applicable locale when you create the text search index. If you do not specify a locale, the database territory will be used to determine the default setting for **LANGUAGE**. To have your documents automatically scanned to determine the locale, in the SYSIBMTS.TSDEFAULTS view, set the **LANGUAGE** attribute to AUTO. The SYSIBMTS.TSDEFAULTS view describes database defaults for text search using attribute-value pairs.
- The locale that you specify in a search query is used to perform linguistic processing on the query and to help identify the base forms of the query term. After the locale of the base form has been identified, the locale does not play any part in the search process itself. Thus, you could use the English language for a query and obtain German documents in the search result if the search term in its base form is present in the documents.

The list of supported locales can be found [here](#).

### Code page specification

You can index documents if they use one of the supported DB2 code pages. Although specifying the code page when creating a text search index is optional, doing so helps to identify the character encoding of binary columns. If you do not specify a code page for binary columns, the code page from the column property is used. .

## Document size considerations

DB2 Text Search has limits on the size of a document that can be indexed and on the number of characters within that document.

The maximum size of documents that can be processed successfully is 100 MB. If a document exceeds the size limit, that document is rejected and an entry is created in the event table with that information, including the primary key to identify it. Processing continues for other documents that are a part of that update operation.

DB2 Text Search limits the number of Unicode characters that you can index for each text document. Sometimes, this character limit results in the truncation of large text documents in the text search index.

The default value for the number of Unicode characters allowed for each text document depends on the text document format:

- Text files that are larger than the value of *max.text.size* (in characters) are truncated to this size before they are indexed. The default value is 60 000 000 characters.
- XML files that are larger than the value of *max.xml.text.size* (in bytes) are not indexed. The default value is 60 000 000 bytes. The count includes tag names, attribute names, and attribute values, but not XML directives and comments.
- Binary files that are larger than the value of *max.binary.text.size* (in bytes) are not indexed. The default value is 60 000 000 bytes. This limit is applied after the document is transformed to text.

When the size of a text file exceeds the maximum text file size (60 million characters by default), the text file is truncated to the size limit before it is indexed. If a text document is truncated during the parsing stage, you receive a warning that some text was not processed correctly or completely.

When the size of a document in binary or XML format exceeds the maximum file size (60 million bytes by default), the document is not indexed and an error is generated.

Search results are incomplete if text is incorrectly or incompletely processed. If possible adjust the size limits or alternatively prune the document for processing. Details about the warning are written to the event table that was created for the text search index.

If you want to increase the file size limits, you must increase the heap size accordingly. You can use the configuration tool to adjust the maximum heap size by specifying the **startupHeapSize** parameter.

---

## DB2 Text Search security overview

DB2 Text Search executes administrative operations based on the authorization ID of the user executing the operation. Different to previous releases, there is no prerequisite for database privileges for the instance owner anymore, and it is not necessary for the fenced user to be in the same primary group as the instance owner.

Executing operations with the authorization ID of the user improves auditability and improves control of text search management. To simplify access control, three new system roles are available:

- Text Search Administrator (SYSTS\_ADM) - executes operations on database level
- Text Search Manager (SYSTS\_MGR) - executes operations on index level
- Text Search User (SYSTS\_USR) - has access to text search catalog data

The security administrator can grant or revoke these roles like user-defined roles, however, roles with prefix SYSTS are system managed otherwise and cannot be dropped or created.

When a database is created, the roles are automatically assigned to the database creator, and in non-restricted databases, the SYSTS\_USR role is assigned to PUBLIC. All other role assignments must be done explicitly by the security administrator, for example, SYSTS\_ADM to enable or disable text search.

In a restricted database setup, the security administrator must grant execute privileges for scheduler procedures to SYSTS\_MGR role and user privileges for the SYSTS\_USR role.

Table privileges to manage or access content in the SYSIBMTS catalog tables are automatically granted to the roles during database enablement for DB2 Text Search. Similarly, table privileges to manage or access content in the SYSIBMTS administration tables for a specific text search index are automatically granted to the roles during text index creation. For example, to create a text index you will need privileges on the base table corresponding to the privileges that are needed to create other types of indexes, and also the SYSTS\_MGR role which provides access privileges to the SYSIBMTS tables.

Certain index-level commands require a connection to the text search server. The relevant connection information is retrieved from the SYSIBMTS.TSSERVERS administrative view and includes an authentication token. The token is generated when the text search server is configured and used as an identification mechanism by callers to ensure that the right text search server is addressed. If the wrong token is used, the index management or search request is rejected.

The following table provides a summary of required role privileges. The security administrator must have granted the appropriate role to the user for successful execution of an operation.

*Table 1. Role privileges*

	<b>Role</b>	<b>Operation</b>
Text Search Administrator	SYSTS_ADM	Enable, Disable, Clear command locks (all), Configure
Text Search Manager	SYSTS_MGR	Create, Update, Alter, Drop, Clear Events, Clear command locks (per index), Reset Pending
Text Search User	SYSTS_USR	Limited access to the text search SYSIBMTS catalog

## User roles

There are different user roles and authorizations for users of DB2 Text Search. System roles control execution privileges for administrative operations and the authorization ID of the user thus needs the adequate text search role in addition to database or table access privileges to execute a text search operation.

Typical users are:

- Text Search Server Administrator
- Text Search Administrator
- Text Search Index Manager
- Users performing text search queries

## DB2 Text Search Server Administrator

The Text Search Server Administrator configures DB2 Text Search server options, starts and stops the text search instance services for integrated and stand-alone text server deployments and monitors text search server operation.

For integrated text search server setups this role is tied to the database instance owner.

The instance owner is determined differently on UNIX and Windows operating systems:

- On UNIX operating systems, the instance owner user is the name and user ID of the instance specified for the **db2icrt** command.
- On Windows operating systems, the instance owner is the user ID running the DB2 instance service.

Contrary to DB2 Version 9.7, the instance owner does not need to hold database privileges. For stand-alone text search server setups, the server administrator must have appropriate access to text search server executable, configuration and index files.

## Text Search Administrator

The Text Search Administrator enables and disables databases for use with DB2 Text Search. Another main task that the Text Search Administrator performs is clearing command locks.

The text search administrator requires the SYSTS\_ADM role in addition to DBADM authorization, which allows the manipulation of all database objects, including text search indexes.

## Text Search Index Manager

The Text Search Index Manager defines and maintains text search indexes.

Typical tasks are:

- Creating text search indexes and defining their characteristics
- Updating text search indexes
- Changing the update characteristics of text search indexes
- Dropping text search indexes
- Clearing the event table periodically

Text Search Index Managers have the SYSTS\_MGR role and usually have CONTROL privilege for the table on which a text search index is created.

## User performing text search queries

Users who perform search queries can use the DB2 Text Search CONTAINS and SCORE functions in an SQL query against a user table. They can also use the xmlcolumn-contains function in an XQuery that references a table with a text search index.

There is no specific DB2 Text Search search authorization. Depending on the access rights that the users are granted on the table that the text search index is created on, the query is permitted or rejected. If users can issue a SELECT statement on a given table, they can also perform a text search on that table.

Users performing the search queries can for example include the following functionality in their queries:

- Limit the text search to a particular document (using SQL or XQuery)
- Return a score indicating how well a document compares with other matching documents for a given search argument (using SQL)

## Access policies and communication security

### File access considerations for the Text Search server

The process owner of the text server process requires read and write access to configuration data and all collection data, including collections located in custom collection directories.

For the integrated text server the process owner is the instance owner, for stand-alone text servers it is the user who starts the text server with the startup command.

Collections may include confidential data that can be partially readable when opening a file directly. To prevent unauthorized access, check and update the access permissions to configuration and collection directories to ensure that only the process owners of the text server may access the files.

### Staging table access policies

To identify changes that need to be applied to a text index, the primary key of modified rows (inserted, updated, deleted) is inserted into the staging table.

The primary key may be based on data columns of the base table that contain confidential data. By default, users with role SYSTS\_ADM and SYSTS\_MGR, and with some restrictions, SYSTS\_USR, have at least read access to the content of staging tables. Access and audit policies for the base table are not inherited for the staging table. If further restrictions for access to a particular staging table are needed, the security administrator will need to revoke read privileges on the specific table for the roles and grant them to a user or a custom role who will manage the specific text index.

### Stand-alone setup

The DB2 database instance uses TCP/IP to communicate with the stand-alone DB2 Text Search server. SSL or GSKit support are not available, however, encryption channels can be used through the stunnel program or SSH tunneling. Restrict access to your document repository and text search index files depending on your security requirements. The stand-alone text search server must be installed on computers with a secure network connection, behind a firewall to prevent unauthorized access to the text search indexes. Setting up TCP/IP access restriction to the stand-alone text search server ensures that it can only be accessed by the host on which the database server is installed.

---

## DB2 Text Search capacity planning and optimization

A number of factors influence performance and resource use in DB2 Text Search. When planning system capacity for DB2 Text Search, consider the query workload, the number of parallel index updates, the expected size and growth rates of your text indexes, and the processing time for the documents you are indexing.



DB2 Text Search enables full-text search queries on most data types within the DB2 database, including support for XML documents and a rich-text or proprietary format feature. Full-text search is supported through a text search server instance that is integrated with the database instance or in a stand-alone setup associated with the database instance. Communication between the database and text search server instance is through TCP/IP. Full-text indexing and search performance depend on the text search server configuration, available system resources, and text index specific settings.

## **Text search server deployment and configuration**

A single text search server is configured for the database instance. The text search server has a recommended minimum memory requirement of 4 GB of memory for production use, which increases according to the number of parallel index updates.

Updating the text search index is resource-intensive, both in terms of disk I/O and CPU or memory requirements. Multiple configuration parameters are available to control the Text Search server resource usage. For workload distribution, for example, in a partitioned database environment, a stand-alone setup is recommended.

## **Size of text search indexes**

On average, a text search index is about 50-150% of the original data.

There is no absolute size limit for text search indexes, however, the combination of throughput factors with completion time dependencies results in practical limits on the total text search index size. For example, when a considerable amount of data is added to or removed from a text search index, the text search index structure is merged to improve query performance, and the time for completion of the merge depends on the size of the index.

## **Factors affecting throughput**

Absolute text index update throughput depends on the data type and the index format. For perceived query performance, the biggest impact is due to the number of matching results, not the size of the text search index. For example, a query with a single predicate using a single-term search term on a 100 GB text search index performs similar to a search on an 800 GB text search index if the number of results is the same.

Optimal processing for text index updates occurs when there is approximately 10-100 KB of text per document. Throughput degrades above 1 MB and below 1 KB of text.

## **DB2 Text Search server configuration**

You can tune your DB2 Text Search configuration by adjusting the queue sizes, heap size, number of indexing threads, and other factors. Balance your adjustments to these different parameters for optimal performance of your system.

For the DB2 Text Search server configuration the number of indexer threads should not exceed the number of CPUs, and the number of parallel updates should not exceed the number of indexer threads. Note that to determine the number of parallel updates in a partitioned database the number of indexes is multiplied with the number of collections for a text index.

Stop the DB2 Text Search instance services using the **db2ts STOP FOR TEXT** command before making any configuration changes.

Start the configUtility.

- For an integrated text search server it is located in the `<TS_HOME>/bin` directory.
- For an stand-alone text search server it is located in the `<ECMTS_HOME>/bin` directory.

For example, to change the number of indexing threads:

```
configTool configureParams -configPath configPath -numberOfIndexingThreads 3
```

For your changes to take effect, restart the DB2 Text Search processes.

## Maximum heap size configuration

When a document is received by the document ingestion thread, its content is placed in the document queue. Documents placed on the document queue remain there until an active indexing thread indexes it. In a typical operation, the speed of placing documents on the document queue is faster than the time required to parse and index the document. Therefore, at some point in time, the document queue reaches its capacity, and the document ingestion thread is blocked until another slot is freed from the document queue.

As the document queue fills with unprocessed documents, it consumes heap memory. Further memory is consumed for document processing like parsing and indexing. The combined heap memory consumption must be less than the maximum heap size of the process. By default, the heap size is configured to be 1500 MB.

Also, consider the ratio between the input and output queue memory size and the heap memory. The queue size is determined by the memory consumption of the documents in the queue. If you intend to process long documents, like 20 MB each, and decide to increase the queue memory size, consider increasing the heap size.

The *startupHeapSize* variable sets the maximum allowed heap size for the integrated or the stand-alone DB2 Text Search server. The default startup heap size is 1.5 GB. This value must be a number between 1.5 GB and the maximum amount of memory allowed by your operating system and JVM version. Consider the following examples:

- If you have a Windows system with a 32-bit JVM, then a process can have a maximum heap size of 2 GB. Therefore, your *startupHeapSize* parameter must be set to less than 2 GB. For example, 1.8 GB.
- If you have an AIX<sup>®</sup> system with a 64-bit JVM, then the maximum heap size is limited only by the amount of virtual memory configured on the system. If many large documents with an average size of 20 MB must be processed continuously, then increase the *startupHeapSize* parameter to approximately 4 GB.

You can set the maximum heap size when you install or upgrade the stand-alone DB2 Text Search server by specifying the **IA\_STARTUP\_HEAP\_SIZE** parameter in the response file. When you set the maximum heap size to a value greater than 2 GB during the installation or upgrade of the stand-alone text search server on a 64-bit operating system, file size limits for text, XML, and binary documents are increased for new collections. File size limits are specified per collection in the `<ECMTS_HOME>\config\collections\collection_name\ parser_config.xml` file. The default file size limits for new collections are specified in the `<ECMTS_HOME>\config\`

defaults\parser\_config.xml file. For each 8.3 MB of heap memory over 2 GBs, the values of the file size limits (60 MB by default) are increased by 1 MB (up to 400 MB).

**Attention:** When you modify the maximum heap size by using the configuration tool after installation, you must manually adjust the file size limits in the parser\_config.xml file. File size limits are automatically adjusted only during installation and upgrade when you specify the **IA\_STARTUP\_HEAP\_SIZE** parameter in the response file.

To change the maximum heap, issue the following command:

```
configTool configureParams -configPath <full-path-to-configuration-folder>
-startupHeapSize <value>
```

where, <value> is the heap size and <full-path-to-configuration-folder> is the full path to the config.xml file for DB2 Text Search server.

On a 32-bit operating system, the typical configuration is:

- Maximum heap size: 1.8 GB
- Queue sizes: 90 MB each
- File size limits: 60 MB

On a 64-bit operating system, the typical configuration is:

- Maximum heap size: 3 GB
- Queue sizes: 150 MB each
- File size limits: 200 MB

## DB2 Text Search indexing threads

Multiple indexing threads work in parallel to parse and index documents. This usually reduces the total elapsed time for text search index updates.

Indexer threads pick documents from the queue and manage the indexing process. They make use of index preprocessing threads to prepare the document content for indexing and write the result to the text index collection.

Index preprocessing threads extract text, identify the language, tokenize and analyze the document.

Usually the number of indexer threads and index preprocessing threads is configured to be the same. However, in some scenarios, for example, when large documents are processed, increasing the number of preprocessing threads might provide a performance benefit.

## Indexing thread usage

If multiple indexer threads work on the same collection, the effect is reduced by the coordination required to synchronize the processing among the threads. Also, indexing threads that are single threaded perform better while parsing, but there can be a performance hit while merging or writing to disk. For example, four indexing threads working on four different text indexes show better throughput than four indexing threads working on a single text index.

## Number of indexing threads

You should have at least two indexing threads and ensure that the number of indexing threads does not exceed the number of available CPUs. The maximum number of parallel index updates should not exceed the number of indexing threads to avoid thread sharing. With too many indexing threads or too many parallel index updates, the overall system performance suffers due to memory usage for process context switches.

For example, if 40 text indexes are frequently updated, and the system contains 8 CPUs, do not use more than eight indexing threads. Also, use a staggered update schedule for the text indexes to minimize contention for index threads.

The default setting for the number of indexer threads is 4, the same default applies to index preprocessing threads.

To configure the number of indexing threads, issue the following command:

```
configTool configureParams -configPath <full-path-to-configuration-folder>  
-numberOfIndexerThreads <value>
```

where *<value>* is the number of threads and *<full-path-to-configuration-folder>* is the full path to the config.xml file for the DB2 Text Search server.

To configure the number of preprocessing threads, issue the following command:

```
configTool configureParams -configPath <full-path-to-configuration-folder>  
-numberOfPreprocessingThreads <value>
```

where *<value>* is the number of threads and *<full-path-to-configuration-folder>* is the full path to the config.xml file for the DB2 Text Search server.

## DB2 Text Search queue memory size

The queue memory size for DB2 Text Search must be set properly for optimal index update processing. Queue memory assignment can be controlled both for the database and for the text server.

The database queue memory determines the number of documents that can be sent to the text server for update processing at any time. To control the size of the database queue memory, update the SYSIBMTS.TSDEFAULTS administration view and set the value for the **DocumentResultQueueSize** parameter. The default value is 10,000. This value is used to limit how much database memory is reserved per update operation for a collection. Note that on a multi-partition setup, a single text index update that is configured for parallel execution will reserve memory space for each collection that needs an update.

The second mechanism for queue memory control applies to the text server. Two configuration values determine the use of queue memory.

- *inputQueueMemorySize*:  
Specifies the memory size of the input queue on the indexing server. The input queue contains documents that are waiting for preprocessing. A larger memory size will be faster, but will consume more resources. The default size is 15 MB.
- *outputQueueMemorySize*:  
Specifies the memory size of the output queue on the indexing server. The output queue contains documents that are waiting to be indexed after preprocessing. A larger memory size will be faster, but will consume more resources. The default size is 15 MB.

Consider the ratio between the input and output queue's memory size and the heap memory. The queue size is determined by the memory consumption of the documents in the queue. If you intend to process long documents, for example 20 MB each, consider increasing the queue memory size and increasing the heap size.

To change, for example, the *inputQueueMemory* size, issue the following command:

```
configTool configureParams -configPath <full-path-to-configuration-folder>
-inputQueueMemorySize <value>
```

where *<value>* is the memory size and *<full-path-to-configuration-folder>* is the full path to the config.xml file for DB2 Text Search.

## DB2 Text Search index planning and optimization

Data source characteristics have major impact on performance.

The time required to complete a text index update depends mainly on the following factors:

- the number of documents to be indexed
- the document size
- the index type
- index update parallelism
- text search server configuration

The processing time for each document is the sum of an approximate fixed time and a variable time. The fixed time is influenced by the document type, such as plain text, XML or INSO. The fixed time is approximate because there can be minor variations in time for memory usage or reuse. The variable time is determined mainly by the document size and linguistic processing variations.

For indexes of INSO documents, handling different MIME types can also affect the processing time.

The number of documents that can be processed in a given timeframe increases for smaller document sizes. However, the total throughput is less for smaller documents than for larger documents due to the fixed cost per document.

### DB2 Text Search index source characteristics

To enhance performance during indexing or search, use the following techniques:

- For primary key columns, use numeric data types, such as INTEGER, instead of a VARCHAR type. Avoid primary keys that are a compound of multiple VARCHAR columns to minimize traffic for query results.
- Ensure that your system has enough real memory available for the index update operation. Index updates require memory that is in addition to that required for any database buffer pools. If there is insufficient memory, the operating system uses paging space instead which decreases search performance considerably.
- If large numbers of small documents must be processed in text search server index updates, consider reducing the number of parallel index updates and instead increase the queue sizes to increase the maximum flow of documents to the text server. See the capacity planning topics for details.
- Ensure that the content to be indexed is accessible and of proper format, as the performance might decrease during an index update if many error and warning messages are written to the event table.

## Asynchronous index updates

To improve performance, a text search index is not synchronized with its associated user table within the scope of a DB2 transaction that updates, deletes text documents from, or inserts text documents into that table. Instead, text search indexes are updated asynchronously.

To facilitate the asynchronous update of a text search index, a staging table (also known as a log table), is created for each text search index. Triggers on the text table capture any changes, for example, inserts, deletes, or updates, to a text column that the text search index is associated with and write them to the staging table. An auxiliary staging table may be used to capture changes that are recognized through integrity processing. The updates to the text search index are applied at a later stage, during either a manual update or an automatic update. The update is made to a copy of a very small part of the index. During the update, you can still perform searches on the index, but you cannot access the updated text search index until the synchronization is complete.

When using the `updateautocommit` option, consider that each commit cycle requires significant processor usage, which increases the total time for completing an index update. Setting this `updateautocommit` option is especially useful for initial updates, or updates that involve all rows or a majority of rows, if the total elapsed time is large to avoid losing completed work due to a rollback caused by a system or server failure.

## Optimizing a DB2 Text Search index

DB2 Text Search index optimization compacts the text search index and speeds up indexing and searching. Optimization removes deleted documents from the text search index and merges the index segment files on the disk.

Optimization and indexing of the same index cannot be performed in parallel. Take this into account when scheduling optimization and indexing sessions. However, optimization and search can be performed in parallel. Disk space consumption during index optimization can be high, especially if the same index is searched in parallel.

You can optimize the index after you completely index your document set or after incremental index updates. Index optimization can take a long time, depending on the index size. If your incremental updates add documents frequently, perform optimization less frequently to minimize the extra processor usage for the optimization process.

To optimize the index:

1. From the `ECMTS_HOME/bin` directory, start the administration tool with the **optimizeIndex** command. For example:  

```
adminTool.bat optimizeIndex -configPath  
"C:\Program Files\IBM\ECMTextSearch\config"  
-collectionName MyCollection
```
2. You can check the status of the last executed optimization process by running the administration tool with the **optimizeIndexStatus** command.

## Disk consumption

### Text index size

The amount of disk space a text search index uses depends highly on the nature of the text in each document. However, there is an approximately linear relationship



between the disk space required for the text search index and the disk space required for the original data. Typically, the size of the index on the disk is 50 - 150% of the original text size. For example, on a table with an integer primary key the text search index for 100,000 20 KB documents is expected to require about 1100 MB of disk space (100,000 x 20 KB x 55%). The size of the text search index relative to the source documents depends on the following factors:

- the average size of the document
- the size of the document key (the primary key columns)
- the number of sortable fields
- the number and distribution of unique terms

During the index update, additional work space is needed. The intermediate space requirements are about a factor 2-3 times the final text search index size, provided the maximum segment size is not reached. The free space required is 2-3 times the maximum segment size. Disk space is reserved even after a segment merge if the old segments have been used in a search.

## Log files

In addition to the `db2diag.log` file, DB2 Text Search generates trace and Configuration tool log files with messages from the DB2 Text Search server.

For an integrated Text Search sever, the default log file location is `db2tss/1og` directory. If you want DB2 database and text search logs in the same location, set the location to `<instanceHome>/sql1lib/db2dump/ts1og` on UNIX or `<instanceProfilePath>\<instance_name>\db2tss\ts1og` on Windows platforms.

For the stand-alone setup, the default location for the DB2 Text Search server logs is `<ECMTS_HOME>/1og`. You can change the default location during installation by setting the **IA\_LOG\_PATH** parameter in the response file.

In either case, ensure that the target location has sufficient free disk space for the log files. A minimum of 100 MB of free disk space is required. Without sufficient space for the log files, the text search service stops logging and throws a disk full error.

## Administrative tables

If you do not specify a table space for the administrative tables for the text search index when you run the **CREATE INDEX FOR TEXT** command, the administrative tables are created in the table space that contains the base table. To determine the appropriate location, consider the following information:

- Staging table for the text index

The staging table holds the reference to rows that have been updated in the base table for an incremental update of the text index. This table is automatically cleaned up with each update:

Size =  
number of rows for index updates \* (length of primary key of base table + 18)

- Event table for the text index

The event table contains status information about text index processing, including errors and warnings during an index update. In the worst case, if each document is rejected due to a nonfatal error, the number of events is the number of documents plus a few begin and end messages for the update process. The

event table is not cleaned automatically, and increases in size until a **CLEAR EVENTS FOR INDEX** operation is completed.

Event table size =  
number of events \* (length of primary key of base table + 1050)

## DB2 Text Search index location

It is important to note that the default index location has changed in this release.

For an integrated Text Search sever, configuration and collection metadata is stored in *instanceHome*/sql/lib/db2tss/config on UNIX or *instanceProfilePath*\instance\_name\db2tss\config on Windows.

The configuration and collection metadata for each text search index require little space. However, unless a custom path is specified, the location for text search indexes is in a subdirectory of db2tss/config. This location is often restricted in size, it is therefore strongly recommended to configure the **defaultDataDirectory** parameter for the Text Search server to a custom location with sufficient disk space if you plan to create multiple or large indexes with an integrated Text Search server.

The location of collection data is determined when you create a collection and is stored in the collection.xml file. For stand-alone DB2 Text Search servers, the location of configuration files for collections is determined by the **defaultDataDirectory** parameter. By default, the collection configuration directory is <ECMTS\_HOME>\config\collections, while the collection data is in a subdirectory under the **defaultDataDirectory**\collection\_name\data\text collection configuration directory.

In any case, if you plan to create multiple large indexes, consider storing them on separate or striped disk devices, in particular if concurrent index updates are scheduled.

## Index specific parameters for DB2 Text Search index updates

You can configure the following collection-specific parameters to improve performance:

- **MaxMergeDocs**
- **MaxMergeMB**
- **MergeFactor**
- **BufferSize**

You can modify indexing parameters for a particular collection by editing the *ECMTS\_HOME*\config\collections\collection\_name\collection.xml file. To modify the default settings for future collections that are created, set the values of these parameters in the *ECMTS\_HOME*\config\defaults\collection.xml file.

- The **MaxMergeDocs** parameter defines the largest segment (measured by the number of documents) that can be merged with other segments in the index. There is a trade-off between overall indexing throughput and segment merge time.

If you specify a low value for the **MaxMergeDocs** parameter (for example, 100,000 documents), your segments will be limited in size. In this case, segment merges are quicker and indexing flows more smoothly without time-outs. However, if your content is very large, there will be numerous segments and a degradation in indexing throughput over time.



If you specify a high value for the **MaxMergeDocs** parameter (for example, 100,000,000 or 500,000,000 documents), you get fewer segments (until the index becomes very large) and the overall indexing throughput is better. However, segment merges take more time and you might encounter time-outs during indexing.

Typically the value of **MaxMergeDocs** should be higher for collections of small documents and lower for collections of larger documents.

- The **MaxMergeMB** parameter defines the largest segment, measured by the physical size of the file, that can be merged with other segments in the index.

There is a trade-off between overall indexing throughput and segment merge time. If you specify a low value for the **MaxMergeMB** parameter, for example 500 MB, your segments will be limited in size. In this case, segment merges are quicker and indexing flows more smoothly. However, if your content is very large, there will be numerous segments and a degradation in indexing throughput over time, as well as degradation in search performance.

If you specify a high value for the **MaxMergeMB** parameter, for example 50,000 MB or 100,000 MB, you get fewer segments (until the index becomes very large) and the overall indexing throughput is better. However, segment merges take more time and you might encounter time-outs during indexing.

- The **MergeFactor** parameter defines the number of segments that are merged at a time and also controls the total number of segments that can accumulate in the index. There is a trade-off between frequent, small merges (for example, two at a time) and less frequent, large merges (for example, 10 at a time). You can specify a smaller value for the **MergeFactor** parameter to avoid time-outs. Modifying the merge factor does not typically impact performance.
- The **BufferSize** parameter specifies the amount of RAM that can be used for buffering added documents before the documents are flushed as a new segment. There is a trade-off between frequent, small flushes to disk and less frequent, large flushes to disk. In some cases you can improve performance by increasing the value of the **BufferSize** parameter. For example, when you index a single collection of small documents, increasing the buffer size will improve performance, especially for the first 100,000 documents in the index.

## DB2 Text Search system tuning

Text index update processing and text search query performance are influenced by various system characteristics.

Take the following into consideration:

- TCP/IP port considerations for Windows
- File descriptors

### TCP/IP port considerations for DB2 Text Search and Windows

On 32-bit Windows operating systems, your ability to handle high query loads is affected by the number of TCP/IP ports and the wait time to reuse a port.

### Port assignments on Windows (32-bit)

The integrated DB2 Text Search runs as a separate process on the same host as the database server. The database server and text server communicate through a TCP/IP connection.

The number of available ports for TCP/IP connections is influenced by the number of ports and the wait time to reuse a port after a connection is closed. The default configuration values for these parameters might not be sufficient to provide

enough available ports to serve a high query load. If you have too few TCP/IP ports, you might get an CIE00756 Connection failed error.

If a CIE00756 Connection failed error occurs, run the following commands to view port usage on the server:

```
netstat -n
netstat -n | c:\windows\system32\find /I <port_number>
```

If the output shows many TCP/IP connections and local addresses `127.0.0.1:port_number` in `TIME_WAIT` state, the server is likely running out of TCP/IP ports.

You can determine the DB2 Text Search port numbers by issuing the following command:

```
configTool printAdminHTTPPort -configPath %INSTPROF%\%DB2INSTANCE%\db2tss\config
```

where, `INSTPROF` is set to the value of the **DB2INSTPROF** registry variable applicable to integrated DB2 Text Search server setups.

## Port settings

Port settings are controlled by the following registry entries that are found in `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\TCP/IP\Parameters`:

- **TcpTimedWaitDelay**

A `DWORD` value, in the range 30 - 300, that determines the time in seconds that elapses before TCP/IP can release a closed connection and reuse its resources. Set the **TcpTimedWaitDelay** value to a low value to reduce the amount of time that sockets stay in `TIME_WAIT` state.

- **MaxUserPort**

A `DWORD` value that determines the highest port number that TCP/IP can assign when an application requests an available user port. Set **MaxUserPort** to a high value to increase the total number of sockets that can be connected to the port.

A system making many connection requests might perform better if **TcpTimedWaitDelay** is set to 30 seconds, and **MaxUserPort** is set to 32678.

After adding or changing the registry entries, reboot the Windows machine to reflect the changes.

## DB2 Text Search file descriptors

For DB2 Text Search index updates and queries, system resources such as file descriptors are consumed to handle multiple index update and search requests.

In a typical system, the number of open file descriptors per process may be limited to a relatively small number like 1024, which can result in the text search server running out of file descriptors. If this occurs, the search and update requests will fail.

To resolve this error

- Check the server logs for an exception with the message string similar to too many open files.
- On UNIX systems, check the system limits with `ulimit -a`.

To increase file descriptors, follow these steps:

1. Shut down the text search server.
2. Increase the number of file descriptors per process by following your operating system manual. This increase in file descriptors must be sufficient to accommodate all requests across login sessions.
3. Restart the text search server.

---

## DB2 Text Search query planning

There are several aspects to consider when planning your text search query.

### DB2 Text Search arguments

Wildcard characters and their expansion limit, the case sensitivity of arguments, and argument options are different types of text search arguments that can all affect query performance.

#### Wildcard characters

Using a wildcard character at the beginning of a search term slows query processing. Where possible, avoid performing searches such as *\*search\_term* or *?search\_term*.

#### Wildcard expansion limit

When a query term includes a wildcard, the query term is expanded so that matching documents are retrieved. A text index collection might include more distinct matching terms than the wildcard expansion limit allows. In that case, only the subset of documents that match the already expanded terms is returned. This limitation applies to the asterisk (\*) wildcard character .

By default, 1024 terms can be returned. To change this limit, specify the **queryExpansionLimit** parameter and a value for the parameter in the *ECMTS\_HOME*\config\config.xml file. For example, to set the limit to 4096, add the following line to the file:

```
<queryExpansionLimit>4096</queryExpansionLimit>
```

#### Case sensitivity

Text search arguments are not case sensitive, even if you specify an exact term or phrase by using double quotation marks. For example, a search for the term "Hamlet" can return both the Shakespearean play Hamlet and hamlet, the term for a small village.

#### Search argument options

Search argument options are properties of the search argument. For example, in the following search query for the word bank, the options of the QUERYLANGUAGE search argument are different:

```
...CONTAINS(column, 'bank', 'QUERYLANGUAGE=en_US')  
and CONTAINS(column, 'bank', 'QUERYLANGUAGE=de_DE')...
```

### DB2 Text Search multiple predicates

If a query contains multiple predicates, consider the following limitations depending on how the predicates are organized.

## UNION versus OR operators

Query performance might improve by using UNION instead of OR to combine multiple predicates.

### Using a JOIN

Text search functions can be a predicate in an outer join, with limitations for LEFT OUTER JOIN and FULL OUTER JOIN. For these cases a text search predicate can only be applied if the search on this text index can be joined back with the primary key of its base table. For example, the following type of query is supported:

```
select place.placenum, location.description from place
LEFT OUTER JOIN location on (location.mgrid = place.ownerid)
where
(location.description is null and contains(place.description, 'Paris')=1 )
```

The CONTAINS and SCORE functions are not supported as a predicate in a LEFT OUTER JOIN or FULL OUTER JOIN.

## DB2 Text Search locale and language

Locale specification can also impact the performance of a text search query.

### Locale specification

When you perform a search on a text search index in a multi-lingual environment, it is suggested that you always use the QUERYLANGUAGE option with your search query to specify which locale (a combination of language and territory information) to use to interpret a search term. For example, if you have a search term such as bald, you can specify to treat it as an English word by setting the QUERYLANGUAGE=en\_US in the search query. Similarly, if you want it to be treated as a German word, QUERYLANGUAGE can be set to de\_DE. However, it should be noted that the results returned are highly dependent on the LANGUAGE used for indexing, regardless of the QUERYLANGUAGE specified in a query.

If the QUERYLANGUAGE is not specified in the search query, then the following logic is used:

- The search term is interpreted to be of the locale that was set for the underlying text index during index creation.
- If the locale set for the index during index creation is AUTO, then this defaults to English (en\_US), and the search term will be treated as an English word.

Restrictions:

- If the locale specified in the search queries is invalid (for example, QUERYLANGUAGE=Mongolian), then the query will be considered invalid and an exception will be thrown.
- Setting QUERYLANGUAGE=AUTO in the search query is an unsupported option and the results of the query are undefined.

Note that the locale specified by QUERYLANGUAGE has no effect on the locale of error messages resulting from search queries. The error-message locale that is used depends on whether you started the text search instance services. If you did not start them, messages are written using en\_US; if you did start them, messages are written in the same locale of the environment in which you issued the **START FOR TEXT** command.

## DB2 Text Search SCORE function

The score of a document is dynamic and calculated independently for each query.

Updates to a document as well as adding or removing documents from a text index can cause a change of the score of a document for a query term.

Assume there is a set of documents discussing transportation and pollution. If you want to locate documents containing references to both terms, but only if the term pollution scores higher than the term transportation, you can use the following command:

```
SELECT document_id
FROM document_library
WHERE SCORE(document_content, 'pollution') >
SCORE(document_content, 'transportation')
and CONTAINS(document_content, 'transportation pollution') = 1
```

To enhance performance, you can format your query to use the boost (^) modifier so that the search function is run only once, as follows:

```
SELECT document_id
FROM document_library
WHERE SCORE(document_content, 'pollution^10 transportation') > 0
ORDER BY SCORE(document_content, 'pollution^10 transportation') DESC
```

The first query does not return any results if pollution scores low. The second query gives higher importance to pollution but still returns documents if pollution scores low in all documents.

## DB2 Text Search RESULTLIMIT function

Multiple instances of **RESULTLIMIT** within a query require the same search argument to produce predictable results.

### Description

If you use multiple text searches that specify **RESULTLIMIT** in the same query, use the same search argument. Using different text search arguments might not return the expected results.

For example, in the following query, it is unpredictable whether the 10 documents specified by **RESULTLIMIT** will be returned:

```
SELECT EMPNO
FROM EMP_RESUME WHERE RESUME_FORMAT = 'ascii'
AND CONTAINS(RESUME, '"ruby on rails"', 'RESULTLIMIT=10') = 1
AND CONTAINS(RESUME, '"java script"', 'RESULTLIMIT=10') = 1
```

Instead, use **RESULTLIMIT** as follows:

```
SELECT EMPNO
FROM EMP_RESUME WHERE RESUME_FORMAT = 'ascii'
AND CONTAINS(RESUME, '"java script" "ruby on rails"', 'RESULTLIMIT=10') = 1
```

Note that this method works only when both **CONTAINS** functions are operating on the same table column. If they are not operating on the same column, try using **FETCH FIRST n ROWS** to improve query performance.

## Parser configuration for DB2 Text Search

You can configure some of the settings that are used for XML search.

All parser configuration parameters are located in the `parser_config.xml` file, in the XML element defining the parser, `com.ibm.es.nuvo.parser.xml.XMLParser`. Each parameter is specified by a `Parameter` element of this form:

```
<Parameter Name="parameter">setting</Parameter>
```

**ParserName: text**

**ParserClass: com.ibm.es.nuvo.parser.text.TextParser**

The class that is invoked when the content type is textual.

**required.text.confidence**

Not in use.

**fall.back.parser**

The parser that is activated when the text parser fails, the content type is specified as unknown, and content detection identifies the content as Binary.

**fall.back.encoding**

The encoding that is used when the encoding is specified as unknown or null.

**detection.encoding.buffer.size**

The buffer size (in bytes) that is passed to the content detection mechanism to determine the encoding. The default is 2000 bytes.

**ParserName: xml**

**titleTagNameList**

A comma separated list of tags that are handled as title fields.

**maxTextUnicodeChars**

Not in use.

**handleExternalFiles**

Not in use.

**handleSkippedEntities**

Not in use.

## DB2 Text Search XML namespaces

Searching on XML namespaces requires a workaround.

You can index XML documents that contain namespace bindings without generating errors, but the namespace information is removed from each tag. As a result, text searches on XML documents with namespace bindings can lead to undesired results.

However, there is a workaround to this limitation for queries that use DB2 XQuery. The DB2 Text Search engine is not namespace aware, but you can use the DB2 XQuery support for namespaces to do namespace filtering for the unwanted documents returned from a text search.

Consider the following example in which the default database environment variable is set to `SAMPLE` and a text search index called `prod_desc_idx` is created on the `PRODUCT` table:

```
db2ts "ENABLE DATABASE FOR TEXT"  
db2ts "CREATE INDEX prod_desc_idx FOR TEXT ON product(description)"
```

Now, a new row with the namespace `http://posample.org/wheelshovel` is added to the `PRODUCT` table, which already has two XML documents with the namespace `http://posample.org`:

```
INSERT INTO PRODUCT VALUES ('100-104-01', 'Wheeled Snow Shovel',
99.99, NULL, NULL, NULL, XMLPARSE(DOCUMENT '<product xmlns=
"http://posample.org/wheelshovel" pid="100-104-01">
<description><name>Wheeled Snow Shovel</name><details>
Wheeled Snow Shovel, lever assisted, ergonomic foam grips,
gravel wheel, clears away snow 3 times faster</details>
<price>99.99</price></description></product>'))
```

The text search index is then updated, as follows:

```
db2ts "UPDATE INDEX prod_desc_idx FOR TEXT"
```

The following XQuery expression, which specifies the default element as `http://posample.org`, returns all documents that have the matching XPath `/product/description/details` that contains the word `ergonomic`:

```
xquery declare default element namespace "http://posample.org";
db2-fn:xmlcolumn-contains('PRODUCT.DESCRPTION', '@xmlxp:
''/product/description/details [. contains ("ergonomic")]''')
```

Three documents are returned, two of which are expected because they have the namespace `http://posample.org` and one of which is unexpected because it has the namespace `http://posample.org/wheelshovel`.

The following XQuery expression uses the path expression `/product/..` to use the DB2 XQuery support for XML search and namespaces to filter the documents returned by DB2 Text Search engine so that only documents with the namespace `http://posample.org` are returned:

```
xquery declare default element namespace "http://posample.org";
db2-fn:xmlcolumn-contains('PRODUCT.DESCRPTION', '@xmlxp:
''/product/description/details [. contains ("ergonomic")]''')/product/..
```

**Note:** SQL queries can use DB2 XQuery to force namespace filtering. Given the previous example, the corresponding expression using an SQL query is as follows:

```
xquery declare default element namespace "http://posample.org";
db2-fn:sqlquery("select description from product where
contains(description, '@xmlxp:''/product/description/details
[. contains ("ergonomic")]''') = 1")
```

The workaround is as follows:

```
xquery declare default element namespace "http://posample.org";
db2-fn:sqlquery("select description from product where
contains(description, '@xmlxp:''/product/description/details
[. contains ("ergonomic")]''') = 1)/product/..
```

Similarly, to access a specific element in the document (as opposed to just having the matching document returned, as in the previous query), the following query can be used:

```
xquery declare default element namespace "http://posample.org";
db2-fn:xmlcolumn-contains('PRODUCT.DESCRPTION', '@xmlxp:
''/product/description/details [. contains ("ergonomic")]''')
/product/description[price > 20]/name
```

**Note:** This workaround is limited and might not work as expected if, for example, multiple product elements exist within a document.





---

## Chapter 4. Installing and configuring DB2 Text Search

DB2 Text Search is an optionally installable component whose installation and configuration are fully integrated with the installation of all DB2 database server products.

You can have the DB2 installer automatically install and configure DB2 Text Search. The steps that you must take are platform dependent. Figure 11 describes the installation and configuration process on Windows operating systems, and Figure 12 on page 42 describes the process on Linux and UNIX operating systems.

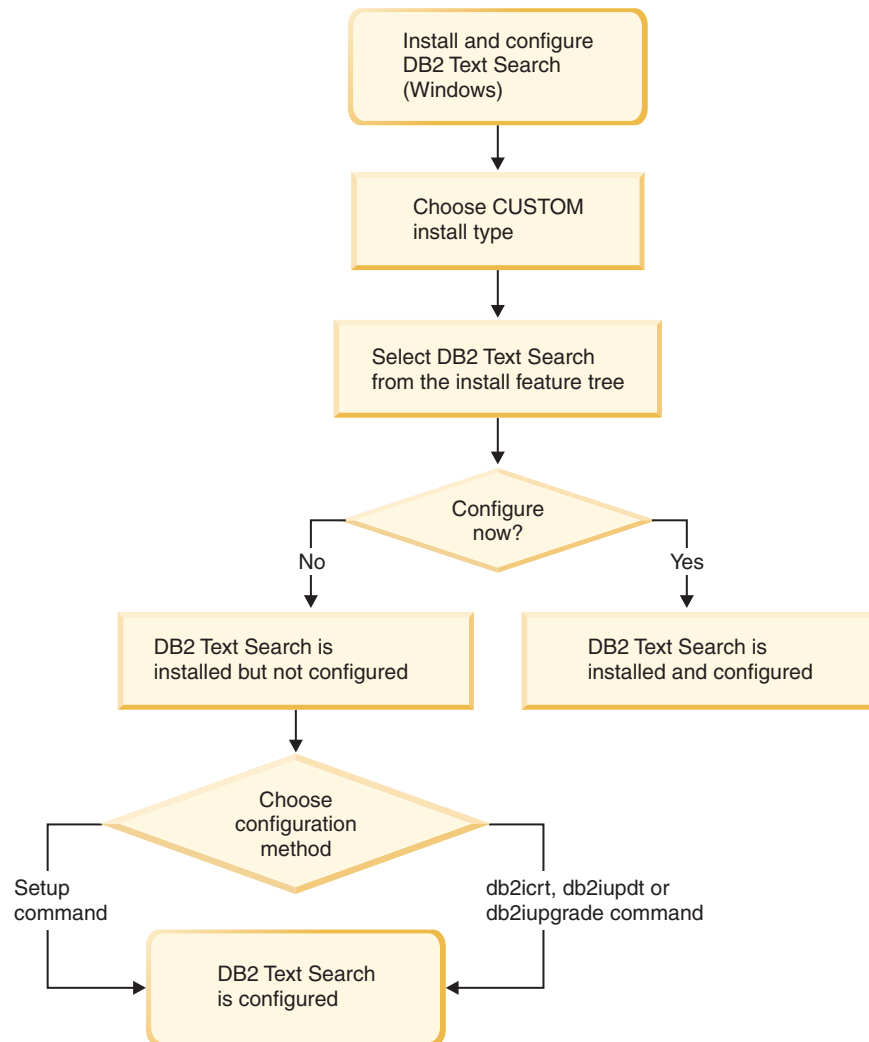


Figure 11. Installation and configuration on Windows platforms

On Windows, choose the installation type, decide whether to configure, and choose the configuration method.

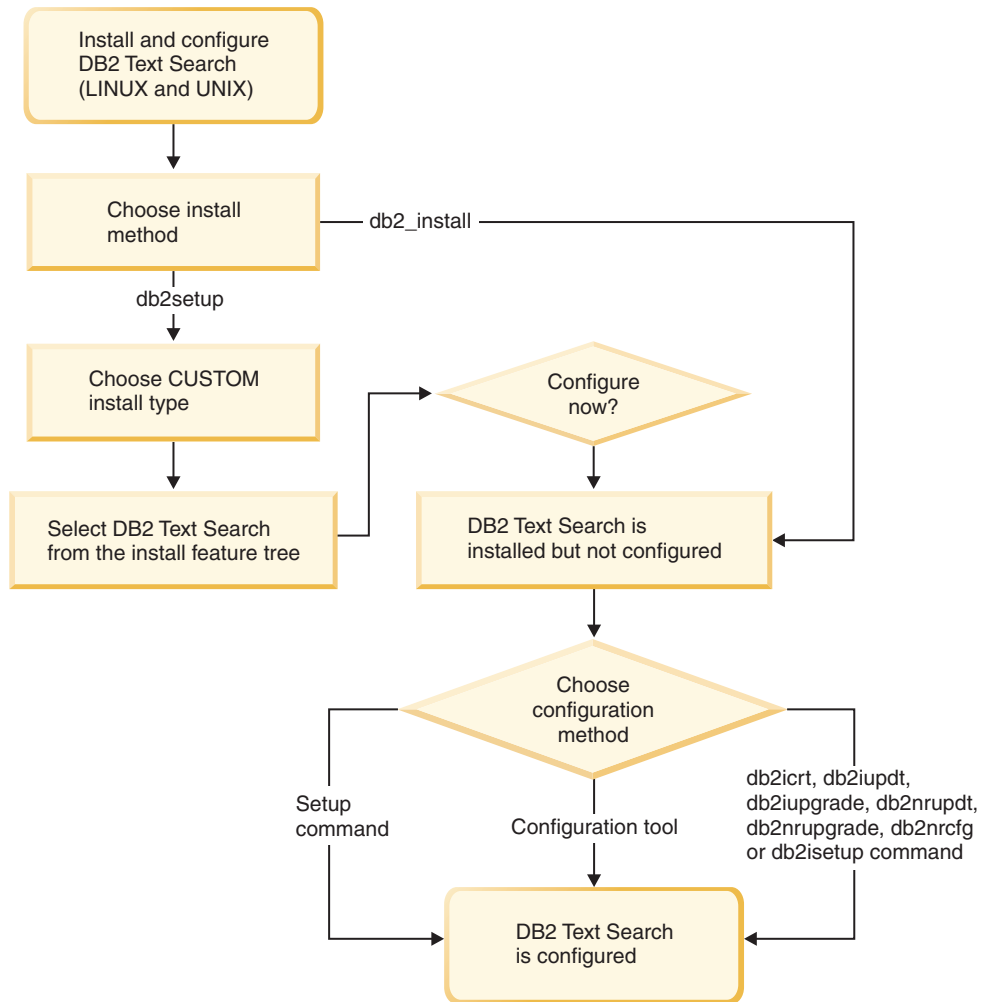


Figure 12. Installation and configuration on Linux and UNIX platforms

On Linux and UNIX, choose the installation method and type, decide whether to configure, and choose the configuration method. If you run **db2setup** as a non-root user, have your system administrator (who has SYSADM authority) run the **DB2RFE** command afterwards to reserve the port number that you want in the services file.

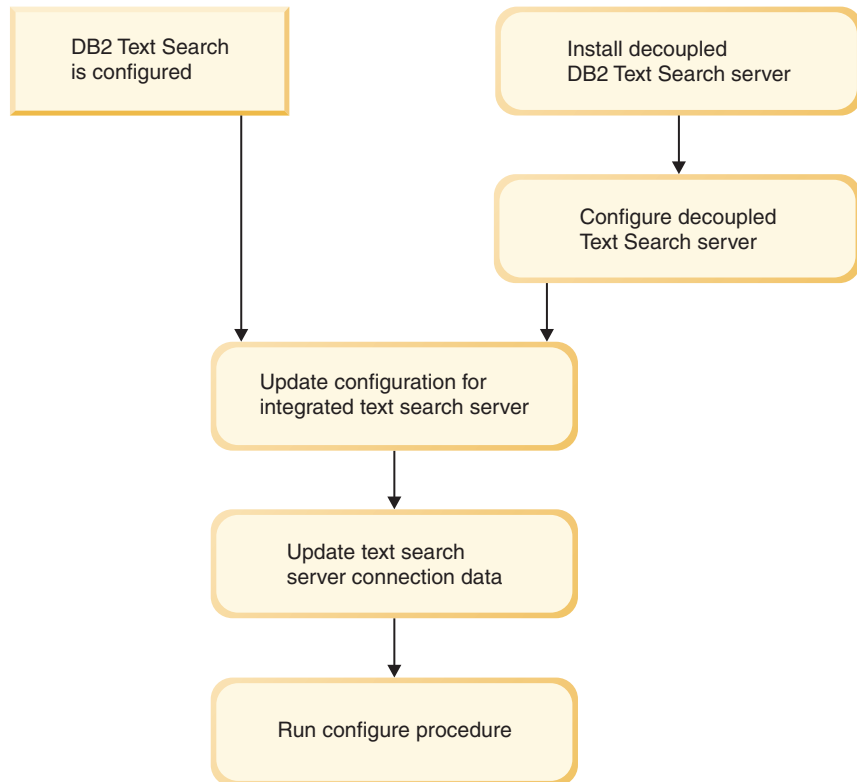


Figure 13. Configuration of a stand-alone DB2 Text Search server

For a stand-alone DB2 Text Search server, update the integrated text search server configuration. Then update the server connection data and run the **CONFIGURE** procedure.

DB2 Text Search has the following restrictions:

- You need to be on the coordinating member or instance owning partition when creating a database partitioned instance using the DB2 Setup Wizard.
- DB2 Text Search is not supported in a DB2 pureScale environment.

---

## Hardware and software requirements for DB2 Text Search

### Software platforms

DB2 Text Search is supported on the following operating systems platforms:

- AIX Version 6.1
- HP-UX 11i (Itanium-based HP Integrity Series platforms)
- Red Hat Enterprise Linux Server 5 (x86 and x64 platforms)
- Red Hat Enterprise Linux Server 6 (x86 and x64 platforms)
- Solaris 10 (UltraSPARC and x64 platforms)
- SUSE Linux Enterprise Server 10 (x86 and x64 platforms)
- SUSE Linux Enterprise Server 11 (x86 and x64 platforms)
- Windows Server 2003 (x86 and x64 platforms)
- Windows Server 2008 (x86 and x64 platforms)

**Important:** The libstdc++.so.5 shared library must be installed on Linux operating systems.

The stand-alone DB2 Text Search server is available for the previously listed platforms except for HP-UX 11i, and Solaris 10 x64 operating systems. Cross-platform usage is supported, a DB2 database instance on these platforms can be configured to use a stand-alone DB2 Text Search server on a supported platform.

## Hardware requirements

The minimum hardware requirements for DB2 Text Search are as follows:

*Table 2. Hardware requirements for DB2 Text Search*

DB2 Text Search Server	Processor	RAM / Memory	Disk
Integrated setup (In addition to DB2 database server requirements)	2 dual-core 2.66 GHz	4 GB	Including temporary working space, each text search index requires about four times the size of all documents that you want to index. For example, a text index on a column with 1 million rows of 1 KB text size needs about 4 GB of disk space.
Stand-alone setup			

Actual disk space, memory, and processor consumption depends on a various factors such as the number of collections, the number of documents per collection, the number of concurrently indexed collections, the required indexing throughput, and the query load. For more information, see the DB2 Text Search capacity planning topics.

For recommended operating system user process resource limits on Linux and UNIX operating systems, see the topic about operating system user limit requirements. These general resource limit requirements apply to both the integrated and stand-alone setups of the DB2 Text Search server.

---

## Installing DB2 Text Search with a default configuration

### Installing and configuring DB2 Text Search with the DB2 Setup Wizard

You can install DB2 Text Search with the DB2 Setup Wizard as a part of a custom installation of your DB2 database product.

#### About this task

Perform a custom installation of your DB2 database product and select DB2 Text Search from the feature tree. You can have DB2 Text Search automatically configured, or you can manually configure it later. You need to be on the coordinating member or instance owning partition if you are creating a partitioned instance using the DB2 Setup Wizard.

## Procedure

To perform a custom installation of DB2 Text Search using **setup** or **db2setup**:

1. Install the DB2 server using the instructions for your platform:
  - "Installing DB2 servers using the DB2 Setup wizard (Windows)" in *Installing DB2 Servers*
  - "Installing DB2 servers using the DB2 Setup wizard (Linux and UNIX)" in *Installing DB2 Servers*

You can select the DB2 Text Search component from the feature tree. During the installation, you have the option to configure DB2 Text Search for the default instance. If you do not want to configure DB2 Text Search, skip step 2.

2. To configure DB2 Text Search yourself, provide a valid service name and port number if these fields do not already have values. You do not have to configure DB2 Text Search immediately after installing it; you can configure it later. For instructions on how to perform the configuration later, see Chapter 5, "Configuring DB2 Text Search," on page 57.

## Installing and configuring DB2 Text Search with a response file

You can install and configure DB2 Text Search as a part of custom silent installation of your DB2 database product. This type of installations uses the **setup** or **db2setup** command with a response file.

### About this task

Perform a custom installation of your DB2 database product to install DB2 Text Search. You must add a number of keywords to your response file to have DB2 Text Search installed and configured.

## Procedure

To perform a custom installation:

1. Add the following line to the response file that you are using to install your DB2 database product:

```
COMP = TEXT_SEARCH
```
2. To configure DB2 Text Search during the installation, add the following lines to the response file:

- For root installations only:

```
db2inst_name.TEXT_SEARCH_HTTP_SERVICE_NAME = db2j_db2inst_name
```

where *db2inst\_name* is the name of the DB2 instance and *db2j\_db2inst\_name* is the service name.

- For root installations and non-root installations:

```
db2inst_name.TEXT_SEARCH_HTTP_PORT_NUMBER = port-number
```

If you provide a value for the **TEXT\_SEARCH\_HTTP\_SERVICE\_NAME** keyword for a non-root installation, an error will be returned.

You can specify any valid service name and port number that are not in use. If you do not provide any values, default values are used for configuration if the response file keyword *db2inst\_name*.**CONFIGURE\_TEXT\_SEARCH** is set to YES.

3. Install the DB2 database product using the instructions for your platform:

- "Installing a DB2 product using a response file (Windows)" in Installing DB2 Servers
- "Installing a DB2 product using a response file (Linux and UNIX)" in Installing DB2 Servers

## What to do next

You do not have to configure DB2 Text Search immediately after installing it; you can configure it later. For instructions on how to perform the configuration later, see Chapter 5, "Configuring DB2 Text Search," on page 57.

## Installing DB2 Text Search using `db2_install` (Linux and UNIX)

When you issue the `db2_install` command, you also install DB2 Text Search.

### About this task

**Important:** The command `db2_install` is deprecated and might be removed in a future release. Use the `db2setup` command with a response file instead.

To install DB2 Text Search, follow the steps outlined in "Install a DB2 product using `db2_install`" in *Installing DB2 Servers*. DB2 Text Search will automatically be installed as a part of the installation of your DB2 database product.

If this is a non-root installation, a DB2 instance is created and DB2 Text Search will be installed. If this a root installation, you must create a DB2 instance and configure DB2 Text Search using one of the available methods.

You do not have to configure DB2 Text Search immediately after you install it. For instructions on how to perform the configuration, see Chapter 5, "Configuring DB2 Text Search," on page 57.

---

## Installing DB2 Text Search without initial configuration

### Installing DB2 database servers using the DB2 Setup wizard (Windows)

This task describes how to start the DB2 Setup wizard on Windows. Use the DB2 Setup wizard to define your installation and install your DB2 database product on your system.

#### Before you begin

Before you start the DB2 Setup wizard:

- If you are planning on setting up a partitioned database environment, refer to "Setting up a partitioned database environment".
- Ensure that your system meets installation, memory, and disk requirements.
- If you are planning to use LDAP to register the DB2 server in Windows operating systems Active Directory, extend the directory schema before you install, otherwise you must manually register the node and catalog the databases. For more information, see the "Extending the Active Directory Schema for LDAP directory services (Windows)" topic.
- You must have a local Administrator user account with the recommended user rights to perform the installation. In DB2 database servers where LocalSystem

can be used as the DAS and DB2 instance user and you are not using the database partitioning feature, a non-administrator user with elevated privileges can perform the installation.

**Note:** If a non-Administrator user account is going to do the product installation, then the VS2010 runtime library must be installed before attempting to install a DB2 database product. The VS2010 runtime library is needed on the operating system before the DB2 database product can be installed. The VS2010 runtime library is available from the Microsoft runtime library download website. There are two choices: choose `vcredist_x86.exe` for 32-bit systems or `vcredist_x64.exe` for 64-bit systems.

- Although not mandatory, it is recommended that you close all programs so that the installation program can update any files on the computer without requiring a reboot.
- Installing DB2 products from a virtual drive or an unmapped network drive (such as `\\hostname\sharename` in Windows Explorer) is not supported. Before attempting to install DB2 products, you must map the network drive to a Windows drive letter (for example, Z:).

#### Restrictions

- You cannot have more than one instance of the DB2 Setup wizard running in any user account.
- The DB2 copy name and the instance name cannot start with a numeric value. The DB2 copy name is limited to 64 English characters consisting of the characters A-Z, a-z and 0-9.
- The DB2 copy name and the instance name must be unique among all DB2 copies.
- The use of XML features is restricted to a database that has only one database partition.
- No other DB2 database product can be installed in the same path if one of the following is already installed:
  - IBM® Data Server Runtime Client
  - IBM Data Server Driver Package
  - *DB2 Information Center*
- The DB2 Setup wizard fields do not accept non-English characters.
- If you enable extended security on Windows Vista or Windows 2008, or higher, users must belong to the DB2ADMNS or DB2USERS group to run local DB2 commands and applications because of an extra security feature (User Access Control) that limits the privileges that local administrators have by default. If users do not belong to one of these groups, they will not have read access to local DB2 configuration or application data.

#### Procedure

To start the DB2 Setup wizard:

1. Log on to the system with the local Administrator account that you have defined for the DB2 installation.
2. If you have the DB2 database product DVD, insert it into the drive. If enabled, the autorun feature automatically starts the DB2 Setup Launchpad. If the autorun does not work, use Windows Explorer to browse the DB2 database product DVD and double-click the **setup** icon to start the DB2 Setup Launchpad.

3. If you downloaded the DB2 database product from Passport Advantage<sup>®</sup>, run the executable file to extract the DB2 database product installation files. Use Windows Explorer to browse the DB2 installation files and double-click the **setup** icon to start the DB2 Setup Launchpad.
4. From the DB2 Setup launchpad, you can view installation prerequisites and the release notes, or you can proceed directly to the installation. You might want to review the installation prerequisites and release notes for late-breaking information.
5. Click **Install a Product** and the Install a Product window displays the products available for installation.  
 If you have no existing DB2 database products installed on your computer, launch the installation by clicking **Install New**. Proceed through the installation following the DB2 Setup wizard prompts.  
 If you have at least one existing DB2 database product installed on your computer, you can:
  - Click **Install New** to create a new DB2 copy.
  - Click **Work with Existing** to update an existing DB2 copy, to add function to an existing DB2 copy, upgrade an existing DB2 Version 9.5 and Version 9.7 copy, or to install an add-on product.
6. The DB2 Setup wizard determines the system language, and launch the setup program for that language. Online help is available to guide you through the remaining steps. To invoke the online help, click **Help** or press **F1**. You can click **Cancel** at any time to end the installation.
7. Sample panels when using the DB2 setup wizard lead you to the installation process. See the related links.

## Results

Your DB2 database product is installed, by default, in the *Program\_Files\IBM\sqllib* directory, where *Program\_Files* represents the location of the Program Files directory.

If you are installing on a system where this directory is already being used, the DB2 database product installation path has *\_xx* added to it, where *xx* are digits, starting at 01 and increasing depending on how many DB2 copies you have installed.

You can also specify your own DB2 database product installation path.

## What to do next

- Verify your installation.
- Perform the necessary post-installation tasks.

For information about errors encountered during installation, review the installation log file located in the *My Documents\DB2LOG\* directory. The log file uses the following format: *DB2-ProductAbbrev-DateTime.log*, for example, *DB2-ESE-Tue Apr 04 17\_04\_45 2012.log*.

If this is a new DB2 product installation on Vista 64-bit, and you use a 32-bit OLE DB provider, you must manually register the IBMDADB2 DLL. To register this DLL, run the following command:

```
c:\windows\SysWOW64\regsvr32 /s c:\Program_Files\IBM\SQLLIB\bin\ibmdadb2.dll
```



where *Program\_Files* represents the location of the Program Files directory.

If you want your DB2 database product to have access to DB2 documentation either on your local computer or on another computer on your network, then you must install the *DB2 Information Center*. The *DB2 Information Center* contains documentation for the DB2 database system and DB2 related products. By default, DB2 information is accessed from the web if the *DB2 Information Center* is not locally installed.

IBM Data Studio can be installed by running the the DB2 Setup wizard

### **DB2 Express® Edition and DB2 Workgroup Server Edition memory limits**

If you are installing DB2 Express Edition, the maximum allowed memory for the instance is 4 GB.

If you are installing DB2 Workgroup Server Edition, the maximum allowed memory for the instance is 64 GB.

The amount of memory allocated to the instance is determined by the **INSTANCE\_MEMORY** database manager configuration parameter.

### **Important notes when upgrading from Versions 9.5 or 9.7:**

- The self tuning memory manager does not increase your overall instance memory limit beyond the license limits.

## **Installing DB2 servers using the DB2 Setup wizard (Linux and UNIX)**

This task describes how to start the DB2 Setup wizard on Linux and UNIX operating systems. The DB2 Setup wizard is used to define your installation preferences and to install your DB2 database product on your system.

### **Before you begin**

Before you start the DB2 Setup wizard:

- If you are planning on setting up a partitioned database environment, refer to “Setting up a partitioned database environment” in *Installing DB2 Servers*
- Ensure that your system meets installation, memory, and disk requirements.
- Ensure you have a supported browser installed.
- You can install a DB2 database server using either root or non-root authority. For more information about non-root installation, see “Non-root installation overview (Linux and UNIX)” in *Installing DB2 Servers*.
- The DB2 database product image must be available. You can obtain a DB2 installation image either by purchasing a physical DB2 database product DVD, or by downloading an installation image from Passport Advantage.
- If you are installing a non-English version of a DB2 database product, you must have the appropriate National Language Packages.
- The DB2 Setup wizard is a graphical installer. You must have X windows software capable of rendering a graphical user interface for the DB2 Setup wizard to run on your machine. Ensure that the X windows server is running. Ensure that you have properly exported your display. For example, export `DISPLAY=9.26.163.144:0`.
- If you are using security software in your environment, you must manually create required DB2 users before you start the DB2 Setup wizard.

Restrictions

- You cannot have more than one instance of the DB2 Setup wizard running in any user account.
- The use of XML features is restricted to a database that is defined with the code set UTF-8 and has only one database partition.
- The DB2 Setup wizard fields do not accept non-English characters.
- For HP-UX 11i V2 on Itanium based HP Integrity Series Systems, users created with Setup Wizard for DB2 instance owner, fenced user, or DAS cannot be accessed with the password specified on DB2 Setup Wizard. After the setup wizard is finished, you need to reset the password of those users. This does not affect the instance or DAS creation with the setup wizard, therefore, you do not need to re-create the instance or DAS.

## Procedure

To start the DB2 Setup wizard:

1. If you have a physical DB2 database product DVD, change to the directory where the DB2 database product DVD is mounted by entering the following command:

```
cd /dvdrom
```

where */dvdrom* represents the mount point of the DB2 database product DVD.

2. If you downloaded the DB2 database product image, you must extract and untar the product file.

- a. Extract the product file:

```
gzip -d product.tar.gz
```

where *product* is the name of the product that you downloaded.

- b. Untar the product file:

### On Linux operating systems

```
tar -xvf product.tar
```

### On AIX, HP-UX, and Solaris operating systems

```
guntar -xvf product.tar
```

where *product* is the name of the product that you downloaded.

- c. Change directory:

```
cd ./product
```

where *product* is the name of the product that you downloaded.

**Note:** If you downloaded a National Language Package, untar it into the same directory. This will create the subdirectories (for example *./nlpack*) in the same directory, and allows the installer to automatically find the installation images without prompting.

3. Enter the **./db2setup** command from the directory where the database product image resides to start the DB2 Setup wizard.
4. The IBM DB2 Setup Launchpad opens. From this window, you can view installation prerequisites and the release notes, or you can proceed directly to the installation. You can also review the installation prerequisites and release notes for late-breaking information.
5. Click **Install a Product** and the **Install a Product** window will display the products available for installation.

Launch the installation by clicking **Install New**. Proceed through the installation following the DB2 Setup wizard's prompts.

6. Sample panels when using the DB2 setup wizard will lead you to the installation process. See the related links.

After you have initiated the installation, proceed through the DB2 Setup wizard installation panels and make your selections. Installation help is available to guide you through the remaining steps. To invoke the installation help, click **Help** or press F1. You can click **Cancel** at any time to end the installation.

## Results

For non-root installations, DB2 database products are always installed in the `$HOME/sql1lib` directory, where `$HOME` represents the non-root user's home directory.

For root installations, DB2 database products are installed, by default, in one of the following directories:

### AIX, HP-UX, and Solaris

`/opt/IBM/db2/V10.1`

**Linux** `/opt/ibm/db2/V10.1`

If you are installing on a system where this directory is already being used, the DB2 database product installation path will have `_xx` added to it, where `_xx` are digits, starting at 01 and increasing depending on how many DB2 copies you have installed.

You can also specify your own DB2 database product installation path.

DB2 installation paths have the following rules:

- Can include lowercase letters (a-z), uppercase letters (A-Z), and the underscore character ( \_ )
- Cannot exceed 128 characters
- Cannot contain spaces
- Cannot contain non-English characters

The installation log files are:

- The DB2 setup log file. This file captures all DB2 installation information including errors.
  - For root installations, the DB2 setup log file name is `db2setup.log`.
  - For non-root installations, the DB2 setup log file name is `db2setup_username.log`, where `username` is the non-root user ID under which the installation was performed.
- The DB2 error log file. This file captures any error output that is returned by Java (for example, exceptions and trap information).
  - For root installations, the DB2 error log file name is `db2setup.err`.
  - For non-root installations, the DB2 error log file name is `db2setup_username.err`, where `username` is the non-root user ID under which the installation was performed.

By default, these log files are located in the `/tmp` directory. You can specify the location of the log files.

There is no longer a `db2setup.his` file. Instead, the DB2 installer saves a copy of the DB2 setup log file in the `DB2_DIR/install/logs/` directory, and renames it `db2install.history`. If the name already exists, then the DB2 installer renames it `db2install.history.xxxx`, where `xxxx` is 0000-9999, depending on the number of installations you have on that machine.

Each installation copy has a separate list of history files. If an installation copy is removed, the history files under this install path will be removed as well. This copying action is done near the end of the installation and if the program is stopped or aborted before completion, then the history file will not be created.

### What to do next

- Verify your installation.
- Perform the necessary post-installation tasks.

IBM Data Studio can be installed by running the the DB2 Setup wizard

National Language Packs can also be installed by running the `./db2setup` command from the directory where the National Language Pack resides, after a DB2 database product has been installed.

On Linux x86, if you want your DB2 database product to have access to DB2 documentation either on your local computer or on another computer on your network, then you must install the *DB2 Information Center*. The *DB2 Information Center* contains documentation for the DB2 database system and DB2 related products.

### DB2 Express Edition and DB2 Workgroup Server Edition memory limits

If you are installing DB2 Express Edition, the maximum allowed memory for the instance is 4 GB.

If you are installing DB2 Workgroup Server Edition, the maximum allowed memory for the instance is 64 GB.

The amount of memory allocated to the instance is determined by the `INSTANCE_MEMORY` database manager configuration parameter.

### Important notes when upgrading from Versions 9.5 or 9.7:

- If the memory configuration for your Versions 9.5 or 9.7 DB2 database product exceeds the allowed limit, the DB2 database product might not start after upgrading to the current version.
- The self tuning memory manager will not increase your overall instance memory limit beyond the license limits.

## Response file installation of DB2 overview (Windows)

On Windows, you can perform a response file installation of a DB2 product on a single machine or on multiple machines. A response file installation might also be referred to as a silent installation or an unattended installation.

### Before you begin

Before you begin the installation, ensure that:

- Your system meets all of the memory, hardware, and software requirements to install your DB2 product.
- You have all of the required user accounts to perform the installation.
- Ensure all DB2 processes are stopped.

## Procedure

- To perform a response file installation of a DB2 product on a single machine:
  1. Create and customize a response file by one of the following methods:
    - Modifying a sample response file. Sample response files are located in (db2\Windows\samples).
    - Using the DB2 Setup wizard to generate a response file.
    - Using the response file generator.
  2. Run the **setup -u** command specifying your customized response file. For example, a response file created during an installation:

```
setup -u my.rsp
```
- To perform a response file installation of a DB2 product on multiple machines:
  1. Set up shared access to a directory.
  2. Create a response file using the sample response file.
  3. Install a DB2 product using a response file.

## Response file installation of DB2 overview (Linux and UNIX)

This task describes how to perform response file installations on Linux or UNIX. You can use the response file to install additional components or products after an initial installation. A response file installation might also be referred to as a silent installation or an unattended installation.

### Before you begin

Before you begin the installation, ensure that:

- Your system meets all of the memory, hardware, and software requirements to install your DB2 database product.
- All DB2 processes are stopped. If you are installing a DB2 database product on top of an existing DB2 installation on the computer, you must stop all DB2 applications, the DB2 database manager, and DB2 processes for all DB2 instances and DB2 DAS related to the existing DB2 installation.

### Restrictions

Be aware of the following limitations when using the response files method to install DB2 on Linux or UNIX operating systems:

- If you set any instance or global profile registry keywords to BLANK (the word "BLANK"), that keyword is, in effect, deleted from the list of currently set keywords.
- Ensure that you have sufficient disk space before installing. Otherwise, if the installation fails, manual cleanup is required.
- If you are performing multiple installations or are installing DB2 database products from multiple DVDs, it is recommended that you install from a network file system rather than a DVD drive. Installing from a network file system significantly decreases the amount of time it takes to perform the installation.
- If you are planning on installing multiple clients, set up a mounted file system on a code server to improve performance.

## Procedure

To perform a response file installation:

1. Mount your DB2 database product DVD or access the file system where the installation image is stored.
2. Create a response file by using the sample response file.  
Response files have a file type of .rsp. For example, ese.rsp.
3. Install DB2 using the response file.

---

## Installing and configuring a stand-alone Text search server

### Installation space requirements for the stand-alone server

The stand-alone text search server installation requires at least 1 GB of disk space.

A small amount of disk space is needed in addition for configuration data for each collection, however, significant space is needed for the index data. For details, see the topic about disk consumption for DB2 Text Search.

The location of index data files can be configured using the default data directory or specified as collection directory parameter when creating a text search index.

### Installing a stand-alone DB2 Text Search server

You can install a stand-alone DB2 Text Search server silently. The silent installation option takes input values from a response file. You can install one or more servers for a stand-alone setup. The stand-alone text search server is also known as ECM Text Search server.

#### Procedure

To install a stand-alone text search server:

1. Create an empty installation directory.
  - For example, on Linux or UNIX systems, create the following directory:  
`/opt/ibm/ECMTextSearch`
  - For example, on Windows systems, create the following directory:  
`C:\Program Files\IBM\ECMTextSearch`  
This directory is known as `<ECMTS_HOME>`.
2. Download the DB2 Accessories Suite for your platform from the IBM DB2 Accessories Suite for Linux, UNIX, and Windows website. Extract it to a temporary directory.
3. Log in as user with the required authorities or permissions.
  - On Linux and UNIX systems, you need read, write, and execute permission for the installation directory.
  - On Windows systems, you need administrator authority
4. Review the license and edit the `ecmts_response.txt` file to customize your settings.
5. Use the setup file `ecmts15_install_<platform>.exe` to install the stand-alone Text Search server. Start the installation by issuing the following command:  
`./<ecmts_setup_file_name> -i silent -f ecmts_response.txt`

For example, on Windows systems, issue the following command:

```
ecmts15_install_win32.exe -i silent -f ecmts_response.txt
```

6. Verify that the installation was successful.

Check the installation log file and the folders that were created in the <ECMTS\_HOME> directory. You should see at least bin, lib, config and resource folders.

7. Start the server by running the startup script from the <ECMTS\_HOME> directory.
  - On Linux and UNIX platforms:  
bin/startup.sh
  - On Windows platforms:  
bin\startup
8. Configure and customize the Text Search server properties. For details, see the topic about configuring a stand-alone DB2 Text Search server.

## Installing and configuring stand-alone server as a Windows service

You can install and configure stand-alone text search server processes as a Microsoft Windows service.

### About this task

The stand-alone server service runs under the local system account and the startup type is set to automatic. You can specify a name for the service and specify whether the service starts automatically after installation.

### Procedure

To install and run stand-alone server as a Windows service:

1. Open the ecmts\_response.txt response file and set the following parameters:
  - IA\_INSTALL\_AS\_WINDOWS\_SERVICE  
Set the value of this parameter to YES.
  - IA\_WINDOWS\_SERVICE\_NAME  
Specify a unique name for the stand-alone DB2 Text Search Windows service. This is an optional parameter.  
When the value of this parameter is not specified or set to AUTO, a default name ECM Text Search Server is assigned to the Windows service. If the service already exists and its name was not specified, a numeric suffix is added to the name, for example ECM Text Search Server1. If you specify a name for the service and a service with the same name already exists, an error is returned.
  - IA\_START\_SERVER  
To automatically start the DB2 Text Search Windows service after installation, ensure that the IA\_START\_SERVER parameter is set to YES. This is an optional parameter. The default setting is YES.
2. Install the stand-alone text search server. From the directory that contains the setup and response files, run the appropriate setup file for your operating system.
3. You can start and stop the Text Search Windows service by using the Microsoft Services window. To access the Services window, open the Windows Control Panel and click Administrative Tools > Services.

## Uninstalling a stand-alone DB2 Text Search server

You can uninstall a stand-alone DB2 Text Search by using the Uninstall\_ECMTTextSearch command.



## Before you begin

Stop any DB2 Text Search services and shutdown the stand-alone DB2 Text Search server before starting the uninstallation process.

## Procedure

To uninstall the stand-alone DB2 Text Search server:

1. Navigate to the *ECMTS\_HOME* directory.
2. Start the uninstallation by issuing one of the following platform-specific commands:

- On Linux and UNIX operating systems:

```
INSTALL_DIR/Uninstall_ECMTTextSearch/Uninstall_ECMTTextSearch -i silent
```

- On Windows operating systems:

```
ECMTS_HOME\Uninstall_ECMTTextSearch\Uninstall_ECMTTextSearch.exe -i silent
```

The uninstall program does not remove all data from the *ECMTS\_HOME* directory. For example, the *uninstall.log* file remains after running the uninstall program. Some or all of the following directories might not be removed by the uninstall program and must be removed manually:

- *ECMTS\_HOME*\config
- *ECMTS\_HOME*\license
- *ECMTS\_HOME*\log
- *ECMTS\_HOME*\resource
- *ECMTS\_HOME*\temp
- *ECMTS\_HOME*\Uninstall\_ECMTTextSearch

**Tip:** You might want to back up collection or configuration data that is stored in the *ECMTS\_HOME*\config directory for future use.

## Results

The DB2 Text Search server is uninstalled and cannot be used anymore for text search index administration or full-text query execution. However, the text index collection and configuration data remains intact.



---

## Chapter 5. Configuring DB2 Text Search

Your options for configuring DB2 Text Search depend on whether you are performing the initial configuration or a reconfiguration and which platform you are using.

### Before you begin

Before reconfiguration of the DB2 Text Search, stop the text search instance service, as outlined in “Stopping the DB2 Text Search instance service” on page 75.

For partitioned instances you need to be on the coordinating member or instance owning partition when using the configuration tool. This is the instance host where the integrated text search server is initially configured and is the lowest numbered partition server host.

### Procedure

- Determine whether DB2 Text Search is configured.

Run the configuration tool by issuing the following command:

```
configTool printAll -configPath absolute-path-to-configuration-folder
```

In the output of the **printAll** option, the authentication token is an empty string if DB2 Text Search is not configured.

- Configure DB2 Text Search for the first time.

On Linux and UNIX operating systems, use one of the following methods to configure DB2 Text Search:

- Rerun the silent installation as described in “Installing and configuring DB2 Text Search with a response file” on page 45.
- Rerun the GUI installation as described in “Installing and configuring DB2 Text Search with the DB2 Setup Wizard” on page 44.
- Use the configuration tool. Refer to “Initial configuration of an integrated DB2 Text Search server” on page 59. Note that using the configuration tool to perform a manual configuration requires you to manually configure most of the parameters, whereas using the installer requires you to configure only two parameters.
- Use one of the following commands to configure DB2 Text Search, depending on the instance type and operation:
  - For root installs, you can issue **db2isetup** command in the GUI to configure existing DB2 instance by selecting DB2 Text Search when it is being configured. You also can issue the **db2iupdt** command with **-j** option to configure integrated DB2 Text Search server. Note that when you create an instance using the **db2icrt** command with **-j** option, DB2 Text Search is also configured by default.
  - For non-root installs, issue the **db2isetup** command to configure the instance in the GUI, or issue the **db2nrupdt** or **db2nrupgrade** command with the **-j** option.

On Windows operating systems, use one of the following methods to configure DB2 Text Search:

- Rerun the silent installation as described in “Installing and configuring DB2 Text Search with a response file” on page 45.

- Rerun the GUI installation as described in “Installing and configuring DB2 Text Search with the DB2 Setup Wizard” on page 44.
- Issue the **db2iupdt** command with the -j option. Note that when you create an instance using **db2icrt** command with the -j option, DB2 Text Search is also configured by default.

- Determine whether the Java developer kit is from IBM.

The DB2 Text Search internally uses a Java developer kit whose location is pointed by **JDK\_PATH** of db2 get dbm cfg command and this Java developer kit has to come from IBM. To verify if the Java developer kit is from IBM, run the following command:

```
JDK_PATH/jre/bin/java -version
```

This command will display the Java version information and IBM should display as part of string if the Java developer kit is from IBM.

- Re-configure DB2 Text Search.

After you have configured DB2 Text Search, you cannot use the GUI installer to re-configure it. You must make any updates to the configuration manually.

On Linux and UNIX operating systems, use one of the following methods to re-configure DB2 Text Search:

- Rerun the silent installation as described in “Installing and configuring DB2 Text Search with a response file” on page 45.
- Use the Configuration Tool. Refer to “Initial configuration of an integrated DB2 Text Search server” on page 59.
- Use one of the following commands to re-configure DB2 Text Search, depending on the instance type and operation:
  - For root installs, you can issue **db2isetup** command in the GUI to configure an existing DB2 instance by selecting the DB2 Text Search instance being configured. You also can issue the **db2iupdt** command with -j option to configure integrated DB2 Text Search server.
  - For non-root installs, issue the **db2isetup** command to configure the instance in the GUI, or issue the **db2nrupdt** or **db2nrupgrade** command with the -j option.

On Windows operating systems, use one of the following methods to re-configure DB2 Text Search:

- Rerun the silent installation as described in “Installing and configuring DB2 Text Search with a response file” on page 45.
- Use the Configuration Tool. Refer to “Initial configuration of an integrated DB2 Text Search server” on page 59.
- Run the **db2iupdt**, or **db2iupgrade** command, specifying the -j option as shown to meet your needs:
  - -j "TEXT\_SEARCH" attempts to configure DB2 Text Search with the default service name and a generated port value.
  - -j "TEXT\_SEARCH, [servicename]" reserves the service name with an automatically generated port number or with the same port number assigned to that service name if the service name is already reserved in the services file.
  - -j "TEXT\_SEARCH, [port number]" reserves the port with the default service name.
  - -j "TEXT\_SEARCH, [servicename], [port#]" reserves the specified service name and port number.

**Note:** On Windows operating systems, the **PATH** in the DB2 command window points to *current-default-copy-install-path\db2tss\bin*, so to configure an instance that is not in the current DB2 copy, first switch to the appropriate DB2 command window for that copy.

---

## Initial configuration of an integrated DB2 Text Search server

The Configuration Tool is a command-line tool that you can use to perform the initial configuration of DB2 Text Search or to change the current configuration.

### Before you begin

To customize most of the configuration settings, you must stop the DB2 Text Search instance services.

### About this task

The most convenient method for the initial configuration after installation is to use the DB2 installer. For a manual initial configuration as well as any configuration updates, you must use the configuration tool.

### Procedure

To perform the initial configuration of the DB2 Text Search server use the following steps. See the topic about the Configuration Tool for further details.

1. Run the **configTool** command with the **configureParams** option to set the configuration path.

- Review the following configuration options and change the defaults as needed:

**-defaultDataDirectory:** location of the text index collections, each collection will be stored in its own subdirectory.

**-logPath:** location of Text Search server log and trace files.

**-tempDirPath:** path to the temporary directory.

**-installPath:** path to DB2 Text Search install directory which is `DB2PATH\db2tss` on Windows and the `DB2DIR/db2tss` directory on Linux and UNIX, where `DB2DIR` is the location of the DB2 copy.

**-startupHeapSize:** maximum heap size of the text search server .

For example, to configure the **defaultDataDirectory** and **installPath** options, issue the following command:

```
configTool configureParams -configPath <absolute-path-to-config-folder>
-defaultDataDirectory dataPath -installPath ipath
```

- On Windows operating systems, specify the command as shown. You need to specify only **configPath**; all of the other parameters are assigned default paths and values.

```
configTool
-configPath absolute-path-to-config-folder
```

2. DB2 Text Search authenticates text search index administration and text search requests by using an authentication token. Generate the authentication token by issuing the **configTool** command with the **generateToken** parameter, as follows:

```
configTool generateToken
-configPath absolute-path-to-config-folder
-seed myInstanceName
```

3. Specify the HTTP port by issuing the **configTool** command with the **configureHTTPListener** parameter, as follows:

```
configTool configureHTTPListener
-configPath absolute-path-to-config-folder
-adminHTTPPort port-number
```

**Note:** The value of the port should be between 1024 and 65535.

The administrative HTTP port allows communication between text search processes using TCP/IP. During the installation of a DB2 database product or during instance creation, you can specify a service name and port if you have root authority. These are used for updating the services file.

4. Update the services file.

Refer to “Updating the services file on the server for TCP/IP communications” on page 63.

When you use the Configuration Tool for configuration, the tool does not update the services file. Therefore, you must update the services file manually,

**Note:** Only root users can update the services file. Non-root users must have the system administrator run the **db2rfe** command first.

---

## Updating DB2 Text Search server information

DB2 Text Search server information is used in the database to connect to the Text Search server to administer and search in text search indexes. Valid settings are therefore required to ensure proper functioning of the system and must be defined in the text search catalog SYSIBMTS.TSSERVERS administrative view.

### Before you begin

Updating text search server information requires the SYSTS\_ADM role and DBADM privileges on the specified database.

### About this task

The server information consists mainly of connection information, like the server host name, the server token value and the server port number, and server characteristics, like server locale, whether the text search setup is enabled for rich text support, and an indication whether the search server utilized by the DB2 instance is integrated (configured by DB2 as part of the DB2 instance) or a separate stand-alone installation of the text search server.

The update is required initially for the following scenarios:

- an incomplete enablement warning message is encountered when enabling the database for text search.
- initial configuration of a stand-alone text search server
- partitioned databases
- DB2 Text Search upgrades
- stored procedures are used for administration from a client machine
- and further on, following any updates to text search server connection information.

During database enablement the SYSIBMTS.TSSERVERS administrative view is updated with initial connection information for the integrated server, if the necessary authorization to access the configuration is available. Review and update

the text server information in SYSIBMTS.TSSERVERS with the relevant text search server data and run the SYSTS\_CONFIGURE procedure to apply the updated information. For multiple databases in the instance, configure each database with the information for the same text search server.

When re-configuration is needed, ensure that no text search administrative operation is active and shut down the text search server before applying any changes.

Certain aspects relating to the text search installation and DB2 instance configuration for text search have to be updated. They include:

- An indication whether the search server utilized by the DB2 instance is integrated (configured by DB2 as part of the DB2 instance), or if it is a separate stand-alone installation of the text search server.
- An indication if the text search setup is enabled for rich text support.

## Procedure

To updating DB2 Text Search server information:

1. Get the needed text search server property values, such as host name, token, and port number, by issuing the **configTool** command with the `printAll` option. For more details, see the topic about `configTool`.
  2. Review the entries in the SYSIBMTS.TSSERVERS administrative view and make any necessary update:
    - If the view is empty then use an INSERT statement. For example:

```
INSERT INTO SYSIBMTS.TSSERVERS (HOST, PORT, TOKEN, SERVERSTATUS)
VALUES ('localhost', 55000, '9kfsjg48=', 0);
```
    - If the view already contains a row then use a UPDATE statement. For example:

```
UPDATE SYSIBMTS.TSSERVERS SET (HOST, PORT, TOKEN)
= ('tsmach1.ibm.com', 55002, 'k3j4fjk9u=')
```
- Make sure to use the actual hostname or IP address instead of localhost if multiple database partitions are used, or administrative operations are executed from a client. This applies not only to local installs of a stand-alone text search server, but also to integrated servers.
3. Execute the **SYSTS\_CONFIGURE** procedure. For more details, see the topic about the SYSTS\_CONFIGURE procedure.
  4. Verify the values in the SYSIBMTS.TSSERVERS administrative view are those returned by configuration tool.
  5. Start the text search service to verify that the text search server can be contacted.

---

## Configuring a stand-alone DB2 Text Search server

Use the configuration tool to customize some default properties after installing the stand-alone DB2 Text Search server. You can configure the relevant system level properties and the security properties for your system.

Before configuring the properties, ensure that the stand-alone DB2 Text Search server is shut down and that the text search services are stopped. Do not restart the text search server until you finish both the configuration of the stand-alone text search server and complete required configuration updates of the enabled databases in the associated DB2 instance.

You can use the configuration tool to view text search server properties even when the text search server is stopped.

## System configuration

Make sure to review and configure at minimum the following properties with the configuration tool:

- **configureHTTPListener**: Configures the DB2 Text Search server port and host name
- **generateToken**: Generates the authentication token and encryption key
- **defaultDataDirectory**: Configures the parameters for the collection

**Remember:** If the value for *configPath* contains blanks, you must enclose the value in quotation marks.

For details, and additional optional configuration see the topic about the configuration tool for DB2 Text Search.

## Security configuration

Every API request from a DB2 database server to a stand-alone DB2 Text Search server is authenticated by an authentication token. An initial token is generated during the installation of the stand-alone text search server.

1. Use the configuration tool to explicitly provide a seed value and generate the authentication token. The maximum length of the token string is 32 bytes.
2. Run the configuration tool on the DB2 instance to set the matching token value.
3. Store the connection information including the token in the SYSIBMTS.TSSERVER administrative view for each enabled database.

You can use the DB2 Text Search configuration tool to show the current authentication token and encryption key values. However, it is impossible to determine the seed value used by the stand-alone DB2 Text Search server. Generate the token explicitly with the **configTool** utility and update the master configuration on the DB2 instance to match the configured values for the token.

To configure the properties for the text search server run the configuration tool by entering the appropriate platform-specific command:

- On Linux and UNIX platforms:

```
configTool.sh configuration_command -configPath value
[-locale value] -command_specific_arguments
```
- On Windows platforms:

```
configTool.bat configuration_command -configPath value
[-locale value] -command_specific_arguments
```

For example, to print the current authentication token on a Linux server, use the following command:

```
configTool.sh printToken -configPath /opt/ibm/ECMTextSearch/config
```

**Note:** For a stand-alone DB2 Text Search server on Linux and UNIX platforms, the configuration tool command must be specified in full including the .sh suffix. Only the integrated DB2 Text Search server supports the script names without the suffix.

---

## Updating the services file on the server for TCP/IP communications

This task is part of the main task of *Configuring TCP/IP communications for a DB2 instance*.

### About this task

The TCP/IP services file specifies the ports that server applications can listen on for client requests. If you specified a service name in the **svcname** field of the DBM configuration file, the services file must be updated with the service name to port number/protocol mapping. If you specified a port number in the **svcname** field of the DBM configuration file, the services file does *not* need to be updated.

Update the services file and specify the ports that you want the server to listen on for incoming client requests. The default location of the services file depends on the operating system:

#### Linux and UNIX operating systems

`/etc/services`

#### Windows operating systems

`%SystemRoot%\system32\drivers\etc\services`

### Procedure

Using a text editor, add the Connection entry to the services file. For example:

```
db2c_db2inst1 3700/tcp # DB2 connection service port
```

where:

#### **db2c\_db2inst1**

represents the connection service name

**3700** represents the connection port number

**tcp** represents the communication protocol that you are using

---

## Installing DB2 Accessories Suite for DB2 Text Search

DB2 Accessories Suite enables indexing and search for documents with rich text and proprietary formats with DB2 Text Search. You can start a new install or run the install on top of an existing installation.

### Before you begin

To install DB2 Accessories Suite on Linux and UNIX, you need to be logged on to the DB2 server as a system administrator. On Windows, you must log on as a user with Local Administrator authority.

Download DB2 Accessories Suite. For the download link, see: <https://www.ibm.com/services/forms/preLogin.do?source=swg-dm-db2accsuite>. Install the most up-to-date version of the DB2 Accessories Suite release or fix pack to ensure proper functioning of the feature.

Ensure the installer file, the license file, and the release info file are in the same directory.



## Procedure

To install DB2 Accessories Suite:

1. Stop the DB2 Text Search instance service. To stop the service, issue the **db2ts STOP FOR TEXT** command.
2. Log on to the DB2 database server as a user with the necessary permissions which have writing privilege in DB2 Text Search installation directory, for example, on Linux platform, the directory locates under *<DB2PATH>/db2tss* directory, where *<DB2PATH>* represents the DB2 database server installation directory
3. There are two installation modes. One option is console installation, while the other is silent installation.
  - To complete a console install:
    - a. Run the accessories suite filter installer.
      - Run the installer **installAccSuiteV10.bin** from the command line for Linux and UNIX platforms.
      - There are two approaches on the Windows platform.
        - Run the installer **installAccSuiteV10.exe** from the command window
        - Double click the installer binary file.
    - b. After accepting the license, enter the location of the db2tss subdirectory in the latest DB2 copy when prompted for the install path.
    - c. The db2tss directory must already exist. If it is missing, DB2 Text Search has not been properly installed and configured.
    - d. Review the summary and confirm the installation.
  - To complete a silent install:
    - a. Modify the response file by setting the **LICENSE\_ACCEPTED** parameter as true and assigning the correct install full path **USER\_INSTALL\_DIR** which should contain the db2tss directory.
    - b. Run the accessories suite filter installer with silent model.
      - Run the **installAccSuiteV10.bin -i silent -f installer.properties** command from the command line on Linux and UNIX platforms.
      - Run the **installAccSuiteV10.exe -i silent -f installer.properties** command from the command window on the Windows platform.

## Results

You have successfully installed DB2 Accessories Suite.

## What to do next

You can now enable rich text document support for DB2 Text Search. See, “Enabling DB2 Text Search for rich text document support” on page 74 for more details.

---

## Uninstalling the DB2 Accessories Suite for DB2 Text Search

You can uninstall a stand-alone DB2<sup>®</sup> Text Search by using the **Uninstall\_DB2AS** command.



## Before you begin

In order to uninstall DB2 Accessories Suite on Linux and UNIX platforms, you must be logged on to the DB2 database server as a system administrator. On Windows platforms you must be logged on as a user with Local Administrator authority.

## Procedure

To uninstall DB2 Accessories Suite:

1. Stop the DB2 Text Search instance service. To stop the service, run db2ts "STOP FOR TEXT".
2. Log on to the DB2 database server with as a user who has the necessary privileges for the operating system.
3. Disable rich text document support for all text search instances which were enabled with rich text feature before. For details, see the topic about disabling DB2 Text Search for rich text document support.
4. Uninstall DB2 Accessories Suite installer. To uninstall the installer:
  - On Linux and UNIX operating systems:  
`<DB2DIR>/db2tss/Uninstall_DB2ASV10/Uninstall_DB2AS.bin`  
  
where `<DB2DIR>` is the location of the latest DB2 copy.
  - On the Windows operating system:  
`<DB2PATH>\db2tss\Uninstall_DB2ASV10\Uninstall_DB2AS.exe`  
  
where `<DB2PATH>` is the location where you installed the latest DB2 copy.

## Results

You have uninstalled the DB2 Accessories Suite.



---

## Chapter 6. Upgrading DB2 Text Search

---

### Upgrading DB2 Text Search for administrator or root install

To obtain the latest functionality upgrade your DB2 Text Search instance. You need to upgrade the DB2 server, instance, and all databases when upgrading the text search instance.

#### About this task

The following steps describe the process to upgrade DB2 Text Search Version 9.5 or Version 9.7 root installations on Linux or UNIX platforms, or for administrators on the Windows platform.

#### Procedure

1. Log in as the instance owner or a user with SYSADM authority.
2. Stop the DB2 database instance and the DB2 Text Search instance service.
3. Back up the DB2 Text Search configuration directory:
  - For Linux and UNIX operating systems, it is located under:  
`$INSTHOME/sqllib/db2tss/config`  
where *INSTHOME* represents the instance home path.
  - For Windows systems, it is located under:  
`<INSTPROF>\<INSTNAME>\db2tss\config`  
where *<INSTPROF>* represents the instance profile directory and *<INSTNAME>* indicates the name of the instance to be upgraded.
4. If you enabled DB2 Text Search for rich text document support and are upgrading from DB2 Version 9.5 Fix Pack 3 or later, disable rich text document support. For more information about how to disable rich text document support, see the topic about disabling DB2 Text Search for rich text document support.
5. Log on to the DB2 server as root on Linux and UNIX operating systems or user with Local Administrator authority on Windows operating systems. If you are upgrading a multi-partitioned instance, you must perform instance upgrade from the instance-owning partition.
6. Install a new copy of DB2 Enterprise Server Edition for Linux, UNIX, and Windows Version 10.1 with a custom installation and make sure DB2 Text Search is selected. DB2 Text Search is an optional component that is only available when you select a custom installation. You also can choose to install a new DB2 Version 10.1 copy over an earlier copy of DB2 by selecting Work-With-Existing mode and selecting DB2 Text Search as the component to be upgraded. You don't need to upgrade the DB2 instances after the install with this approach.
7. Upgrade DB2 Text Search server for your DB2 instances by issuing the **configTool upgradeInstance** command.
  - For Linux and UNIX operating systems:  
`$DB2DIR/db2tss/bin/configTool upgradeInstance  
-installedConfigPath $DB2DIR/cfg/db2tss/config  
-configPath $INSTHOME/sqllib/db2tss/config`

where, *INSTNAME* is the instance home directory and *DB2DIR* is the location of the newly installed DB2 Version 10.1 copy.

- For Windows operating systems:

```
<DB2PATH>\db2tss\bin\configTool upgradeInstance  
-installedConfigPath "<DB2PATH>\CFG\DB2TSS\CONFIG"  
-configPath "<INSTPROFDIR>\<INSTANCENAME>\DB2TSS\CONFIG"
```

where, *<DB2PATH>* is the location of the newly installed DB2 Version 10.1 copy and *<INSTPROFDIR>* is the instance profile directory.

**Note:** For Windows systems, if the DB2 instance was not configured previously for DB2 Text Search and the DB2 version to be upgraded is Version 9.7 Fix Pack 1 or later, you can skip this step.

The **configTool upgradeInstance** command replaces, modifies, and merges text search configuration and data files and directories.

### The config directory

The command copies the following files into the *<ECMTS\_HOME>\config* directory if the files do not already exist:

- constructors.xml
- ecmts\_logging.properties
- ecmts\_config\_logging.properties
- mimetypes.xml
- monitoredEventsConfig.xml

The following files are copied and any existing files are overwritten:

- build\_info.properties
- constructors.xsd
- ecmts\_config\_logging.properties

The following files are merged. Values are added for new settings, and values are maintained for existing settings.

- config.xml
- jetty.xml

The following files are not modified:

- authentication.xml
- key.txt
- All files in the collections subdirectory

### The log directory

The command does not change the contents of the existing log directory. However, when new log files are generated, those new files might replace existing log files.

The **configTool upgradeInstance** command does not upgrade text search filters for an integrated text search server.

8. Upgrade the current DB2 instance by issuing the **db2iupgrade** command.
  - For Linux and UNIX operating systems, the command is located under *\$DB2DIR/instance* directory, where *DB2DIR* is the location of the newly installed DB2 database server Version 10.1 copy.

```
db2iupgrade -j "TEXT_SEARCH [[,service-name] | [,port-number]]" DB2INST
```

- For Windows operating systems, the property file is located under <DB2PATH>\bin directory, where <DB2PATH> is the location of the newly installed DB2 Version 10.1 copy.

```
db2iupgrade DB2INST /j "TEXT_SEARCH [[,service-name] | [,port-number]]"
```

For more information, see the topic about **db2iupgrade** command.

**Note:** If you installed a new copy of DB2 Version 10.1 with the upgrade option, selecting DB2 Text Search as a feature to be upgraded, then you can skip this step.

9. Back up values for all configurable properties of DB2 Text Search used in previous release by running the following script:

- For Linux and UNIX operating systems:

```
$DB2DIR/db2tss/bin/bkuptscfg.sh $INSTNAME
```

where, *DB2DIR* represents the location of the newly installed DB2 Version 10.1 copy, and *INSTNAME* represents the name of the instance to be upgraded.

- For Windows operating systems:

```
<DB2PATH>\db2tss\bin\bkuptscfg.bat <INSTANCENAME> <DB2PATH>
```

where, <DB2PATH> represents the location of the newly installed DB2 Version 10.1 copy, <INSTANCENAME> represents the name of the instance to be upgraded.

The backed-up configurable properties are redirected into one property file:

- For Linux and UNIX operating systems, the property file is located under *\$INSTHOME*/sql1lib/db2tss/config/db2tssrvupg.cfg directory, where *INSTHOME* represents the instance home directory.
- For Windows operating systems, the property file is located under <INSTPROFDIR>\<INSTANCENAME>\db2tss\config\db2tssrvupg.cfg directory, where <INSTPROFDIR> represents the instance profile directory, which can be obtained by issuing the **db2set DB2INSTPROF** command, and <INSTANCENAME> represents the name of the instance to be upgraded.

**Note:** If the DB2 instance was not configured with DB2 Text Search in an earlier copy of a DB2 release, you can skip this step.

10. Set the *DB2INSTANCE* environment variable to the current upgraded instance.
11. Upgrade the databases by issuing **db2 upgrade database** command.
12. For each upgraded database, verify whether the text search server properties information in text search SYSIBMTS.TSSERVERS catalog table is correct, by comparing the property values backed up in step 7. If the value of token or port number in the catalog table is empty or incorrect, you need to update the text server information manually. For details about how to update, see the topic about updating DB2 Text Search server information.
13. Review the values for all DB2 Text Search configurable properties. Compare with the values that you backed up to ensure that they have correct values. Issue the following command to check the configuration values:
 

```
configTool printAll -configPath <configuration-directory>
```
14. If you disabled DB2 Text Search for rich text document support, you have to install DB2 V10 Accessories Suite. For more information, see the topic about installing DB2 Accessories Suite.
15. Then enable rich text document support. For more information, see the topic about enabling DB2 Text Search for rich text and proprietary format support

16. Verify that the upgrade was successful by starting the DB2 Text Search instance service. If you disabled rich text document support, verify that rich text document support is enabled by issuing text search queries and compare with pre-upgrade results.

---

## Upgrading DB2 Text Search for non-root install (Linux and UNIX)

To obtain the latest functionality, upgrade your DB2 Text Search instance. You need to upgrade the DB2 server, instance, and all databases when upgrading the text search instance.

### About this task

The following steps describe the process to upgrade DB2 Text Search Version 9.5 or Version 9.7 non-root installations on Linux or UNIX platforms.

### Procedure

To upgrade DB2 Text Search non-root installations:

1. Log in as the instance owner or a user with SYSADM authority. Then stop the DB2 instance and the DB2 Text Search instance service.
2. Back up the old DB2 copy into a *<backup-dir>* directory.
3. If you enabled DB2 Text Search for rich text document support and are upgrading from DB2 Version 9.5 Fix Pack 3 or later, disable rich text document support. For more information about how to disable rich text document support, see disabling DB2 Text Search for rich text document support.
4. Log on to the DB2 server as a non-root user. Review the database instance type to ensure it can be upgraded as a non-root installation.
5. Install a new copy of DB2 V10.1 with the upgrade option selecting DB2 Text Search component to be upgraded. If you specified the **-f nobackup** parameter and the DB2 database product installation failed, you need to manually install the DB2 database product selecting the DB2 Text search component from the feature tree and then upgrade the non-root instance by issuing the following command:

```
db2nrupgrade -b <backup-dir> -j "TEXT_SEARCH"
```

where, *<backup-dir>* specifies the directory where the configuration files from the old DB2 version are stored. For more details about the **db2nrupgrade** command, see the topic about db2nrupgrade.

6. Back up values for all configurable properties of DB2 Text Search used in previous release before the database upgrade by running the following script :  
`$INSTHOME/sqllib/db2tss/bin/bkuptscfg.sh`

The backed-up configurable properties are redirected into the `$INSTHOME/sqllib/db2tss/config/db2tssrvupg.cfg` property file.

7. Upgrade the existing databases by issuing **UPGRADE DATABASE** command.
8. For each upgraded database, verify whether the text search server properties information in text search catalog table SYSIBMTS.SYSTSSERVERS catalog table is correct, by comparing the property values backed up in step 6. If the value of token or port number in the catalog table is empty or incorrect, you need to update the text server information manually. For how to update, see the topic about updating DB2 Text Search server information.

9. Enable root-based features. You might ask a system administrator with root access to issue the **db2rfe** command
10. Upgrade DB2 Text Search server for your instances by issuing the **configTool upgradeInstance** command.

- For Linux and UNIX operating systems:

```
configTool upgradeInstance
  -installedConfigPath $DB2DIR/cfg/db2tss/config
  -configPath $INSTHOME/sql1lib/db2tss/config
```

where, *INSTHOME* is the instance home directory and *DB2DIR* is the location of the DB2 Version 10.1 copy.

11. Review the values for all DB2 Text Search configurable properties. Compare with the values that you backed up to ensure that they have correct values. issue the following command to check the configuration values:
 

```
configTool printAll -configPath configuration-directory
```
12. If you disabled DB2 Text Search for rich text document support, you have to install DB2 V10 Accessories Suite. For more information, see the topic about installing DB2 Accessories Suite.
13. Then enable rich text document support. For more information, see the topic about enabling DB2 Text Search for rich text and proprietary format support
14. Verify that the upgrade was successful by starting the DB2 Text Search instance service. If you disabled rich text document support, verify that rich text document support is enabled by issuing text search queries and compare with pre-upgrade results.

---

## Upgrading a multi-partition instance without DB2 Text Search

To obtain the latest functionality upgrade your DB2 Text Search instance. You need to upgrade the DB2 server, instance, and all databases when upgrading the text search instance.

### About this task

Starting in DB2 Version 10.1, text search now support indexes in a partitioned database environment. The following steps describe the process to upgrade a DB2 Version 9.5 or Version 9.7 multi-partition instance for root install. DB2 Text Search should not be installed on the instances.

### Procedure

1. Log in as the instance owner or a user with SYSADM authority.
2. Install a new copy of the DB2 Text Search version you are upgrading to, and perform a custom installation. DB2 Text Search is an optional component that is only available when you select a custom installation.
3. Upgrade your instances by issuing the **db2iupgrade** command:
 

```
db2iupgrade /j "text_search [[,service-name] | [,port-number]]"
```
4. Upgrade the existing databases by issuing **DB2 UPGRADE DATABASE** command.
5. For each upgraded database, update the text server information manually. For more information, see the topic about updating DB2 Text Search server information.

---

## Upgrading a stand-alone DB2 Text Search Server

If you already installed the stand-alone DB2 Text Search server, you will need to install fixes to your existing installation to obtain the latest supported features and functionality.

### Before you begin

Before you install a fix, read all attached release notes to determine the prerequisites or migration procedures that apply.

### Procedure

1. Navigate to the `<ECMTS_HOME>` directory for your current installation of the stand-alone DB2 Text Search server.
2. Stop the text search server by running the shutdown script.
  - On Linux and UNIX platforms:  
`<ECMTS_HOME>/bin/shutdown.sh`
  - On Windows platforms:  
`<ECMTS_HOME>\bin\shutdown`
3. Copy the fix installation executable file to the `<ECMTS_HOME>` directory. Ensure that the `ecmts_response.txt` file from the installation resides in this directory.
4. Start installing the fix by entering one of the following commands. The information in the response file should not be modified.  
`<ecmts_setup_file_name> -i silent -f ecmts_response.txt`

**Note:** Installing the updates does not change the authentication token.



---

## Chapter 7. Configuring and administering text search indexes

---

### Command-line tools for DB2 Text Search

Five command-line tools are included with DB2 Text Search to facilitate its use.

#### The Configuration Tool

For performing both the initial and subsequent configurations of DB2 Text Search

#### The Administration Tool

For performing various administrative tasks related to the DB2 Text Search server

#### The Synonym Tool

For adding synonym dictionaries to text search indexes and removing synonym dictionaries from text search indexes

#### The Stop Word Tool

For removing frequently occurring terms, referred to as stop words, from text search queries

#### The Log Formatter Tool

For viewing and saving system messages and trace messages

---

### Issuing text search commands

You can issue commands by running the **db2ts** command shell or by calling one of the administrative SQL routines that is a stored procedure for DB2 Text Search.

#### About this task

To use the **db2ts** command shell, pass the command string as a parameter. The **db2ts** command shell acts like the DB2 command shell in that a command must contain the connection information if a remote database is used. Unlike the DB2 command shell, however, **db2ts** does not provide a session; instead, each command is a separate unit and thus must establish a connection separately. You do not have to specify the database connection if you are running the command locally for the default database specified using the **DB2DBDFT** environment variable. Set the **DB2DBDFT** environment variable at the operating system level. If you also set it using the **db2set** command, ensure that the same value is used.

Using an administrative SQL routine enables you to issue administration calls from a DB2 client on which you have not installed DB2 Text Search. You can call either the generic **SYSTS\_ADMIN\_CMD** administrative SQL routine with a command string as a parameter or the specific administrative SQL routine for that command.

**Note:** Error messages resulting from **db2ts** commands are written in the client locale, but messages resulting from the administrative routines are written in the locale specified by the message-locale argument or in **en\_US** if you do not specify a locale.

Because some commands are not related to a specific database, for example, **START FOR TEXT** and **STOP FOR TEXT**, you can run them only using the **db2ts** command shell.

---

## Rich text and proprietary format support

### Enabling DB2 Text Search for rich text document support

Rich text support can be enabled on properly configured DB2 Text Search servers.

#### Before you begin

To enable rich text document support for DB2 Text Search servers you must, as the instance owner, run the **richtextTool** utility with the enable option.

Before enabling rich text document support, each DB2 Text Search server must be prepared for rich text document support. For more information, see “Installing DB2 Accessories Suite for DB2 Text Search” on page 63

#### Restrictions

In order to run **richtextTool enable**, you must be logged on as the instance owner.

#### Procedure

1. Log on as the instance owner.
2. Stop the DB2 Text Search instance service. To stop the service, run **db2ts STOP FOR TEXT**.
3. Run the **richtextTool** utility from a DB2 command window to enable support.

- For Linux and UNIX operating systems:

```
$INSTHOME/sql1lib/db2tss/bin/richtextTool enable DB2DIR
```

where *INSTHOME* is the instance home directory and *DB2DIR* is the location of the latest DB2 copy.

- For Windows operating systems:

```
DB2PATH\db2tss\bin\richtextTool.bat enable DB2PATH
```

where *DB2PATH* is the location where you installed the latest DB2 copy.

4. Start the DB2 Text Search instance service. To start the service, run **db2ts START FOR TEXT**.

#### Results

You have enabled rich text support for a DB2 Text Search server.

### Disabling support for rich text and proprietary formats

Support for rich text and proprietary formats can be disabled at any time on the integrated DB2 Text Search servers.

#### Before you begin

To disable rich text document support for DB2 Text Search servers you must, as the instance owner, run the **richtextTool** utility with the **disable** option.

#### Restrictions

To run the **richtextTool disable** command, you must login as the instance owner.

## Procedure

1. Log on as the instance owner.
2. Stop the DB2 Text Search instance service. To stop the service, run **db2ts "STOP FOR TEXT"**. For more information about this command, see "Stopping the DB2 Text Search instance service."
3. Run the richtextTool utility from the DB2 command window to disable support.
  - For Linux and UNIX operating systems:  

```
$INSTHOME/sql1lib/db2tss/bin/richtextTool disable DB2-install-directory
```

where *INSTHOME* is the instance home directory.
  - For Windows operating systems:  

```
DB2PATH\db2tss\bin\richtextTool.bat disable DB2-install-directory
```

where *DB2PATH* is the location where you installed your DB2 database server copy.
4. Start the DB2 Text Search instance service. To start the service, run **db2ts "START FOR TEXT"**. For more information about this command, see "Starting the DB2 Text Search instance service."

## Results

You have disabled rich text support for a DB2 Text Search server.

---

## Starting the DB2 Text Search instance service

Before you can create and search text indexes, you must start the DB2 Text Search instance service.

### About this task

To start the integrated DB2 Text Search instance service, enter the following command:

```
db2ts "START FOR TEXT"
```

To start the stand-alone text search server, run the startup script from the *<ECMTS\_HOME>* directory:

- On Windows:  

```
<ECMTS_HOME>\bin\startup
```
- On Linux and UNIX:  

```
<ECMTS_HOME>/bin/startup.sh
```

You can check the status of the Text Search server with the following command:

```
db2ts "START FOR TEXT status"
```

---

## Stopping the DB2 Text Search instance service

When you stop the DB2 Text Search instance services, the text search server will close all commands which are currently active.

### About this task

The active commands will be closed as follows:

- creating the collection for the text search index will be completed. This implies that a CREATE INDEX FOR TEXT operation may fail in a multi-partition setup, as a text search index is partitioned into multiple collections.
- if drop collection already started to remove files irreversibly, the drop will be completed, otherwise the command is rolled back
- update will process current documents in the queue but not accept other documents, an initial update will be marked as attempted and restart, an incremental update will repeat processing all entries in the staging table.

New commands will not be accepted while the text search server completes the stop processing.

## Procedure

To stop the DB2 Text Search server:

- for the integrated DB2 Text Search instance service, enter the following command:
 

```
db2ts "STOP FOR TEXT"
```
- for the stand-alone text search server, run the shutdown script from the `<ECMTS_HOME>` directory, where `<ECMTS_HOME>` represents the installation directory of the stand-alone text search server.
  - On Windows:
 

```
<ECMTS_HOME>\bin\shutdown
```
  - On Linux and UNIX:
 

```
<ECMTS_HOME>/bin/shutdown.sh
```

---

## Enabling a database for DB2 Text Search

You must enable each database that contains columns of text to be searched. You can enable a database for DB2 Text Search by using the **db2ts ENABLE DATABASE FOR TEXT** command or the `SYSPROC.SYSTS_ENABLE` stored procedure.

### Before you begin

The authorization ID of the statement must hold the `SYSTS_ADM` role and `DBADM` authority.

### About this task

When you enable a database, you can use the following views to get information about the text search indexes in the database and their properties:

#### **SYSIBMTS.TSDEFAULTS**

Shows the database default values for index, text, and processing characteristics

#### **SYSIBMTS.TSLOCKS**

Shows information about command locks set at the database and index level

#### **SYSIBMTS.TSINDEXES**

Shows all text search indexes and their settings

#### **SYSIBMTS.TSCONFIGURATION**

Shows the index configuration parameters

## SYSIBMTS.TSCOLLECTIONNAMES

Shows the collection names for each index

## SYSIBMTS.TSERVERS

Shows the Text Search server connection information

After you enable a database for text search, it remains enabled until you explicitly disable it.

To prepare the database for use with DB2 Text Search, use one of the following methods:

- Enter the following command:

```
db2ts "ENABLE DATABASE FOR TEXT CONNECT TO databaseName"
```

The enable operation attempts to populate the connection information for the text search server in the SYSIBMTS.TSERVERS administrative view. However, the information might be incomplete or insufficient. After the command completes either successfully or with a warning for incomplete enablement, review the values in SYSIBMTS.TSERVERS view and update as necessary.

You must do this step only once for each database. You do not have to enable a database each time that you stop and restart the instance services.

For example, to enable a database named SAMPLE, enter the following command:

```
db2ts "ENABLE DATABASE FOR TEXT CONNECT TO SAMPLE"
```

- Call one of the administrative SQL routines, as follows:

```
- CALL SYSPROC.SYSTS_ADMIN_CMD  
  ('ENABLE DATABASE FOR TEXT', 'en_US', ?)  
- CALL SYSPROC.SYSTS_ENABLE('en_US', ?)
```

---

## Disabling a database for DB2 Text Search

Disable a database when you no longer intend to perform text searches in that database.

### About this task

When you disable a database for text search, catalog tables and administrative views are dropped from the SYSIBMTS schema.

### Procedure

To disable a database for text search, use one of the following methods:

1. Drop any text search indexes defined in the database, using the **DROP INDEX** command.
2. To disable a database for text search, use one of the following methods:
  - Issue the **DISABLE DATABASE FOR TEXT** command:

```
db2ts "DISABLE DATABASE FOR TEXT CONNECT TO databaseName"
```
  - Call the SYSPROC.SYSTS\_DISABLE procedure:

```
CALL SYSPROC.SYSTS_DISABLE('en_US', ?)
```

**Note:** Text search indexes can also be dropped using the **FORCE** option. However, it is possible that some data, specifically a text search collection, will remain after you disable the database. This can occur because the **FORCE** option

allows you to drop text search indexes even if the DB2 Text Search server cannot be reached. Such a remaining collection needs to be explicitly removed with the CLEANUP operation.

---

## Deleting orphaned DB2 Text Search collections

You can delete orphaned collections with the **db2ts CLEANUP FOR TEXT** command or use the following process to identify and remove orphaned collections by using the administration tool.

### About this task

A text search index is associated with a single collection for non-partitioned or single-partition databases, and with *n* collections for multi-partition databases with *n* the number of relevant data partitions. Although **db2ts** commands and procedures operate on text search indexes, the text search tools operate on the text search collections. When a text search index no longer exists but its corresponding text search collection does, it is called an orphaned collection.

A collection will get orphaned in the following scenarios:

- dropping a database that contains the text index
- using the FORCE option with the DISABLE or DROP index operation

These operations succeed even if the Text Search server is not reachable.

A collection may also get an orphaned or an invalid status in some failure scenarios. For example, a disk crash may cause an inconsistency in the text index metadata.

To determine whether any orphaned collections exist:

1. Use the administration tool to report all text search collections. Issue the following command:  

```
adminTool status -configPath <absolute-path-to-configuration-folder>
```
2. Query the SYSIBMTS.TSCOLLECTIONNAMES administrative view to report all text search indexes on the current database:  

```
SELECT collectionname FROM SYSIBMTS.TSCOLLECTIONNAMES
```

Perform this query on all the databases enabled for DB2 Text Search, and combine the results into a list.

The administration tool lists all text search collections, while the query on the SYSIBMTS.TSCOLLECTIONNAMES view lists only text search indexes on the current database.

3. Compare the lists returned by the administration tool and by the SELECT statement. Any text search collection returned by the administration tool but not by the SELECT statement is an orphaned collection. The only exception to this rule is the default collection that is created when the DB2 Text Search server is started.

Remove the orphaned text search collection with the following command:

```
adminTool delete -configPath <absolute-path-to-configuration-folder>  
-collectionName collection-name
```

**Important:** The action performed by the **adminTool delete** command is not recoverable and is equivalent to dropping an index or rendering an index inconsistent.

## Example

You currently have DB2 Text Search enabled for a database called DBCP1208, which is running on a UNIX system. To determine whether any orphaned text search collections exist, use the administration tool and a SELECT statement:

```
adminTool.sh status -configPath $HOME/sqllib/db2tss/config
```

CollectionName	IndexSize	NumOfDocuments
Default	13,159B	0
tigertail_DBCP1208_TS542717_0000	13,159B	11
tigertail_DBCP1208_TS012817_0000	13,159B	17
tigertail_DBCP1208_TS082817_0000	13,159B	16
tigertail_DBCP1208_TS152817_0000	13,159B	18
tigertail_DBCP1208_TS212817_0000	13,159B	16
tigertail_DBCP1208_TS302817_0000	13,159B	17
tigertail_DBCP1208_TS392817_0000	13,159B	10
tigertail_DBCP1208_TS462817_0000	13,159B	10
tigertail_DBCP1208_TS542817_0000	13,159B	12
tigertail_DBCP1208_TS022917_0000	13,159B	10
tigertail_DBCP1208_TS112917_0000	13,159B	16
tigertail_DBCP1208_TS192917_0000	13,159B	11
tigertail_DBCP1208_TS262917_0000	13,159B	12
tigertail_DBCP1208_TS867530_0000	13,159B	16

```
db2 select collectionname from sysibmts.tscollectionnames
```

```
COLLECTIONNAME
```

```
-----  
tigertail_DBCP1208_TS542717_0000  
tigertail_DBCP1208_TS012817_0000  
tigertail_DBCP1208_TS082817_0000  
tigertail_DBCP1208_TS152817_0000  
tigertail_DBCP1208_TS212817_0000  
tigertail_DBCP1208_TS302817_0000  
tigertail_DBCP1208_TS392817_0000  
tigertail_DBCP1208_TS462817_0000  
tigertail_DBCP1208_TS542817_0000  
tigertail_DBCP1208_TS022917_0000  
tigertail_DBCP1208_TS112917_0000  
tigertail_DBCP1208_TS192917_0000  
tigertail_DBCP1208_TS262917_0000
```

```
13 record(s) selected.
```

Comparing the two outputs, you see that the text search collection `tigertail_DBCP1208_TS867530_0000` does not have a corresponding text search index. Use the administration tool to delete that orphaned collection:

```
adminTool.sh delete -configPath $HOME/sqllib/db2tss/config  
-collectionName tigertail_DBCP1208_TS867530_0000
```

---

## Synonym dictionaries for DB2 Text Search

A synonym dictionary contains words that are synonyms of each other. You can use a synonym dictionary to search for synonyms of your query terms in a text search index, thus improving the results of your search queries.

Using a synonym dictionary, you can search for words specific to your organization, such as acronyms and technical jargon.

By default, a synonym dictionary is not used for a search. To use a synonym dictionary, you must explicitly add it to a specific text search index. The text search

index needs to be updated at least once before you can add a synonym dictionary. After the synonym dictionary has been added, you can modify it as frequently as you want.

A synonym dictionary consists of synonym groups that you define in an XML file, as shown in the following example:

```
<?xml version="1.0" encoding="UTF-8"?>
<synonymgroups version="1.0">
  <synonymgroup>
    <synonym>ball</synonym>
    <synonym>globe</synonym>
    <synonym>sphere</synonym>
    <synonym>orb</synonym>
  </synonymgroup>
  <synonymgroup>
    <synonym>worldwide patent tracking system</synonym>
    <synonym>wpts</synonym>
  </synonymgroup>
</synonymgroups>
```

## Adding a synonym dictionary for DB2 Text Search

You can easily add a synonym dictionary to a text search index by using the Synonym Tool.

### Before you begin

- You must activate the DB2 Text Search instance service before you can add a synonym dictionary to a text search index.
- You must have updated the text search index at least once.
- You must also have a synonym XML file that specifies synonym groups.

### Procedure

To add a synonym dictionary:

1. Copy the XML file to any directory on the DB2 Text Search server.
2. Determine the name of the text search collection associated with the text search index to which you want to add the synonym dictionary. You can use the Administration Tool to report all text search collections, as follows:  

```
adminTool status -configPath absolute-path-to-config-folder
```
3. Use the Synonym Tool to add the synonym dictionary to the specific text search index. You can add the synonyms in **append** or **replace** mode, meaning that you either add them to or replace the existing synonyms defined for that text search index.

```
synonymTool importSynonym -synonymFile absolute-path-to-syn-file
  -collectionName collection-name -replace true or false
  -configPath absolute-path-to-config-folder
```

**Note:** If the XML format is not valid or if the XML file is empty, an error is returned.

### Example

For example, to add the synonym file `synfile.xml` in append mode, use the following command:



```
synonymTool importSynonym
-synonymFile $HOME/sql1lib/misx/xmlsynfile.xml
-collectionName tigertail_DBCP1208_TS867530_0000
-replace false
-configPath $HOME/sql1lib/db2tss/config
```

## Removing a synonym dictionary for DB2 Text Search

You need to remove synonym dictionaries on a collection-by-collection basis, so you must use the Synonym Tool on all collections that exist for a text search index.

### About this task

To remove a synonym dictionary, use the following command:

```
synonymTool removeSynonym -collectionName collection-name
-configPath absolute-path-to-config-folder
```

Where *collection-name* specifies the text search collection and *absolute-path-to-config-folder* specifies the absolute path to the text search configuration folder.

---

## Text search index creation

A text search index is a compilation of significant terms extracted from text documents. Each term is associated with the document from which it was extracted.

You create a text search index once for each column that contains text to be searched. When you create a text search index, you also create the following objects:

### A staging table

This keeps track of all changed rows in the user table.

### An auxiliary staging table (optional)

This keeps track of inserts and updates in the user table via integrity processing.

### An event table

This collects information about the status of an update index command or any errors encountered during its processing. If errors occur during indexing, *index update events* are added to the event table.

### Triggers on the user table

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

**Note:** If you use the **LOAD** command to populate your documents, triggers are not activated, and incremental indexing of the loaded documents will not work. Instead, use the **IMPORT** command, which does activate triggers. Alternatively you can add the auxiliary infrastructure for integrity processing, this will recognize changes for example, with the **LOAD INSERT** command.

After you create a text search index, it is empty and, therefore, not searchable, until you update it. When creating the text search index, you can specify a frequency which is used by the scheduler to check periodically whether an update of the text search index is required and that the update command is to be run if necessary.

## Creating a text search index

After you enable a database for DB2 Text Search, you can create text search indexes on columns that contain the text that you want to search.

### Before you begin

Creating a text search index requires one of following authorization levels:

- CONTROL privilege on the index table
- INDEX privilege on the index table with either the IMPLICIT\_SCHEMA authority on the database or the CREATEIN privilege on the index table schema
- DBADM with DATAACCESS authority

A primary key must exist for this table. If a primary key does not exist, you must create one before creating the index.

### About this task

If you do not want to manually apply document changes from the table to the text search index, you can specify the UPDATE FREQUENCY parameter to schedule automated updates. Use the UPDATE MINIMUM parameter to control whether the update only runs when a minimum number of changes is made to the table. For example, to specify that MYSCHEMA.MYTEXTINDEX is to be updated after at least five changes have occurred and that the update services are to check every Monday and Wednesday at 12 midnight and 12 noon, issue the following command:

```
db2ts "CREATE INDEX MYSCHEMA.MYTEXTINDEX FOR TEXT ON PRODUCT(NAME)
      UPDATE FREQUENCY d(1,3) h(0,12) m(0) UPDATE MINIMUM 5"
CALL SYSPROC.SYSTS_CREATE('myschema', 'myTextIndex', 'product (name)',
'UPDATE FREQUENCY D(1,3) H(0,12) M(0)' 'UPDATE MINIMUM 5', 'en_US', ?)
```

When you create an index, you can specify its locale (language and territory) by using the **LANGUAGE** option. To have your documents automatically scanned to determine the locale, set the **LANGUAGE** to AUTO. If you do not specify **LANGUAGE**, a default is used. This default is derived using the DEFAULTVALUE from SYSIBMTS.TSDEFAULTS where DEFAULTNAME='LANGUAGE'. (In this case, DEFAULTVALUE is set at the time the database is enabled for text search. This value is derived from the database territory if the database territory can be mapped to one of the document locales supported. If the database territory cannot be used to determine a supported document locale, DEFAULTVALUE is set to AUTO.)

#### Restrictions

- A text column in an index must be one of the following supported types:
  - CHAR
  - VARCHAR
  - LONG VARCHAR
  - CLOB
  - GRAPHIC
  - VARGRAPHIC
  - LONG VARGRAPHIC
  - DBCLOB
  - BLOB
  - XML

- Text search related objects must follow not only DB2 naming conventions, their identifiers must also contain these characters only:
  - [A-Za-z][A-Za-z0-9@#\$\_]\* or
  - "[A-Za-z ][A-Za-z0-9@#\$\_ ]\*"

This limitation applies to the following:

- the name of the schema containing the text search index
- the name of the table the text search index is associated with
- the name of the text column
- the name of the text search index

## Procedure

Create a text search index using one of the following methods:

- Issue the **CREATE INDEX** command:
 

```
db2ts "CREATE INDEX index-name FOR TEXT ON table-name (column-name)"
```
- Call the SYSPROC.SYSTS\_CREATE stored procedure:
 

```
CALL SYSPROC.SYSTS_CREATE('index-schema', 'index-name', 'table-name
(column-name)', 'options', 'locale', ?)
```

**Note:** Schema name and index name are case-sensitive when the stored procedure is used.

## Example

For example, the PRODUCT table in the SAMPLE database includes columns for the product ID, name, price, description, and so on. To create a text search index called MYSCHEMA.MYTEXTINDEX for the NAME column, issue the command or call the stored procedure, as follows:

```
db2ts "CREATE INDEX MYSCHEMA.MYTEXTINDEX FOR TEXT ON PRODUCT(NAME)"
CALL SYSPROC.SYSTS_CREATE('MYSCHEMA', 'MYTEXTINDEX', 'PRODUCT(NAME)', '', 'en_US',?)
```

Similarly, to create a text search index called MYSCHEMA.MYXMLINDEX for the XML column DESCRIPTION, enter the following command:

```
db2ts "CREATE INDEX MYSCHEMA.MYXMLINDEX FOR TEXT ON PRODUCT(DESCRIPTION)"
```

or

```
CALL SYSPROC.SYSTS_CREATE('MYXMLINDEX', 'MYXMLINDEX',
'PRODUCT (DESCRIPTION)', '', 'en_US', ?)
```

## Creating a text search index on binary data types

When creating a text search index, you have the option of specifying a code page for a binary column. Doing so helps the DB2 Text Search engine identify the character encoding.

### About this task

To specify the code page when creating the text search index, use the following command:

```
db2ts "CREATE INDEX index-name FOR TEXT ON table-name
CODEPAGE code-page"
```

When you store data in a column having a binary data type, such as BLOB or FOR BIT DATA, the data is not converted. This means that the documents retain their

original code pages, which can cause problems when you create a text search index because you might have two different code pages. Therefore, you need to determine whether you are using the code page of the database or the code page specified for the db2ts **CREATE INDEX** command. If you do not know which code page was used to create the text search index, you can find out by issuing the following statement:

```
db2 "SELECT CODEPAGE FROM SYSIBMTS.TSINDEXES where INDSHEMA='schema-name'
and INDNAME='index-name'"
```

## Creating a text search index on unsupported data types

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

### About this task

A text column in an index must be one of the following supported types:

- CHAR
- VARCHAR
- LONG VARCHAR
- CLOB
- GRAPHIC
- VARGRAPHIC
- LONG VARGRAPHIC
- DBCLOB
- BLOB
- XML

To convert the data type of the column to one of valid types, use one of the following methods:

- Run the **db2ts CREATE INDEX** command with the name of a transformation function.

```
db2ts "CREATE INDEX index-name FOR TEXT ON
table-name (function-name(text-column-name))"
```

- Use a user-defined external function (UDF), which is specified by *function-name*, that accesses text documents in a column that is not of a supported type for text searching, performs a data-type conversion of that value, and returns the value as one of the supported data types.

### Example

In the following example, there is a table UDTTABLE that contains a column of a user-defined type (UDT) named "COMPRESSED\_TEXT", which is defined as CLOB(1M). To create an index on that data type, first create a UDF called UNCOMPRESS, which receives a value of type COMPRESSED\_TEXT. Next, create your text search index in the following way:

```
db2ts "CREATE INDEX UDTINDEX FOR TEXT ON
UDTTABLE (UNCOMPRESS(text)) ..."
```

## Sample: Creating N-gram and morphological indexes for plain text

### About this task

Use the following instructions to setup and synchronize DB2 Text Search indexes for morphological and N-gram indexing in the SAMPLE database. Search for linguistically meaningful Chinese words.

### Procedure

1. Create two tables for morphological and N-gram indexing. The tables have columns for the book name, author, story, ISBN number and the year the book was published.

```
db2 "CREATE TABLE morphobooks (  
  isbn VARCHAR(18) not null PRIMARY KEY,  
  bookname VARCHAR(30),  
  author VARCHAR(30),  
  story blob(1G),  
  year integer  
)"
```

```
db2 "CREATE TABLE ngrambooks (  
  isbn VARCHAR(18) not null PRIMARY KEY,  
  bookname VARCHAR(30),  
  author VARCHAR(30),  
  story blob(1G),  
  year integer  
)"
```

2. Issue the **CREATE INDEX** command to create a text search index on the STORY column of MORPHOBOOOKS table. The name of the text search index is MORPHOINDEX.

```
db2ts " CREATE INDEX db2ts.morphoindex FOR TEXT  
ON morphobooks (story) LANGUAGE zh_TW  
INDEX CONFIGURATION (CJKSEGMENTATION 'morphological')  
CONNECT TO sample";
```

3. Issue the **CREATE INDEX** command to create a text search index on the STORY column of NGRAMBOOKS table. The name of the text search index is NGRAMINDEX.

```
db2ts " CREATE INDEX db2ts.ngramindex FOR TEXT  
ON ngrambooks (story) LANGUAGE zh_TW  
INDEX CONFIGURATION (CJKSEGMENTATION 'ngram')  
CONNECT TO sample";
```

4. Load data into the two tables.

```
db2 "import from ./data/books.del of DEL lobs from ./data/  
replace into morphobooks";
```

```
db2 "import from ./data/books.del of DEL lobs from ./data/  
replace into ngrambooks";
```

The books.del file has the entry:

```
"0-13-086755-4", "book1", "Julie", "books_zh_TW1.lob.0.449/", 2004
```

The Books\_zh\_TW1.lob large object has the following content:

```
△△△△ △△△△  
△△△△ △△△△  
△△△△ △△△△  
△△△△ △△△△
```

```
△△△△ △△△△
△△△△ △△△△
△△△△ △△△△
△△△△ △△△△
```

5. Synchronize the text search indexes with data from the corresponding table by issuing following commands:

```
db2ts "UPDATE INDEX db2ts.morphoindex FOR TEXT CONNECT TO sample";
```

```
db2ts "UPDATE INDEX db2ts.ngramindex FOR TEXT CONNECT TO sample";
```

6. A search for linguistically meaningful Chinese words is successful here for both morphological and N-gram segmentation.

```
db2 "select bookname from morphobooks where contains (story, '△△') = 1";
```

```
BOOKNAME
-----
book1
```

1 record(s) selected.

```
db2 "select bookname from ngrambooks where contains (story, '△△') = 1";
```

```
BOOKNAME
-----
book1
```

1 record(s) selected.

The output indicates that the result from morphological segmentation is the same as N-gram segmentation

7. Search for meaningless Chinese words to see the difference between morphological and N-gram segmentation.

```
db2 "select bookname from morphobooks where contains (story, '△△') = 1";
```

```
BOOKNAME
-----
```

0 record(s) selected.

```
db2 "select bookname from ngrambooks where contains (story, '△△') = 1";
```

```
BOOKNAME
-----
book1
```

1 record(s) selected.

Only N-gram segmentation returns a book name.

## Sample: Creating N-gram and morphological indexes for rich text and proprietary formats

### About this task

Use the following instructions to setup and synchronize DB2 Text Search indexes for morphological and N-gram indexing in the SAMPLE database. Search for meaningless Chinese words.

## Procedure

1. Create two tables for morphological and N-gram indexing. The tables contain columns k and b, where column k is the primary key, and column b will have rich text data.

```
db2 "create table richtext_morpho(  
k varchar(50)not null,  
b blob (1G),  
primary key(k)  
)"
```

```
db2 "create table richtext_ngram(  
k varchar(50)not null,  
b blob (1G),  
primary key(k)  
)"
```

2. Issue the **CREATE INDEX** command to create a text search index on column b of table RICHTEXT\_MORPHO. The name of the text search index is MORPHOINDEX.

```
db2ts " CREATE INDEX db2ts.morphoindex FOR TEXT  
ON richtext_morpho (b) LANGUAGE zh_CN FORMAT INSO  
INDEX CONFIGURATION (CJKSEGMENTATION 'morphological')  
CONNECT TO sample";
```

3. Issue the **CREATE INDEX** command to create a text search index on on column b of table RICHTEXT\_NGRAM. The name of the text search index is NGRAMINDEX.

```
db2ts " CREATE INDEX db2ts.ngramindex FOR TEXT  
ON richtext_ngram (b) LANGUAGE zh_CN FORMAT INSO  
INDEX CONFIGURATION (CJKSEGMENTATION 'ngram')  
CONNECT TO sample";
```

4. Load data into the two tables.

```
db2 "import from ./data/cjk_richtext.del of DEL lobs from ./data/  
replace into richtext_morpho ";
```

```
db2 "import from ./data/ cjk_richtext.del of DEL lobs from ./data/  
replace into richtext_ngram ";
```

The cjk\_richtext.del file has the entries:

```
"rt_CJK.pdf","rt_CJK.pdf.0.864885/",  
"rt_CJK.pdf.doc","rt_CJK.pdf.doc.0.90112/",  
"rt_CJK.pdf.txt","rt_CJK.pdf.txt.0.37913/"
```

The rt\_CJK.pdf, rt\_CJK.pdf.doc and rt\_CJK.pdf.txt files all have the same content. One segment of the content in Simplified Chinese is as follows:

```
"@@@@@@@@@  
IBM Rational License Key Center @@@@@@@@@@@@@@@@@@@@,@@@@@@@@@@@ Rational  
@@ @@@@@@@@@@@@@@ IBM Rational  
License Key Center @@@@,@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@,  
@@@@@@@@@@@@@@@@@@@@ License  
Key Center @@@@@@@@@@@@@@@@@@@@@@@@@@@@:  
@@ 1 - @@@@@@@@@@@@@@,@@@@"@@@@@@"@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@,  
@@@@@@@@@@ License Key Center  
@@@@@,@@@@@@@@@@@@@@@@@@@@  
@@ 2 - @@ License Key Center @@@@@@@@@@@ License Key Center  
@@@@@@@,@@@@@@@@@@@"
```

5. Synchronize the text search indexes with data from the corresponding table by issuing following commands:

```
db2ts "UPDATE INDEX db2ts.morphoindex FOR TEXT
CONNECT TO sample"
```

```
db2ts "UPDATE INDEX db2ts.ngramindex FOR TEXT
CONNECT TO sample"
```

6. A search for linguistically meaningful Chinese words is successful here for both morphological and N-gram segmentation.

```
db2 "select k from richtext_morpho where contains(b,'△△')=1"
```

```
K
-----
rt_license.pdf
rt_license.pdf.doc
rt_license.pdf.txt
```

3 record(s) selected.

```
db2 "select k from richtext_ngram where contains(b,'△△')=1"
```

```
K
-----
rt_license.pdf
rt_license.pdf.doc
rt_license.pdf.txt
```

3 record(s) selected.

The output indicates that the result from morphological segmentation is the same as N-gram segmentation

7. Search for meaningless Chinese words to see the difference between morphological and N-gram segmentation.

Search for a linguistic meaningless Chinese words.

```
db2 "select k from richtext_morpho where contains(b,'△△')=1"
```

```
K
-----
```

0 record(s) selected.

```
db2 "select k from richtext_ngram where contains(b,'△△')=1"
```

```
K
-----
rt_license.pdf
rt_license.pdf.doc
rt_license.pdf.txt
```

3 record(s) selected.

Only N-gram segmentation returns a book name.

---

## Text search index maintenance

After you create text search indexes, there are several maintenance tasks that you need to perform. There are several ways to perform these tasks, including using various administration commands, stored procedures, and the Administration Tool.

The routine text search index maintenance tasks include the following ones:

- Running periodic updates

Unless you specified that automatic updates are to be performed, you must update the text search indexes to reflect changes in the indexed text columns that they are associated with.



- Monitoring the event table  
You can use the event table to determine whether there are document errors or whether the index update frequency needs to change.

Less frequent maintenance tasks include altering and dropping text search indexes.

## Administration commands for DB2 Text Search

There are a number of commands that allow you to administer DB2 Text Search at the instance, database, table, and text-index levels. You run all of the commands using `db2ts`.

Use the instance-level administration commands to start and stop the DB2 Text Search instance services and clean up text search indexes that are no longer usable:

**db2ts START FOR TEXT**

Starts the DB2 Text Search instance services

**db2ts STOP FOR TEXT**

Stops the DB2 Text Search instance services

**db2ts CLEANUP FOR TEXT**

Cleans up any text search collections that are not usable

Use the database-level administration commands to set up or disable databases for DB2 Text Search and clear command locks:

**db2ts ENABLE DATABASE FOR TEXT**

Enables the current database to create, manage, and use text search indexes

**db2ts DISABLE DATABASE FOR TEXT**

Disables DB2 Text Search for a database and drops a number of text search catalog tables and views

**db2ts CLEAR COMMAND LOCKS**

Deletes command locks for all indexes in a database

Use table- and index-level commands to create and manipulate text search indexes on columns of a table:

**db2ts CREATE INDEX**

Creates a text search index

**db2ts DROP INDEX**

Drops a text search index associated with a text column

**db2ts ALTER INDEX**

Changes the characteristics of a text search index

**db2ts UPDATE INDEX**

Populates or updates a text search index based on the current contents of a text column

**db2ts CLEAR EVENTS FOR TEXT**

Deletes events from the `SYSIBMTS.TSEVENT` view, an events view that provides information about indexing status and errors

**db2ts CLEAR COMMAND LOCKS FOR INDEX**

Deletes all command locks for a specific text search index

**db2ts RESET PENDING FOR TABLE**

Identifies all dependent tables that are maintained for text search and executes set integrity, if necessary

## **db2ts HELP**

Displays the list of **db2ts** command options and information about specific error messages

## **DB2 Text Search stored procedures**

DB2 Text Search provides several administrative SQL routines for running commands and for returning the result messages of the commands that you run and the result message reason codes.

You can run the following **db2ts** commands using the administrative SQL routines:

- Enable a database - **SYSPROC.SYSTS\_ENABLE**
- Configure a database - **SYSPROC.SYSTS\_CONFIGURE**
- Disable a database - **SYSPROC.SYSTS\_DISABLE**
- Create a text index - **SYSPROC.SYSTS\_CREATE**
- Update a text index - **SYSPROC.SYSTS\_UPDATE**
- Alter a text index - **SYSPROC.SYSTS\_ALTER**
- Drop a text index - **SYSPROC.SYSTS\_DROP**
- Clear events for a text index - **SYSPROC.SYSTS\_CLEAR\_EVENTS**
- Clear command locks - **SYSPROC.SYSTS\_CLEAR\_COMMANDLOCKS**
- Reset pending status - **SYSPROC.SYSTS\_ADMIN\_CMD**
- Cleanup inactive indexes - **SYSPROC.SYSTS\_CLEANUP**

## **Updating a text search index**

You can update a text search index automatically or manually. Automatic updates occur based on how you defined the update frequency for the text search index. You can update indexes manually by issuing a command or by calling a stored procedure.

### **Before you begin**

Updating a text search index requires the SYSTS\_MGR role and either the CONTROL privilege or DATAACCESS authority on the target table.

### **About this task**

After creating and updating (filling) the text search index for the first time, you must keep it up to date. For example, when you add a text document to a database or change an existing document in a database, you must index the document to keep the content of the text search index synchronized with the content of the database. Also, when you delete a text document from a database, you must remove its terms from the text search index.

You should plan periodic indexing carefully because indexing text documents is a time- and resource-consuming task. The time taken depends on many factors, including how big the documents are, how many documents you added or changed since the previous text search index update, and how powerful your processor is.

The Administration Tool's status option can be used to retrieve information about the progress of document updates while the **db2ts UPDATE INDEX** command is running. If an index update is still in progress when a new update starts, the new update fails.

- Automatic updates

To have text search index updates performed automatically, use one of the following commands to set an UPDATE FREQUENCY:

- **db2ts CREATE INDEX**
- **db2ts ALTER INDEX**

The **UPDATE FREQUENCY** parameter has a minimum setting of five minutes. The **UPDATE MINIMUM** parameter specifies the minimum number of text changes that must be queued.

If there are not enough changes in the staging table for the specified day and time, the text search index is not updated.

- Manual updates

- There are also times when you want to update a text search index immediately. For example, after you create a text search index, when the index is still empty, or after you have added several text documents to a database and want to search.

To fill or synchronize (update) a text search index with the table data, use one of the following methods:

- Issue the **UPDATE INDEX** command:  
db2ts "UPDATE INDEX *index-name* FOR TEXT"
- Call the SYSPROC.SYSTS\_UPDATE administrative SQL routine.

## Example

For example, suppose that there are two text search indexes on the PRODUCT table: MYSCHEMA.MYTEXTINDEX on the NAME column and MYSCHEMA.MYXMLINDEX on the DESCRIPTION column. A new entry is added to PRODUCT as follows:

```
INSERT INTO PRODUCT VALUES ('100-104-01', 'Wheeled Snow Shovel', 99.99, NULL,
NULL, NULL, XMLPARSE(DOCUMENT '<product xmlns="http://posample.org/wheelshovel"
pid="100-104-01"><description><name>Wheeled Snow Shovel</name>
<details>Wheeled Snow Shovel, lever assisted, ergonomic foam grips, gravel wheel,
clears away snow 3 times faster</details><price>99.99</price>
</description></product>'))
```

To make the information in the new entry searchable, issue the following command:

```
db2ts "UPDATE INDEX MYSCHEMA.MYTEXTINDEX FOR TEXT"
```

To make the information in the new entry searchable, use the following stored procedure:

```
db2 "call sysproc.systs_update('MYSCHEMA', 'MYXMLINDEX', '', 'en_US', ?)'
```

## Sample: Incrementally updating a DB2 Text Search index on range-partitioned tables

Incremental updates of DB2 Text Search indexes on range-partitioned tables require the extended text-maintained staging infrastructure to apply changes from attaching or detaching partitions.

### About this task

When the extended staging infrastructure is enabled for the text search indexes, document updates are captured through an update trigger into the primary staging table, and document inserts and deletes are captured in the auxiliary staging table through integrity processing.

When the extended staging infrastructure is not enabled, you cannot use an incremental update to process changes related to attaching or detaching ranges or to process documents that you loaded into an added partition by using the **LOAD** command with the **INSERT** parameter. You must re-create the text index to synchronize it with the base table.

By default, the extended text-maintained infrastructure will be added for text search indexes on range-partitioned tables, however, for scenarios where the text search index is not refreshed with incremental updates, you can create the text search index with the **AUXLOG** option set to OFF as shown in the following example:

```
db2ts create index sampleix for text on sample(comment) administration tables in
mytablespace index configuration(auxlog off) connect to mydb
```

In this case, only a primary staging table is added, and document changes are recognized through triggers, which excludes changes for example, from attach or detach operations. You must specify the **ADMINISTRATION TABLES IN** parameter when creating indexes on range-partitioned tables; otherwise, an error is generated.

## Example

### Scenario 1: To attach a partition for a table with the extended text search staging infrastructure

1. Create a range-partitioned table.
 

```
db2 "create table uc_007_customer_archive (pk integer not null
primary key, customer varchar(128) not null,
year integer not null, address blob(1M) not null) partition
by range(year)(starting(2000)ending(2001)every 1)"
```
2. Create the text search index.
 

```
db2ts "create index uc_007_idx for text on
uc_007_customer_archive (address)
administration tables in mytablespace"
```
3. View the index name and logging information.
 

```
db2 "select indexname, stagingviewname, auxstagingname
from sysibmts.tsindexes"
```
4. Update the text search index.
 

```
db2ts "update index uc_007_idx for text"
```
5. Create another table and import data into the table.
 

```
db2 "create table uc_007_customer_2001 (pk integer not null
primary key,
customer varchar(128) not null, year integer not null,
address blob(1M) not null)"
db2 "import from uc_007_2001.del of del lobs
from ./data modified by codepage=1208
insert into uc_007_customer_2001"
```
6. Add the data from the new table as a new partition.
 

```
db2 "alter table uc_007_customer_archive attach
partition p2001 starting(2001) ending(2002)
exclusive from uc_007_customer_2001"
```
7. View the contents.
 

```
db2 "select * from sysibmts.sysauxlog_ix253720"
```

The output is as follows:

PK	GLOBALTRANSID	GLOBALTRANSTIME	OPERATIONTYPE
-----			
0 record(s) selected.			

8. The changes are not visible, so integrity processing is required. Integrity processing places dependent tables in pending mode.  
db2 "set integrity for uc\_007\_customer\_archive immediate checked"
9. View the contents.  
db2 "select \* from sysibmts.sysauxlog\_ix253720"

The following error message is returned:

```
PK          GLOBALTRANSID    GLOBALTRANSTIME    OPERATIONTYPE
-----
SQL0668N Operation not allowed for reason code "1" on table
"SYSIBMTS"."SYSTSAUXLOG_IX253720". SQLSTATE=57016
```

10. Perform integrity processing for the text search staging tables. The command processes all text indexes for the table.  
db2ts "reset pending for table uc\_007\_customer\_archive for text"  
db2 "select \* from sysibmts.sysauxlog\_ix253720"

The output is as follows:

```
PK          GLOBALTRANSID    GLOBALTRANSTIME    OPERATIONTYPE
-----
1  x'000000000002215B'    x'20081020204612500381000000'    1
2  x'000000000002215B'    x'20081020204612500602000000'    1
3  x'000000000002215B'    x'20081020204612500734000000'    1
5  x'000000000002215B'    x'20081020204612500864000000'    1
```

11. Use incremental update to process data from the newly attached partition.  
db2ts "update index uc\_007\_idx for text"

### Scenario 2: To detach a partition for a table with extended text search staging infrastructure

1. Alter the table from the partition.  
db2 alter table uc\_007\_customer\_archive detach partition p2005 into t4p2005

The following message is returned:

```
SQL3601W The statement caused one or more tables
to automatically be placed in the Set Integrity Pending state.
SQLSTATE=01586
```

2. Issue the **RESET PENDING** command to perform integrity processing for the text search staging tables.  
db2ts "reset pending for table uc\_007\_customer\_archive for text"

Use incremental update to process data from the newly detached partition.

```
db2ts "update index uc_007_idx for text"
```

## Clearing text search index events

If you no longer need the messages in the event view of an index, you can clear (delete) them.

### Before you begin

For details, including authorization requirements, see the description for the **CLEAR EVENTS FOR INDEX** command or the **SYSTS\_CLEAR\_EVENTS** procedure.

## About this task

Information about indexing events, such as the update start and end times, the number of indexed documents, or document errors that occurred during the update, are stored in the event view of a text search index. This information can help you determine the cause of a problem.

## Procedure

To clear the event view of a text search index, use one of the following methods:

- Run the **db2ts CLEAR EVENTS FOR INDEX** command, as follows:  

```
db2ts "CLEAR EVENTS FOR INDEX index-name FOR TEXT"
```
- Use the SYSPROC.SYSTS\_CLEAR\_EVENTS administrative SQL routine, as follows:  

```
CALL SYSPROC.SYSTS_CLEAR_EVENTS('index-schema',  
'index-name', 'localē', ?)
```

## Altering a text search index

You can alter the update properties of a text search index.

### Before you begin

For details, including authorization requirements, see the description for the **ALTER INDEX** command or the **SYSTS\_ALTER** procedure.

## Procedure

To alter an index, use one of the following methods:

- Run the following command:  

```
db2ts "ALTER INDEX index-name FOR TEXT update-characteristics"
```

Where *update-characteristics* is a characteristic such as the update frequency of the text search index.

- Call the SYSPROC.SYSTS\_ALTER administrative SQL routine:  

```
CALL SYSPROC.SYSTS_ALTER('db2ts', 'myTextIndex', 'alter-option', 'en_US', ?)
```

Where *alter-option* is a characteristic such as the update frequency of the text search index.

## Results

The text index properties are updated with the new values, except if the text search index is locked by another operation, in which case an error message is displayed, informing you that the text search index is currently locked and that no changes can be made.

## Example

You can use either method to change both the update frequency of a text search index and the minimum number of changes required to trigger an update. (If you do not specify any parameters, the current settings are left unchanged.) For example, to change the update frequency for the text search index MYTEXTINDEX

so that it is updated from Monday to Friday at 12 noon and 3 p.m., provided that at least 100 changes have occurred to the indexed column, issue the following command:

```
db2ts "ALTER INDEX MYTEXTINDEX FOR TEXT
UPDATE FREQUENCY d(1,2,3,4,5) h(12,15) m(00) UPDATE MINIMUM 100"
```

To stop the periodic updating of MYTEXTINDEX, issue the following command:

```
db2ts "ALTER INDEX MYTEXTINDEX FOR TEXT UPDATE FREQUENCY NONE"
```

## Viewing text search index status

To get information about the current text search indexes within a database, you can query the administrative views or use the Administration Tool.

### About this task

Text search index properties can be viewed in the SYSIBMTS.TSINDEXES administrative view. For example, to list all text search indexes with their status, issue the following query:

```
db2 "select indschema, indname, indstatus from SYSIBMTS.TSINDEXES"
```

To check the status of all text search collections and their properties using the Administration Tool, use the following command:

```
adminTool status -configPath absolute-path-to-config-folder
```

## Changing the location of a DB2 Text Search collection

You might need to change the location of a collection, for example, for computer and disk administration and maintenance purposes.

### Before you begin

You can change the location of a text search collection only when the collection location in the SYSIBMTS.TSINDEXES table is empty.

### About this task

To change the location of a collection:

#### Procedure

1. Verify that the collection location is empty .  

```
db2 "select indschema, indname, collectiondirectory, collectionnameprefix
from sysibmts.tsindexes"
```
2. If the targeted collection has no directory information, stop the DB2 Text Search server.
3. Edit the collection configuration `collection.xml` file. The default location of the collection configuration file is `<ECMETS_HOME>\config\collections\  
<collection_name>\collection.xml`.
  - a. Specify the location of the index data.

```
<indexes>
  <index>
    <type>Text</type>
    <path><directory_name></path>
```
  - b. Specify the location of the synonym configuration.

```
<indexes>
  <index>
    <type>Synonym</type>
    <path><directory_name></path>
```

**Note:**

- Escape characters as required in XML. For example, escape a backslash character (the default path separator on Windows) by using "\\".
- If the collection configuration and index data is located in the collection directory, you can specify a path that is relative to the location of the collection.xml file, for example:

```
<indexes>
  <index>
    <type>Synonym</type>
    <path>data/text</path>
```

4. Save your changes to the collection.xml file.
5. Restart the DB2 Text Search services.

## Backing up and restoring text search indexes

### Procedure

- To back up a database with DB2 Text Search indexes:
  1. Get a current list of text index locations for DB2 Text Search indexes.
 

```
db2 "select indschema, indname, collectiondirectory, collectionnameprefix
from sysibmts.tsindexes"
```

If a value for collectiondirectory is not specified, then locations are set using the **defaultDataDir** parameter.

2. Ensure that no DB2 Text Search administrative command is running.
  3. Stop the DB2 Text Search services.
 

```
db2ts stop for text
```
  4. Back up the database. Issue the following command:
 

```
db2 backup database db_name
```
  5. Back up the text search configurations, index directories and subdirectories.
  6. Restart DB2 Text Search services.
- To restore a database with DB2 Text Search indexes:
    1. Make sure that no DB2 Text Search administrative command is running.
    2. Stop the DB2 Text Search services.
 

```
db2ts stop for text
```
    3. Restore the database. Issue the following command:
 

```
db2 restore database db_name
```
    4. Restore the backup of text search configuration and index locations to the same path as before.
    5. Restart DB2 Text Search services.
 

```
db2ts start for text
```

## Dropping a text search index

When you no longer intend to perform text searches in a text column, you can drop the text search index.



## Before you begin

For details, including authorization requirements, see the command description for DROP INDEX or the procedure SYSTS\_DROP.

## About this task

When you drop a text search index, the following other objects are also dropped:

- Index staging and event tables
- Triggers on the user table

If the text search index has an associated schedule, make sure no task is running. Otherwise the scheduled task may need to be removed manually.

Always drop the text search indexes on a table before dropping a table space. If you drop table spaces that contains text search indexes, you may create what is called an *orphaned collection*. When you create a text search index, a collection (the file system representation of the index) is created with an automatically generated name. If the collection remains after the index has been dropped, it can lead to problems with future queries if the following are also true:

- the same database connection is being used,
- a table is created with the same table name,
- a text index with the same name as before is created on this table, and
- the same query is reissued as before.

In this case, a cached query plan might be reused, which could result in a wrong query result.

The **db2ts CLEANUP FOR TEXT** command can only drop obsolete collections and relevant text index catalog records. Administration Tool can be used to remove orphaned collections in this case.

If you plan to drop a database that is enabled for text search, make sure all text search indexes are dropped to avoid orphaned collections.

## Procedure

To drop a text search index, use one of the following methods:

- Issue the **DROP INDEX** command:  

```
db2ts "DROP INDEX index-name FOR TEXT"
```
- Call the SYSPROC.SYSTS\_DROP stored procedure:  

```
CALL SYSPROC.SYSTS_DROP(' index-schema', ' index-name', ' locale', ?)
```

Where *locale* is the five-character locale code, such as en\_US, that specifies the language in which messages will be written to the log file.

## What to do next

**Note:** If any orphaned collections exist after you drop a text search index, you can remove them using the Administration Tool.

If, after dropping a text search index, you plan to create a new one on the same text column, you must first disconnect from and then reconnect to the database.

## Sample: Scheduling a DB2 Text Search index update

Schedule a DB2 Text Search index update and verify execution result.

### Before you begin

Complete the following tasks before you start any scheduler jobs:

1. Set the `ATS_ENABLE` registry variable
2. Check that the `SYSTOOLSPACE` table space exists
3. Ensure that the database is activated

For details about the prerequisites for scheduling a DB2 Text Search index update, see the topic about setting up the administrative task scheduler.

### About this task

Create a scheduler task using the DB2 Scheduler and execute the task in the specified frequency.

### Procedure

1. Create a text search index and specify the update frequency.

```
db2ts "create index simix for text on simple(comment)
update frequency (D(*) H(*) M(30))"
```

2. Connect to your database.

```
db2 connect to testdb
```

3. Find the scheduler task name

```
db2 "select indexidentifier from sysibmts.tsindexes"
```

For the following steps, let's assume the numeric part of the index identifier is 12345. So, the scheduler name is `TSSCH_12345`.

4. Find the scheduler task in the `SYSTOOLS.ADMIN_TASK_LIST` administrative view.

```
db2 "select * from systools.admin_task_list"
```

5. Verify text index update status.

```
db2 "select * from sysibmts.tsevent_123456"
```

6. If no message is shown, but data was available for an update, verify that the scheduler task was started.

```
db2 "select * from systools.admin_task_status"
```

Otherwise, use the scheduler task name to restrict the `SELECT` operation to the data belonging to the new scheduler task for the example shown previously:

```
db2 "select * from systools.admin_task_status
where name = 'TSSCH_12345'"
```

---

## Chapter 8. Searching with text search indexes

After you populate a text search index with data, you can search that index. DB2 Text Search supports searches in SQL, XQuery, and SQL/XML.

You can use the following search functions:

- The SQL function CONTAINS and the XML function xmlcolumn-contains, to create queries for specific words or phrases
- The SQL function SCORE, to obtain the relevancy of a found text document

Searches on text search indexes can range from the simple, such as queries for the occurrence of a single word in a title, to the complex, such as queries that use Boolean operators or term boosting. In addition to the operators that allow you to refine the complexity of your search, features such as synonym dictionaries and linguistic support can enhance searches on text search indexes.

---

### Search functions for DB2 Text Search

After you update a text search index, you can search using the CONTAINS or SCORE SQL scalar search function or using the xmlcolumn-contains function.

Searches on text search indexes can range from the simple, such as queries for the occurrence of a single word in a title, to the complex, such as queries that use Boolean operators or term boosting. In addition to the operators that allow you to refine the complexity of your search, features such as synonym dictionaries and linguistic support can enhance searches on text search indexes.

You can use the following search functions:

- The SQL function CONTAINS and the XML function xmlcolumn-contains, to create queries for specific words or phrases
- The SQL function SCORE, to obtain the relevancy of a found text document

The scalar text search functions, CONTAINS and SCORE, are seamlessly integrated within SQL. You can use the search functions in the same places that you would use standard SQL expressions within SQL queries. The SQL SCORE scalar function returns an indicator of how well the text documents matched a given text search condition. The SELECT phrase of the SQL query determines which information is returned to you.

The CONTAINS function searches for matches of a word or phrase and can be used with wildcard characters to search for substring matches in a manner similar to the SQL LIKE predicate and can search for exact string matches in a manner similar to the SQL = operator. However, there are key distinctions between using the CONTAINS function and using the SQL LIKE predicate or the = operator. The LIKE predicate and the = operator search for patterns in a document, while CONTAINS uses linguistic processing: that is, it searches for different forms of the search term. For example, even without using wildcard characters, searches for the term work also return documents containing working and worked. Moreover, you can add a synonym dictionary to the text search index, increasing the scope of a search. For example, you can group laptop and ThinkPad together so they are

returned from searches for notebook computers. For XML documents, the XML search argument syntax allows you to search for text inside tags and attributes. As well, XQuery searches are case sensitive.

Note that the DB2 optimizer estimates how many text documents can be expected to match a CONTAINS predicate and how costly different access plan alternatives will be. The optimizer chooses the cheapest access plan.

The function `xmlcolumn-contains` is a built-in DB2 function that returns XML documents from a DB2 XML data column based on a text search performed by the DB2 Text Search engine. You can use `xmlcolumn-contains` in XQuery expressions to retrieve documents based on a search of specific document elements. For example, if your XML documents contain product descriptions and prices for toys that you sell, you can use `xmlcolumn-contains` in an XQuery expression to search the description and price elements and return only the documents that have the term `outdoors` but not `pool` and cost less than \$25.00.

There are key distinctions between using the `xmlcolumn-contains` function and the XQuery `contains` function. The XQuery `contains` function searches for a substring inside a string; it looks for an exact match of the search term or phrase. The XQuery `xmlcolumn-contains` function, however, has similar functionality to the CONTAINS function, except that it operates on XML columns only. As well, it returns XML documents containing the search term or phrase, whereas `contains` returns only a value such as 1, 0, or NULL to indicate whether the search term was found.

---

## Full-text search methods

You can use an SQL statement or XQuery to search through text search indexes.

### Procedure

To search a text search index for a specific term or phrase, use one of the following methods:

- Search with SQL.

To search a text search index for a specific term or phrase with an SQL statement, use the CONTAINS function as follows:

```
db2 "SELECT column-name FROM table-name
WHERE CONTAINS (...) = 1"
```

For example, the following query searches the PRODUCT table for the names and prices of various snow shovels:

```
db2 "SELECT NAME, PRICE FROM PRODUCT
WHERE CONTAINS (NAME, 'snow shovel') = 1"
```

- Search with XQuery.

To search a text search index for a specific term or phrase using XQuery, use the `db2-fn:xmlcolumn-contains()` function.

For example, the following query searches the PRODUCT table for the names and prices of various snow shovels:

```
db2 "xquery for \${info} in db2-fn:xmlcolumn-contains
('PRODUCT.DESCRPTION','snow shovel')
return <result> {\${info}/description/name, \${info}/description/price} </result>"
```

**Note:** Depending on the operating system shell that you are using, you might need a different escape character in front of the dollar sign of the variable information. The previous example uses the backward slash ( \ ) as an escape character for UNIX operating systems.

## Basic search

You can use boolean operators and modifiers in your search queries. The more specific the search term that you use, the more precise the results.

### Example

Example 1: Searches for documents that contain the terms 'wizard' and 'dragon'. The default operator is AND if there is no explicit boolean operator specified.

```
select title from books where contains(story, 'dragon wizard')=1
```

Example 2: Searches for documents that contain the phrase 'dragon wizard'. It will not include documents that contain for example, the term 'dragons'.

```
select title from books where contains(story, "dragon wizard")=1
```

Example 3: Searches for documents that contain the term 'dragon' and optionally the term 'wizard'. Documents that contain both terms will receive a higher score.

```
select title from books where contains(story, 'dragon ?wizard')=1
```

Example 4: Searches for documents that contain the terms 'dragon' or 'wizard', but not the term 'hobbit'.

```
select title from books where contains(story, '(dragon OR wizard) NOT hobbit')=1
```

## Fuzzy search

A fuzzy search is used to find documents that contain words with similar spelling to the search term.

To start a fuzzy search use the tilde (~) symbol at the end of a phrase and specify the degree of fuzziness as a fraction between 0 and 1. The lower the value of the fraction, then the greater the degree of fuzziness. Special characters are not supported in fuzzy search queries.

### Example

Step 1. Create a table called BOOKS:

```
create table books (  
    isbn varchar(18) not null primary key,  
    author varchar(30),  
    story varchar(100),  
    year integer);
```

Step 2. Create a text search index on the STORY column:

```
db2ts "create index bookidx for text on books(story) connect to test";
```

Step 3. Import data into the table:

```
insert into books values ('0-13-086755-1','John','The Blue Can',2001)  
insert into books values ('0-13-086755-2','Mike','Cats and Dogs', 2000)  
insert into books values ('0-13-086755-3','Peter','Hats on the Rack',1999)  
insert into books values ('0-13-086755-4','Agatha','Cat among the Pigeons',1997)  
insert into books values ('0-13-086755-5','Edgar','Cars Unlimited',2010)  
insert into books values ('0-13-086755-6','Roy','Carson and Lemon',2008)
```

Step 4. Update the text search index:

```
db2ts "update index bookidx for text connect to test"
```

Step 5. Issue a fuzzy search with the CONTAINS function:

```
select author, year, story from books where contains(story, 'cat~0.4') = 1
```

The following is the sample output:

```
AUTHOR YEAR STORY
-----
John 2001 The Blue Can
Mike 2000 Cats and Dogs
Agatha 1997 Cat among the Pigeons
```

3 record(s) selected.

To see the associated score, issue the following query which has been modified for increased fuzziness:

```
select author, year, story, integer(score(story, 'cat~0.3')*1000) as score
from books where contains(story, 'cat~0.3') = 1 order by score desc
```

The following is the sample output:

```
AUTHOR YEAR STORY SCORE
-----
Agatha 1997 Cat among the Pigeons 32
John 2001 The Blue Can 17
Mike 2000 Cats and Dogs 17
Peter 1999 Hats on the Rack 1
Edgar 2010 Cars Unlimited 1
```

5 record(s) selected.

## Proximity search

A proximity search retrieves documents that contain search words which are located within a specified distance from each other.

To start a proximity search use the tilde (~) symbol at the end of a phrase and specify the distance in words as a valid integer number. When determining the distance consider that sentence breaks count as 10 position increments. Special characters are not supported in proximity search queries.

### Example

Step 1. Create table called BOOKS:

```
create table books (
    isbn varchar(18) not null primary key,
    author varchar(30),
    story varchar(100),
    year integer);
```

Step 2. Create text search index on the STORY column:

```
db2ts "create index bookidx for text on books(story) connect to test";
```

Step 3. Import data into the table:

```
insert into books values ('0-13-086755-1','John','Understanding Astronomy.',
,2001)
insert into books values ('0-13-086755-2','Mike','The cat hunts some mice.'
,2000)
insert into books values ('0-13-086755-3','Peter','Some men were standing
```

```

beside the table.',1999)
insert into books values ('0-13-086755-4','Astrid','The outstanding
adventure of Pippi Longst.',1997)
insert into books values ('0-13-086755-6','Agatha','Cat among the pigeons'
,2004)
insert into books values ('0-13-086755-7','John','Pigeons land in the square
, and a cat plays with a ball',2001)
insert into books values ('0-13-086755-8','Sam','Pigeon on the roof',2007)

```

Step 4. Update the text search index:

```
db2ts "update index bookidx for text connect to test"
```

Issue a proximity search for the terms cat and pigeon within 4 words of each other in a document and use the following search syntax within the DB2 Text Search CONTAINS phrase:

```

select author, year, substr(story,1,30) as title from books
where contains(story, '"cat pigeon"~4') = 1

```

## Searching for special characters

Special characters, for example, common punctuation characters, are indexed as part of a text index update, and you can search for special characters like other query terms.

To find a special character in a document, include the special character in the query expression. In some cases, escaping special characters is required.

### Escaping special characters

Special characters can serve different functions in the query syntax.

To search for a special character that has a special function in the query syntax, you must escape the special character by adding a backslash before it, for example:

- To search for the string “where?”, escape the question mark as follows:  
“where\?”
- To search for the string “c:\temp,” escape the colon and backslash as follows:  
“c:\\temp”

Not escaping such special characters can result in syntax errors.

*Table 3. Special characters that must be escaped to be searched*

Special character	Notes on behavior when not escaped
Ampersand (&)	
Asterisk (*)	Used as a wildcard character.
At sign (@)	A syntax error is generated when an at sign is the first character of a query. In xmlxp expressions, the at sign is used to refer to an attribute.
Brackets [ ]	Used in xmlxp expressions to search the contents of elements and attributes.
Braces { }	Generates a syntax error.
Backslash (\)	
Caret (^)	Used for weighting (boosting) terms.
Colon (:)	Used to search in the contents of fields.
Equal sign (=)	Generates a syntax error.

Table 3. Special characters that must be escaped to be searched (continued)

Special character	Notes on behavior when not escaped
Exclamation point (!)	A syntax error is returned when an exclamation point is the first character of a query.
Forward slash (/)	In xpath expressions, a forward slash is used as an element path separator.
Greater than symbol (>) Less than symbol (<)	Used in xpath expressions to compare the value of an attribute. Otherwise, these characters generate syntax errors.
Minus sign (-)	When a minus sign is the first character of a term, only documents that do not contain the term are returned.
Parentheses ( )	Used for grouping.
Percent sign (%)	Specifies that a search term is optional.
Plus sign (+)	
Question mark (?)	Handled as a wildcard character.
Semicolon (;)	
Single quotation mark (')	Single quotation marks are used to contain xpath expressions.
Tilde (~)	Handled as proximity and fuzzy search operators.
Vertical bar ( )	

Escaping special characters that do not serve a special function in the query syntax is optional. The following table shows some examples of special characters that do not require escaping.

Table 4. Examples of special characters that do not require escaping

Special character	Notes
Comma (,)	
Dollar sign (\$)	
Period (.)	In xpath expressions, a period is used to search the content of elements.
Pound sign (#)	
Underscore (_)	

### Special characters adjacent to query terms

When a special character is adjacent to a word in a query, documents that contain the special character and word in the same order are returned.

For example, searching for "30\$" finds documents that contain "30\$", but does not find documents that contain "\$30". However, searching for "30 \$" (with a space) finds all documents that contain "30" and "\$" anywhere in the documents including both "30\$" and "\$30".



When a special character is adjacent to a stop word in a query, the stop word is not removed from the query. For example, searching for "at&t" does not remove the stop word "at". However, searching for "at & t" with spaces removes the stop word "at".

When a special character separates two words, the sequence of tokens is searched as a sequence. For example, searching for "jack\_jones" finds documents that contain "jack\_jones" but not documents that contain "jack\_and\_jones".

Words that are adjacent to special characters are lemmatized. For example, searching for "cats&dogs" in English finds documents that contain "cat&dog".

You can use special characters in wildcard search expressions. For example, searching for "ja\*\_" finds documents that contain "jack\_jones". However, you cannot use wildcard characters to find special characters. For example, searching for ca\*s finds documents that contain cats, categories, cast-members, or cas, but not documents that contain ca\_s.

### **Indexing special characters**

During tokenization and language processing, DB2 Text Search identifies and indexes special characters as punctuation.

Special characters are token delimiters. For example, "jack\_jones" is tokenized as three separate tokens: "jack", "\_", and "jones". Emails, URLs, and file paths are broken down into tokens. For example:

- Jack\_jones@ibm.com is tokenized as jack \_ jones @ ibm . com
- http://www.ibm.com is tokenized as http :// www . ibm . com

Special characters do not occupy a token position in the file. For example, "jack\_jones" is indexed with the underscore in the same token position as "jack". Special characters also do not occupy a token position when spaces are included. For example, "jack\_jones" is indexed in the same way as "jack \_ jones".

The token position is used for exact phrase search and for proximity search. For example, if a document contains the expression jack\_jones, searching for the exact phrase "'jack jones'" finds this document.

When a sequence of special characters are indexed separately, they are searched in no particular order. For example, searching for "#\$" also finds documents that contain "\$#".

### **Special characters in CJK languages**

To find a sequence of characters that includes special characters in a Chinese, Japanese or Korean (CJK) document, the query expression must include the special characters. This is different to non-CJK languages, where a whitespace can substitute for a special character.

If a document is mixed language, for example, a Chinese language document contains some English terms, a search with whitespace will still substitute for special characters for the non-CJK terms.

For example, if an indexed document contains john\_smith, you can search for john\_smith or "john smith" (exact match, without the underscore) and both queries return the document that contains john\_smith.

**Note:** The characters '?', '\*', and '\' have semantic meaning as wildcards and escape character, but are not searchable as special characters.

## Structural full-text search in XML documents

DB2 Text Search supports using XML search for searching XML documents.

By using a subset of the XPath language with extensions for text search, XML search allows you to index and search XML documents. You can use structural elements (tag names, attribute names, and attribute values) separately or combine them with free text in queries.

The following list highlights the key features of XML search:

### XML structural search

By using XML search syntax in text search queries, you can search XML documents for structural elements (tag names, attribute names, and attribute values) and text scoped by those elements. Note that plain searches do not search the attribute field in an XML document.

### XML query tokenization

The text used in the XML search predicate expression as XML query terms is tokenized the same way that text in non-XML query terms is tokenized, except that spelling corrections, fielded terms, and nested XML search terms are unsupported. Synonyms, wildcard characters, phrases, and lemmatization are supported.

### Disregarding of XML namespaces

Namespace prefixes are not retained in the indexing of XML tag and attribute names. You can index and search XML documents declaring and using namespaces, but namespace prefixes are discarded during indexing and removed from XML search queries.

### Numeric values

Predicates comparing attribute values to numbers are supported.

### Complete match

The operator = (equal sign) with a string argument in a predicate means that a complete match of all tokens in the string with all tokens in the identified text span is required, with the order being significant.

The subset of XPath that is implemented in XML search differs from standard XPath in the following ways:

- It does not support iteration and ranges in path expressions.
- It eliminates filter expressions: that is, it allows filtering only in the predicate expression, not in the path expression.
- It disallows absolute path names in predicate expressions.
- It implements only one axis (tag) and allows propagation only in the forward direction.

The following table lists some valid XML search queries.

*Table 5. Valid XML search queries*

Query	Description
/	The root node; any document
/sentences	Any document with a top-level tag of sentences

Table 5. Valid XML search queries (continued)

Query	Description
//sentences	Any document with a tag at any level of sentences
sentences	Any document with a tag at any level of sentences
/sentence/paragraph	Any document with a top-level tag of sentences having a direct child tag of paragraph
/sentence/paragraph/	Any document with a top-level tag of sentences having a direct child tag of paragraph
/book/@author	Any document with a top-level book tag having an attribute author
/book//@author	Any document with a top-level book tag having a descendant tag at any level with attribute author
/book[@author contains("barnes") and @title contains("lemon")]	Any document with a top-level book tag with the attributes author and title with values that contain the normalized strings shown
/book[@author contains("barnes") and (@title contains("lemon") or @title contains("flaubert"))]	Any document with a top-level book tag with the specified author attribute and either of the two specified title attributes
/program[. contains("hello, world.")]	Any document with a top-level program tag containing at least the tokens hello and world
/book[paragraph contains("flaubert")]/sentence	Any document with a top-level tag book tag with a direct child tag of paragraph containing "flaubert" and, referring to the book tag, having a descendant tag sentence at any level
/auto[@price <30000]	Any document with a top-level auto tag having an attribute price with a numeric value that is less than 30000
//microbe[@size <3.0e-06]	Any document containing a microbe tag at any level with a size attribute with a value that is less than 3.0e-06

**Note:** The following are unsupported in the XML search syntax:

- /\*
- //\*
- /@\*
- //@\*

A *plain* search does not search the attribute field in the XML document.

## Searching text search indexes using SCORE

You can use the SCORE function to find out the extent to which a document matches a search argument.

## About this task

SCORE returns a double-precision floating-point number between 0 and 1 that indicates how well a document meets the search criteria. The better a document matches the query, the more relevant the score and the larger the result value.

The score is calculated dynamically based on the content of a text index collection at the time of the query and is therefore only meaningful for a non-partitioned text index.

Scoring algorithms may differ for different text index formats or query types. Note that deleted documents impact the relative value returned by SCORE until they are removed from the text search index. However, significant differences in scores would be observed only when large chunks of data have been deleted from the index.

## Example

To search in the SAMPLE database for the number of employees who indicated on their resumes that they know how to program in Java or COBOL, you can issue the following query:

```
SELECT EMPNO, INTEGER(SCORE(RESUME, 'programmer AND (java OR cobol)') * 100)
AS RELEVANCE FROM EMP_RESUME WHERE RESUME_FORMAT = 'ascii'
ORDER BY RELEVANCE DESC
```

However, the following query using CONTAINS is superior. The DB2 optimizer evaluates the CONTAINS predicate in the WHERE clause first and thereby avoids evaluating the SCORE function in the SELECT list for every row of the table. Note that this is possible only if the SCORE and CONTAINS arguments in the query are identical.

```
SELECT EMPNO, INTEGER(SCORE(RESUME, 'programmer AND (java OR cobol)') * 100)
AS RELEVANCE FROM EMP_RESUME WHERE RESUME_FORMAT = 'ascii'
AND CONTAINS(RESUME, 'programmer AND (java OR cobol)') = 1
ORDER BY RELEVANCE DESC
```

---

## DB2 Text Search argument syntax

A search argument comprises one or more terms and optional search parameters, separated by white space, that you specify to search text documents.

When you specify a term, the search engine returns documents that contain that term and, by default, variations on that term. For example, if you search by using the term king, documents containing king and kings are returned. If you search by using multiple terms, the search engine returns only documents containing all the terms.

If you want to search by using an exact phrase, surround the phrase in quotation marks. Use a fuzzy search to find documents that contain words with spelling similar to that of the search term. A common reason to perform a fuzzy search is to include documents that contain misspellings in the search result.

Perform a proximity search to retrieve documents containing search words that are located within a specified distance from each other.

**Remember:**

- Searches are not case sensitive, so a search in Spanish for the exact term "DOS" might return documents containing DOS or dos.
- Text search queries must not exceed DB2 SQL query limits.

The more specific the search term that you use, the more precise the results. However, you can also refine your searches by using options such as the following ones:

#### **Boolean operators**

Use the AND operator to search for documents that contain all the specified terms. The AND operator is the default conjunction operator. If there is no logical operator between the two terms, AND is used.

Use the OR operator to search for documents that contain all the specified terms. The OR operator links the two or more terms and finds a matching document if either of the terms exists in a document.

#### **Occurrence modifiers**

Use a plus sign (+) to specify that terms are required. The plus sign (+) modifier is distinct from the AND operator because the plus sign (+) modifier indicates that the second term must be an exact match. No synonym is used.

Use a minus sign (-) or the NOT modifier to specify that terms are prohibited.

#### **The boost modifier**

Use the caret (^) character to give higher importance to occurrences of a particular term. The caret (^) character provides a *boost* to the term or phrase that precedes it when the specified number is greater than 1. If you want to reduce the ranking of the term or phrase in the returned list, specify a number that is greater than 0 but less than 1.

Use the boost modifier with the SCORE function or the ORDER BY clause.

#### **Wildcard characters**

Use a question mark (?) to specify that a single character can be added to your search term. Use an asterisk (\*) to specify that any number of characters can be added to your search term. Use these wildcard characters to search terms and data for spelling variations and increase search scope.

**Important:** Using the asterisk (\*) wildcard at the beginning of a search term negatively affects the performance of the search query.

Wildcard searches with an asterisk (\*) apply a term expansion to find documents. If the number of matching terms in the text index collection exceeds the expansion limit, only a subset of documents that match the criteria is returned. See the text search arguments topic for further details. Also, wildcard searches find regular characters, not special characters. For example, searching for US-\*-abc finds strings such as US-xxx-abc, US-x-abc, and US-x#-abc but not US-#-abc.

#### **The percentage sign (%)**

Use a percentage sign (%) to specify that a term or phrase is optional.

#### **The backslash (\) escape character**

Use a backslash (\) to include special characters in your search. All of the following characters are special characters in text search queries:

- <
- >

- &&
- | |
- !
- (
- )
- %
- =
- "
- {
- }
- ~
- \*
- ?
- [
- ]
- :
- \
- -

#### Double quotation marks (")

Use quotation marks (") around your search term or phrase to have only exact matches returned.

#### Parentheses

Use parentheses to have search terms and the relationship between them treated as a single item.

For XML search queries that are sent to the XML parser, write the queries by using opaque terms in a subset of the XPath language. The query parser recognizes an opaque term by the syntax that you use in the query.

For any language-specific processing during a search, a locale is assumed for the search-argument parameter. This query language is the locale of the text search index that is used when you perform the search function.

The search argument syntax is as follows:

*Search argument*

**QualifiedClause** ((Operator) (QualifiedClause))

*Operator*

AND | OR

*QualifiedClause*

(Modifier) **Clause** (^number)

*Modifier*

+ | - | NOT

*Clause* Unqualified term | opaque term

- An unqualified term is a term or a phrase. A term can be a word, such as king; an exact word, such as "king"; or a word that includes a wildcard, such as king\* or king?. Similarly, a phrase can be a group of

words, such as cabbages and kings; an exact phrase, such as "The King and I"; or a phrase that includes a wildcard, such as all the king's h\* or all the kin?'s horses.

- An opaque query term is not parsed by the linguistic query parser; opaque terms are identified by their syntax. The opaque term that is used for text search queries is @xpath, for example, @xpath: '/TagA/TagB[. contains("king")]'

## Examples

Table 6. Boolean operators

Operator	Example	Query results
AND King AND Lear King Lear	Returns documents that contain the terms King and Lear. If you enable a synonym dictionary, words such as monarch can also be returned.	
OR	Hamlet OR Othello	Returns documents that contain either Hamlet or Othello.

Table 7. Occurrence modifiers

Modifier	Example	Query result
NOT -	Hamlet NOT Othello Hamlet -Othello	Returns documents that contain Hamlet but not Othello.
+	Lear + King	Returns documents that contain the terms Lear and King. Documents containing Lear and monarch are not returned.

Table 8. Other modifiers

Modifier	Example	Query result
<i>term1</i> or <i>phrase1</i> ^ <i>number</i> <i>term2</i> or <i>phrase2</i>	Hamlet^2 Othello Hamlet Othello^.5	Returns documents containing Hamlet and Othello but gives more importance to the term Hamlet. In both example queries, each occurrence of the term Hamlet is given twice as much importance as each occurrence of Othello is given.
*	king* k*ng *ing	Returns documents that contain possible combinations of the search term with the wildcard character. The example query might return results such as king and kingdom in the first example, king and kissing in the second example, and king and skiing in the third example.
*	www.*.com	Searching using wildcards does not return terms that contain special characters. The example query might return www.ibm.com but does not return www.#.com.

Table 8. Other modifiers (continued)

Modifier	Example	Query result
?	mea? be?n ?ean	Returns documents that contain possible combinations of the search term with the wildcard character. The first example returns results such as meal and mean, the second example returns results such as bean and been, and the third example returns results such as mean and bean.
%	King James %Edition	Returns documents that contain both king and james, but edition is an optional term.
"phrase" "exact term" "phrase with wildcard"	"King Lear" "king" "John * Kennedy" "John ? Kennedy"	Returns documents that contain the exact word or phrase. The first example returns King Lear. The second example returns the word king and no other forms, such as kings or kingly.  You can use quotation marks with wildcards. The third example returns occurrences of John Kennedy with or without various middle names or initials. The fourth example returns John <i>initial</i> Kennedy.
()	(Hamlet OR Othello) AND plays	Returns documents that contain the following terms: <ul style="list-style-type: none"> <li>• The term Hamlet or Othello</li> <li>• The term plays</li> </ul>
\	\(1+1\)\:2	Returns documents that contain (1+1):2. Use the backslash (\) character to escape special characters that are part of the query syntax.

## Search syntax for XML documents

Using an XML search expression, you can use the DB2 Text Search engine to search specific portions of an XML document in a DB2 XML column.

### Syntax

►► @xmlxp: ' XML search query ' ◀◀

### XML search query:

|—location-path—| [—search-predicate—]

### @xmlxp:

The keyword that starts a text search query on an XML document.

**Note:** The keyword @xpath has been deprecated.



### *XML search query*

A text search query used by DB2 Text Search to search XML documents. The query is enclosed in single quotation marks. The XML search query is an XML search expression that consists of a location path specifying the portion of the XML document to search and an optional predicate that specifies the search criteria.

### *location-path*

An XML search expression that uses a subset of the XPath abbreviated syntax to specify an XML document node or attribute. More information is provided in the "Location path" section.

### *search-predicate*

The optional search criteria used by DB2 Text Search when searching an XML document. More information is provided in the "Search predicate" section.

The DB2 Text Search engine returns the XML document if it finds the text specified in the *search-predicate* in the specified nodes or attributes of the XML document.

## **Location path**

When performing a text search on an XML document, DB2 Text Search uses local node and attribute names and a subset of the XPath syntax to specify nodes and attributes in an XML document. DB2 Text Search supports the following XML search elements:

- Local node or attribute names
- . (period) as the current context node
- / or // as the separator character
- @ as the abbreviated symbol for attribute
- Name normalization

XML node and attribute names are not normalized when they are indexed for use by the DB2 Text Search engine: they are not converted to lowercase, tokenized, or modified in any way. Case is significant in XML node and attribute names, so the strings that you use for them in queries must match exactly the names appearing in documents to get a match.

- Namespace handling

When creating a text search index, you can use XML documents that contain XML namespace specifiers, but namespace specifiers are not retained in the index. For example, the tag `<nsdoc:heading>` is indexed under `heading` only, and the query term `@xmlns:'/nsdoc:heading'` is parsed as `@xmlns:'/heading'`. XML namespace prefixes are discarded during query parsing.

## **Examples**

The following example is a valid text search query using XML search that searches for the term `snow shovel` in the description node of product information:

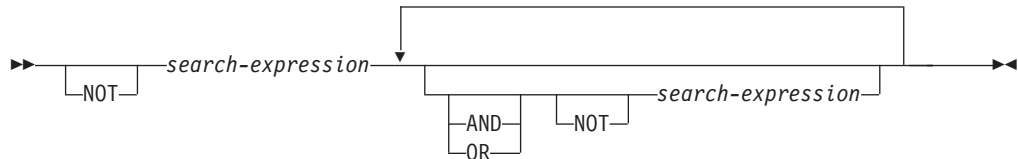
```
@xmlns:'/info/product/description[. contains("snow shovel")]'
```

The following example is not a valid text search query using XML search because it uses `..`, the XML search abbreviation for `parent::node()`:

```
@xmlns:'/info/product/description/..@ID[. contains("A2")]'
```

## Search predicate

### Syntax



#### *search-expression*

A DB2 Text Search XML search query. DB2 Text Search uses a search expression to search node or attribute values in an XML document.

You can use the following operators to create search expressions:

- Logical operators: AND, OR, and NOT
- Containment operators: contains and excludes
- Comparison operators: =, >, <, >=, <=, and !=

#### **Note:**

Comparison operators can be applied to attribute values only, not node values.

Thus, for the `<root><aaa id="10">100</aaa><aaa id="11">101</aaa></root>`, the following query is invalid:

```
select id from testtable where contains(item,'@xmlxp: '/root/aaa[. > 20]''')>0
```

An example of a valid query would be:

```
select id from testtable where contains(item,'@xmlxp: '/root/aaa/@id[. > 20]''')>0
```

You can combine the comparison and containment operators with the logical operators AND, OR and NOT to create complex search expressions. You can also use parentheses to group expressions.

Use single or double quotation marks to enclose a string. A string that contains quotation marks cannot be enclosed by the same type of quotation marks. For example, a string enclosed in single quotation marks cannot contain a single quotation mark.

In XML search predicates, comparison operators take precedence over logical operators, and all logical operators have the same precedence. You can use parentheses to ensure intended evaluation precedence.

Free text in XML documents (text between tags, not inside a tag itself) and attribute values are normalized before indexing. Free text in XML queries (in containment operators) is normalized in the same way that it is in non-XML queries.

### Example

The following example uses an XML search query to search for products that contain the term snow shovel in the product description and that have a price lower than \$29.99.

```
@xmlxp:'/info/product [(description contains("snow shovel"))
and (@price < 29.99)]'
```

## Comparison expressions

Comparison expressions compare the value of an attribute with a specified value.

## Syntax

► *path-expression*—*operator*—*literal* ◄

### *path-expression*

The path expression using a subset of the XML search abbreviated syntax to specify a node or attribute.

### *operator*

The type of comparison to perform. The operator can be one of the following types:

- = *path-expression* value is equal to *literal*.
- > *path-expression* value is greater than *literal*.
- < *path-expression* value is less than *literal*.
- >= *path-expression* value is greater than or equal to *literal*.
- <= *path-expression* value is less than or equal to *literal*.
- != *path-expression* value is not equal to *literal*.

### *literal*

A string or number used to compare against the *path-expression* node or attribute value.

Enclose the string in single or double quotation marks. A string that contains quotation marks cannot be enclosed by the same type of quotation marks. For example, a string enclosed in single quotation marks cannot contain a single quotation mark. Use the backslash character (\) to escape double quotation marks (").

If the string contains double quotation marks, you can enclose the string in single quotation marks. The following example shows a string containing double quotation marks enclosed in single quotation marks:

```
'he said "Hello, World"'
```

If the a string contains single quotation marks, you can enclose the string in escaped double quotation marks. The following example shows a string containing a single quotation mark enclosed in double quotation marks:

```
\ "the cat's toy\"
```

DB2 Text Search features such as phrases, wildcards, and synonyms are not supported in XML search queries.

## Example

The following example uses the = operator to find product IDs equal to the string 100-200-101:

```
@xmlxp:'/info/product/@pid[. = "100-200-101" ]'
```

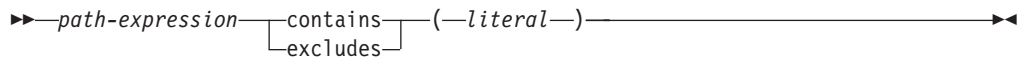
### Note:

The only comparison operators that are supported with string arguments are = and !=, so <, <=, >, >= cannot be used. All six operators are supported with numeric arguments. Numeric arguments are supported for comparison to attribute values, but not to tag (node) content

### Containment expressions

Containment expressions determine whether the value of a node or an attribute contains a specified value.

### Syntax



#### *path-expression*

The XML search expression that specifies an XML node or attribute.

#### **contains**

An expression that specifies that *path-expression* value contains *literal*.

#### **excludes**

An expression that specifies that *path-expression* value excludes *literal*.

#### *literal*

A string used to compare against the *path-expression* node or attribute value.

Use single or double quotation marks to enclose a string. A string cannot contain enclosing quote type: for example, a string enclosed in single quotation marks cannot contain a single quotation mark. Use the backslash character (\) to escape double quotation marks (").

If the string contains double quotation marks, you can enclose the string in single quotation marks.

The following example shows a string containing double quotation marks enclosed in single quotation marks:

```
'he said "Hello, World"'
```

If the string contains single quotation marks, you can enclose the string in escaped double quotation marks. The following example shows a string containing a single quotation mark enclosed in double quotation marks:

```
\ "the cat's toy\"
```

### Example

The following example uses the XQuery abbreviated syntax for path expressions to specify that the description node excludes the term ice scraper:

```
@xmlxp:'/info/product/description[. excludes('ice scraper')]'
```

---

## Enhancing performance for full-text queries

To enhance performance during searches, use one or more of the following approaches:

### Procedure

- Use the EXPLAIN statement to check the access plan of the DB2 optimizer when searching with SQL.

- Avoid using the SCORE function without the CONTAINS function. Also, to avoid duplicate processing, ensure that the string (that is, the search argument and any search options) that you specify for the CONTAINS function exactly matches the string (including white spaces) that you use for the SCORE function.
- Ensure that the DB2 compiler has the correct table statistics. Use the **RUNSTATS** command to update the statistics.
- Review the database **STMT\_CONC** configuration parameter. When the parameter is set to use the LITERAL option, a performance degradation may occur for text search queries.



---

## Chapter 9. SQL and XML built-in search functions

You can use the following DB2 built-in search functions in DB2 Text Search. The schema of these functions is SYSIBM.

### CONTAINS

Returns a NULL or an INTEGER value of 0 or 1 depending on whether the input text document matches the text search condition

### SCORE

Returns a NULL or a DOUBLE value between 0 and 1 indicating the extent to which the text document meets the search criteria.

### xmlcolumn-contains

Returns a NULL or an INTEGER value 1 or 0 depending on whether the input text document of XML data type matches the text search condition

---

## CONTAINS function

The CONTAINS function searches a text search index using criteria that you specify in a search argument and returns a value that indicates whether a match is found.

### Function syntax

```
▶▶ CONTAINS ( (column-name) , (search-argument) [ (1) ] , (string-constant) )
```

### Notes:

- 1 *string-constant* must conform to the rules for search-argument-options.

### search-argument-options:

```
(1)
|-----|
| QUERYLANGUAGE == locale |
| RESULTLIMIT == value   |
| SYNONYM == OFF         |
|                       |
| SYNONYM == ON          |
```

### Notes:

- 1 You cannot specify the same clause more than once.

The schema is SYSIBM.

### Function parameters

#### *column-name*

A qualified or unqualified name of a column that has a text search index that is to be searched. The column must exist in the table or view

identified in the FROM clause in the statement and the column of the table, or the column of the underlying base table of the view, must have an associated text search index (SQLSTATE 38H12). The underlying expression of the column of a view must be a simple column reference to the column of an underlying table, either directly or through another, nested view.

*search-argument*

An expression that returns a value that is a string value (except a LOB) that contains the terms to be searched for and is not all blanks or the empty string (SQLSTATE 42815). The string value that results from the expression should be less than or equal to 4096 bytes (SQLSTATE 42815). The value is converted to Unicode before it is used to search the text search index. The maximum number of terms per query must not exceed 1024 (SQLSTATE 38H10).

*string-constant*

A string constant that specifies the search argument options that are in effect for the function.

The options that you can specify as part of the *search-argument-options* are as follows:

**QUERYLANGUAGE = locale**

Specifies the locale that the DB2 Text Search engine uses when performing a text search on a DB2 text column. The value can be any of the supported locales. If you do not specify **QUERYLANGUAGE**, the default is the locale of the text search index. If the **LANGUAGE** parameter of the text search index is **AUTO**, the default value for **QUERYLANGUAGE** is **en\_US**.

**RESULTLIMIT = value**

If the optimizer chooses a plan that calls the search engine for each row of the result set to obtain the **SCORE**, then the **RESULTLIMIT** option has no effect on performance. However, if the search engine is called once for the entire result set, **RESULTLIMIT** acts like a **FETCH FIRST** clause.

When using multiple text searches that specify **RESULTLIMIT** in the same query, use the same *search-argument*. If you use different *search-argument* values, you might not receive the results that you expect.

For partitioned text indexes, the result limit is applied to each partition separately.

**SYNONYM = OFF | ON**

Specifies whether to use a synonym dictionary that is associated with the text search index. The default is **OFF**. To use synonyms, add the synonym dictionary to the text search index using the Synonym Tool.

**OFF** Do not use a synonym dictionary.

**ON** Use the synonym dictionary associated with the text search index.

The result of the function is a large integer. If the second argument can be null, the result can be null; if the second argument is null, the result is null. If the third argument is null, the result is as if you did not specify the third argument.



CONTAINS returns the integer value 1 if the document contains a match for the criteria specified in the search argument. Otherwise, it returns 0.

CONTAINS is a non-deterministic function.

**Note:** You must take additional steps when using parameter markers as a search argument inside the text search functions. Parameter markers do not have a type when precompiled in JDBC and ODBC programs, but the search argument in the text search functions must resolve to a string value. Because the unknown type of the parameter marker cannot be resolved to a string value (SQLCODE -418), you must explicitly cast the parameter marker to the VARCHAR data type.

## Examples

- The following query is used to find all of the employees who have COBOL in their resumes. The text search argument is not case-sensitive.

```
SELECT EMPNO
FROM EMP_RESUME
WHERE RESUME_FORMAT = 'ascii'
AND CONTAINS(RESUME, 'COBOL') = 1
```

- In the following C program, the exact term ate is searched for in the COMMENT column:

```
char search_arg[100]; /* input host variable */
...
EXEC SQL DECLARE C3 CURSOR FOR
SELECT CUSTKEY
FROM CUSTOMERS
WHERE CONTAINS(COMMENT, :search_arg) = 1
ORDER BY CUSTKEY;
strcpy(search_arg, "ate");
EXEC SQL OPEN C3;
...
```

- The following query is used to find any 10 students who wrote online essays that contain the phrase fossil fuel in Spanish, which is combustible fósil. A synonym dictionary was created for the associated text search index. Because only 10 students are needed, the query is optimized by using the **RESULTLIMIT** option to limit the number of results from the underlying text search server.

```
SELECT FIRSTNAME, LASTNAME
FROM STUDENT ESSAYS
WHERE CONTAINS(TERM_PAPER, 'combustible fósil',
'QUERYLANGUAGE= es_ES RESULTLIMIT = 10 SYNONYM=ON') = 1
```

---

## SCORE function

The SCORE function searches a text search index using criteria that you specify in a search argument and returns a relevance score that measures how well a document satisfies the query as compared with the other documents in the column.

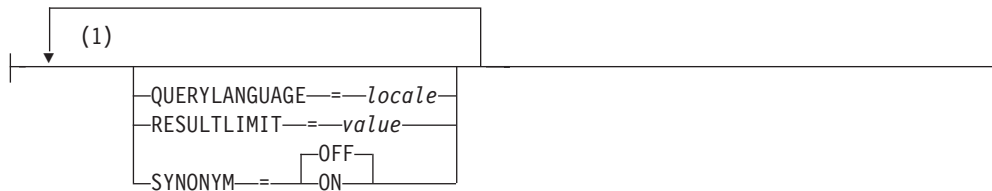
### Function syntax

```
►► SCORE ( (column-name) , (search-argument) (1) ) ►►
          , (string-constant)
```

#### Notes:

- 1 *string-constant* must conform to the rules for search-argument-options.

## search-argument-options:



### Notes:

- 1 You cannot specify the same clause more than once.

The schema is SYSIBM.

## Function parameters

### *column-name*

A qualified or unqualified name of a column that has a text search index that is to be searched. The column must exist in the table or view identified in the FROM clause in the statement and the column of the table, or the column of the underlying base table of the view, must have an associated text search index (SQLSTATE 38H12). The underlying expression of the column of a view must be a simple column reference to the column of an underlying table, either directly or through another, nested view.

### *search-argument*

An expression that returns a value that is a string value (except a LOB) that contains the terms to be searched for and is not all blanks or the empty string (SQLSTATE 42815). The string value that results from the expression should be less than or equal to 4096 bytes (SQLSTATE 42815). The value is converted to Unicode before it is used to search the text search index. The maximum number of terms per query must not exceed 1024 (SQLSTATE 38H10).

### *string-constant*

A string constant that specifies the search argument options that are in effect for the function.

The options that you can specify as part of the *search-argument-options* are as follows:

#### **QUERYLANGUAGE = locale**

Specifies the locale that the DB2 Text Search engine uses when performing a text search on a DB2 text column. The value can be any of the supported locales. If you do not specify **QUERYLANGUAGE**, the default is the locale of the text search index. If the **LANGUAGE** parameter of the text search index is **AUTO**, the default value for **QUERYLANGUAGE** is **en\_US**.

#### **RESULTLIMIT = value**

If the optimizer chooses a plan that calls the search engine for each row of the result set to obtain the **SCORE**, then the **RESULTLIMIT** option has no effect on performance. However, if the search engine is called once for the entire result set, **RESULTLIMIT** acts like a **FETCH FIRST** clause.

When using multiple text searches that specify **RESULTLIMIT** in the same query, use the same *search-argument*. If you use different *search-argument* values, you might not receive the results that you expect.

For partitioned text indexes, the result limit is applied to each partition separately.

**Note:** If the number of results is an issue, limit the number of results through a refinement of the search terms, rather than by using **RESULTLIMIT**. Because **RESULTLIMIT** returns at most the specified number of results with no consideration of their scores, the highest-ranking documents might not be included.

#### **SYNONYM = OFF | ON**

Specifies whether to use a synonym dictionary that is associated with the text search index. The default is OFF. To use synonyms, add the synonym dictionary to the text search index using the Synonym Tool.

**OFF** Do not use a synonym dictionary.

**ON** Use the synonym dictionary associated with the text search index.

The result of the function is a double-precision floating-point number. If the second argument can be null, the result can be null; if the second argument is null, the result is null. If the third argument is null, the result is as if you did not specify the third argument.

The result is greater than 0 but less than 1 if the column contains a match for the search criteria specified by the search argument. The more frequently a match is found, the larger the result value. If the column does not contain a match, the result is 0.

SCORE is a non-deterministic function.

**Note:** You must take additional steps when using parameter markers as a search argument inside the text search functions. Parameter markers do not have a type when precompiled in JDBC and ODBC programs, but the search argument in the text search functions must resolve to a string value. Because the unknown type of the parameter marker cannot be resolved to a string value (SQLCODE -418), you must explicitly cast the parameter marker to the VARCHAR data type.

### **Example**

- The following query is used to generate a list of employees in order of how well their resumes satisfy the query "programmer AND (java OR cobol)", along with a relevance value that is normalized between 0 and 100:

```
SELECT EMPNO,  
       INTEGER(SCORE(RESUME,  
                   'programmer AND (java OR cobol)') * 100) AS RELEVANCE  
FROM EMP_RESUME  
WHERE RESUME_FORMAT = 'ascii'  
      AND CONTAINS(RESUME, 'programmer AND (java OR cobol)') = 1  
ORDER BY RELEVANCE DESC
```

## Usage notes

- The SCORE value reflects a document's relative relevance when compared to the SCORE value of all documents from the same text index collection. For a partitioned database a text index may consist of multiple collections, however document scores are not normalized across partitions. Comparing or sorting SCORE values across text index collections is therefore not meaningful and does not provide a proper measure of relevance for documents in a partitioned text index.

---

## xmlcolumn-contains function

The db2-fn:xmlcolumn-contains function returns a sequence of XML documents from an XML data column based on a text search performed by the DB2 Text Search engine for specified search terms.

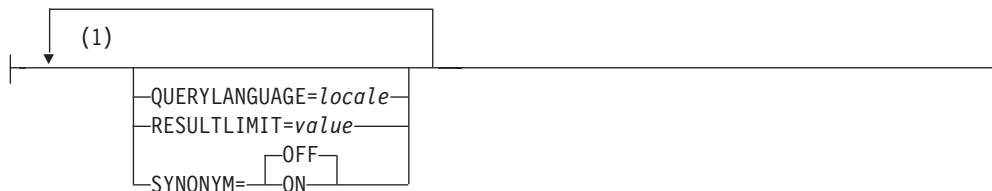
### Syntax

```
db2-fn:xmlcolumn-contains(string-literal,search-argument [options-string-literal](1))
```

#### Notes:

- 1 *options-string-literal* must conform to the rules for search-argument-options.

#### search-argument-options:



#### Notes:

- 1 You can specify each option only once.

#### *string-literal*

Specifies the name of a XML data type column to be searched by db2-fn:xmlcolumn-contains. The value of *string-literal* is case sensitive and must match the case of the table and column name. You must qualify the column name using a table name or a view name. The SQL schema name is optional. If you do not specify the SQL schema name, the value of CURRENT SCHEMA is used.

The column must have a text search index.

#### *search-argument*

An expression that returns an atomic string value or an empty sequence. The string cannot be all space characters or an empty string. The string must be castable to the type VARCHAR according to the rules of XMLCAST with a maximum length of 4096 bytes.

#### *options-string-literal*

Specifies the search argument options that are in effect for the function.

The options that you can specify as part of the *search-argument-options* are as follows:

**QUERYLANGUAGE = *locale***

Specifies the locale that the DB2 Text Search engine uses when performing a text search on a DB2 text column. The value can be any of the supported locales. If you do not specify **QUERYLANGUAGE**, the default is the locale of the text search index. If the **LANGUAGE** parameter of the text search index is **AUTO**, the default value for **QUERYLANGUAGE** is **en\_US**.

**RESULTLIMIT = *value***

If the optimizer chooses a plan that calls the search engine for each row of the result set to obtain the **SCORE**, then the **RESULTLIMIT** option has no effect on performance. However, if the search engine is called once for the entire result set, **RESULTLIMIT** acts like a **FETCH FIRST** clause.

When using multiple text searches that specify **RESULTLIMIT** in the same query, use the same *search-argument*. If you use different *search-argument* values, you might not receive the results that you expect.

For partitioned text indexes, the result limit is applied to each partition separately. For an example of what might happen when using multiple text searches and a solution, see the last example in “Examples” on page 126.

**SYNONYM = OFF | ON**

Specifies whether to use a synonym dictionary that is associated with the text search index. The default is **OFF**. To use synonyms, add the synonym dictionary to the text search index using the Synonym Tool.

**OFF** Do not use a synonym dictionary.

**ON** Use the synonym dictionary associated with the text search index.

## Returned values

The returned value is a sequence that is the concatenation of the non-null XML values from the column that is specified by *string-literal*. The non-null XML values are returned in a nondeterministic order. The XML values are the XML documents where the SQL **CONTAINS** function using *search-argument* for the column specified by *string-literal* would return 1. If there are no such XML values, an empty sequence is returned.

If *search-argument* is an empty sequence, an empty sequence is returned. If *search-argument* is an empty string or string containing all space characters, an error is returned. If the third argument is null, the result is as if you did not specify the third argument.

If the column that you specify using *string-literal* does not have a text search index, an error is returned.

The `db2-fn:xmlcolumn-contains` function is related to the `db2-fn:sqlquery` function, and both functions can produce the same result. However, the arguments of the two functions differ in case sensitivity. The first argument, *string-literal*, in the `db2-fn:xmlcolumn-contains` function is processed by XQuery and is case sensitive.

Because table names and column names in a DB2 database are uppercase by default, the first argument of `db2-fn:xmlcolumn-contains` is usually uppercase. The first argument of the `db2-fn:sqlquery` function is processed by SQL, which automatically converts identifiers to uppercase.

The following function calls are equivalent and return the same results assuming that the `PRODUCT` table is in the schema currently assigned to `CURRENT SCHEMA`:

```
db2-fn:xmlcolumn-contains("PRODUCT.DESCRPTION", "snow shovel")

db2-fn:sqlquery("select description from product
  where contains(description, 'snow shovel') = 1")
```

## Examples

The following examples use the DB2 Text Search engine to perform searches. The columns being searched are XML columns and have a text search index.

The first function searches for XML documents stored in the `PRODUCT.DESCRPTION` column that contain the words `snow` and `shovel`. The function sets the maximum number of returned documents to two. If the text search returns a large number of documents, you can optimize the search by using the **RESULTLIMIT** option to limit the maximum number of documents returned.

```
db2-fn:xmlcolumn-contains('PRODUCT.DESCRPTION', 'snow shovel', 'RESULTLIMIT=2')
```

The function returns the XML documents that match the search criteria. The documents might contain more than just a product description. For example, the following XML fragment consists of two product descriptions from an XML column. Each document contains a product description and information such as the product name, price, weight, and product ID.

```
<product xmlns="http://posample.org" pid="100-100-01">
  <description>
    <name>Snow Shovel, Basic 22 inch</name>
    <details>Basic Snow Shovel, 22 inches wide, straight handle with
      D-Grip</details>
    <price>9.99</price>
    <weight>1 kg</weight>
  </description>
</product>
<product xmlns="http://posample.org" pid="100-101-01">
  <description>
    <name>Snow Shovel, Deluxe 24 inch</name>
    <details>A Deluxe Snow Shovel, 24 inches wide, ergonomic curved handle
      with D-Grip</details>
    <price>19.99</price>
    <weight>2 kg</weight>
  </description>
</product>
```

The following function searches the XML column `STUDENT_ESSAYS.ABSTRACTS` for 10 student essays that contain the phrase `fossil fuel` in Spanish, which is combustible `fósil`. The function specifies `es_ES` (Spanish as spoken in Spain) as the language to use for the text search and uses the synonym dictionary that was created for the associated text search index. The function optimizes the search by using **RESULTLIMIT** to limit the number of results.

```
db2-fn:xmlcolumn-contains('STUDENT_ESSAYS.ABSTRACTS', '"combustible fosil"',
  'QUERYLANGUAGE=es_ES RESULTLIMIT=10 SYNONYM=ON')
```

The following example uses `db2-fn:xmlcolumn-contains` to find XML documents stored in the `PRODUCT.DESCRPTION` column that contain the word `ergonomic`. The expression returns the name of the product whose price is less than 20.

```
xquery
declare default element namespace "http://posample.org";
db2-fn:xmlcolumn-contains(
  'PRODUCT.DESCRPTION', 'ergonomic')/product/description[price < 20]/name
```

The previous expression returns only the name elements from the returned XML documents. For example, if the term `ergonomic` is in the product description of the product `Snow Shovel, Deluxe 24 inch`, the expression returns a name element similar to the following one:

```
<name xmlns="http://posample.org" >Snow Shovel, Deluxe 24 inch</name>
```

The following expression uses `db2-fn:xmlcolumn-contains` to find the XML documents from the `PRODUCT.DESCRPTION` column that contain the words `ice` and `scraper`. The expression uses the product IDs from the product descriptions to find purchase orders in the `PURCHASEORDER` table that contain the product IDs. The expression returns the customer IDs from purchase orders that contain the product IDs from the matched XML description documents.

```
xquery
declare default element namespace "http://posample.org";
for $po in db2-fn:sqlquery('
  select XMLElement(Name "po", XMLElement(Name "custid", purchaseorder.custid),
    XMLElement(Name "porder", purchaseorder.porder))
  from purchaseorder')
let $product := db2-fn:xmlcolumn-contains('PRODUCT.DESCRPTION',
  'ice scraper')/product
where $product/@pid = $po/porder/PurchaseOrder/item/partid
order by $po/custid
return $po/custid
```

The expression returns `custid` elements containing the customer IDs. The elements are in ascending order. For example, if three purchase orders matched and the purchase orders had customer IDs 1001, 1002, and 1003, the expression returns the following elements:

```
<custid xmlns="http://posample.org">1001</custid>
<custid xmlns="http://posample.org">1002</custid>
<custid xmlns="http://posample.org">1003</custid>
```

If there are multiple text searches in the same query, the DB2 Text Search engine combines the multiple text search results and returns them. For example, the following `SELECT` statement searches for employee resumes that contain the exact phrases `ruby on rails` and `ajax web`. The `WHERE` clause contains two text searches. Each text search returns a maximum of 10 results, and each text search uses a different search argument to search for employee resumes. The statement might return fewer than 10 employee IDs even if there are more than 10 employee resumes that contain both phrases.

```
SELECT EMPNO FROM EMP_RESUME
WHERE XMLEXISTS('db2-fn:xmlcolumn-contains(''EMP_RESUME.XML_FORMAT'',
  ''"ruby on rails"'', ''RESULTLIMIT=10'')')
AND XMLEXISTS('db2-fn:xmlcolumn-contains(''EMP_RESUME.XML_FORMAT'',
  ''"ajax web"'', ''RESULTLIMIT=10'')')
```

For the previous statement, DB2 Text Search returns at most 10 rows for each text search. However, if the resumes in the returned rows contain only one of the phrases (not both phrases), no employee IDs are returned.

One way to modify the SELECT statement is to combine the two text searches in the WHERE clause into a single text search. The following statement uses a single text search and returns employee IDs whose resumes have both the phrase ruby on rails and ajax web:

```
SELECT EMPNO FROM EMP_RESUME
WHERE XMLEXISTS('db2-fn:xmlcolumn-contains(''EMP_RESUME.XML_FORMAT'',
      '"ruby on rails" AND "ajax web"', 'RESULTLIMIT=10')')
```

Use a single back slash to escape the colon of the attribute of a XQuery:

```
xquery for $i in db2-fn:xmlcolumn-contains('DBCP1208.T_AUTO.T_XML',
'@xpath:'//en//en[. contains("purpose") and @a1 = "value for en\:attribute1"
and @slope = "9"] '' ') return $i/*//fn:string
```



---

## Chapter 10. Administration commands for DB2 Text Search

There are a number of commands that allow you to administer DB2 Text Search at the instance, database, table, and text-index levels. You run all of the commands using db2ts.

Use the instance-level administration commands to start and stop the DB2 Text Search instance services and clean up text search indexes that are no longer usable:

**db2ts START FOR TEXT**

Starts the DB2 Text Search instance services

**db2ts STOP FOR TEXT**

Stops the DB2 Text Search instance services

**db2ts CLEANUP FOR TEXT**

Cleans up any text search collections that are not usable

Use the database-level administration commands to set up or disable databases for DB2 Text Search and clear command locks:

**db2ts ENABLE DATABASE FOR TEXT**

Enables the current database to create, manage, and use text search indexes

**db2ts DISABLE DATABASE FOR TEXT**

Disables DB2 Text Search for a database and drops a number of text search catalog tables and views

**db2ts CLEAR COMMAND LOCKS**

Deletes command locks for all indexes in a database

Use table- and index-level commands to create and manipulate text search indexes on columns of a table:

**db2ts CREATE INDEX**

Creates a text search index

**db2ts DROP INDEX**

Drops a text search index associated with a text column

**db2ts ALTER INDEX**

Changes the characteristics of a text search index

**db2ts UPDATE INDEX**

Populates or updates a text search index based on the current contents of a text column

**db2ts CLEAR EVENTS FOR TEXT**

Deletes events from the SYSIBMTS.TSEVENT view, an events view that provides information about indexing status and errors

**db2ts CLEAR COMMAND LOCKS FOR INDEX**

Deletes all command locks for a specific text search index

**db2ts RESET PENDING FOR TABLE**

Identifies all dependent tables that are maintained for text search and executes set integrity, if necessary

**db2ts HELP**

Displays the list of **db2ts** command options and information about specific error messages

## db2ts ALTER INDEX

This command changes the update characteristics of an index.

For execution, the command needs to be prefixed with **db2ts** at the command line.

### Authorization

The privileges held by the authorization ID of the statement must include the SYSTS\_MGR role and at least one of the following authorities:

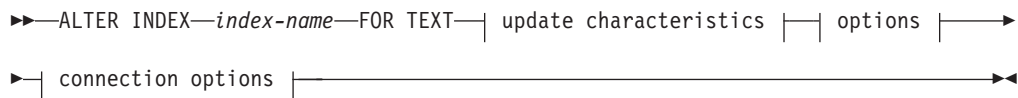
- DBADM authority
- ALTERIN privilege on the base schema
- CONTROL or ALTER privilege on the base table on which the text search index is defined

To change an existing schedule, the authorization ID must be the same as the index creator or must have DBADM authority.

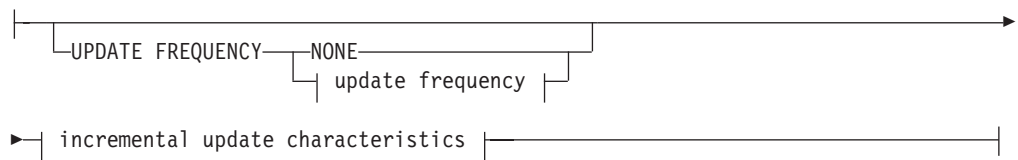
### Required connection

Database

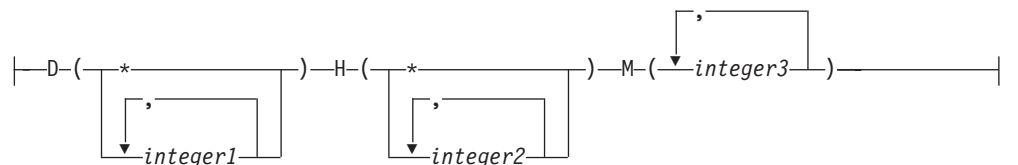
### Command syntax



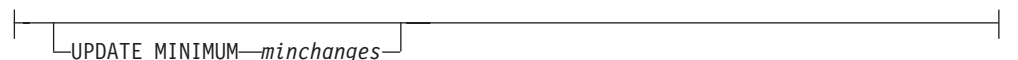
#### update characteristics:



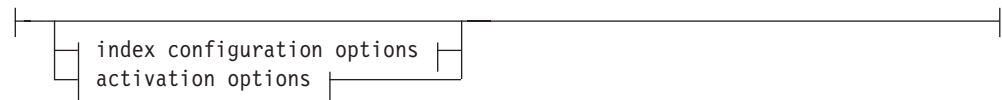
#### update frequency:



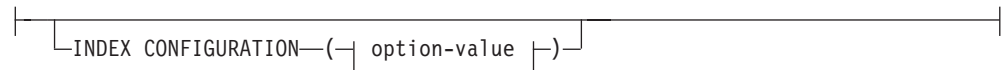
#### incremental update characteristics:



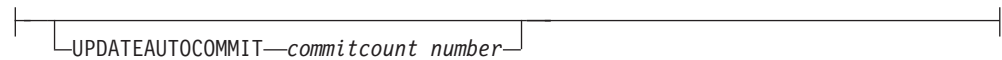
### options:



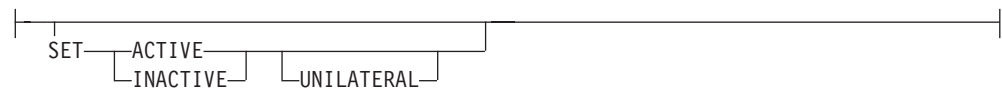
### index configuration options:



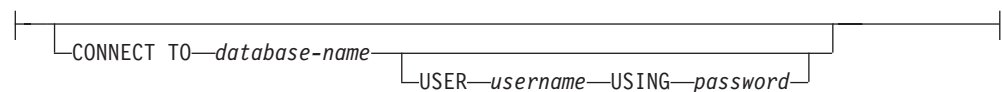
### option-value:



### activation options:



### connection options:



## Command parameters

### ALTER INDEX *index-name*

The schema and name of the index as specified in the **CREATE INDEX** command. It uniquely identifies the text search index in a database.

### UPDATE FREQUENCY

Specifies the frequency with which index updates are made. The index will be updated, if the number of changes is at least the value set for **UPDATE MINIMUM**. The update frequency **NONE** indicates that no further index updates will be made. This can be useful for a text column in a table with data that will not change. It is also useful when the user intends to manually update the index (using the **UPDATE INDEX** command). Automatic updates can only be done if the **START FOR TEXT** command has been run and the DB2 Text Search instance services are running.

The default frequency value is taken from the view SYSIBMTS.TSDEFAULTS, where DEFAULTNAME='UPDATEFREQUENCY'.

### NONE

No automatic updates will be applied to the text index. Any further index update will have to be started manually.

**D** The day(s) of the week when the index is updated.

\* Every day of the week.

*integer1*

Specific days of the week, from Sunday to Saturday: 0 to 6

**H** The hour(s) of the specified day(s) when the index is updated.

\* Every hour of the day.

*integer2*

Specific hours of the day, from midnight to 11 pm: 0 to 23

**M** The minute(s) of the specified hour(s) when the index is updated.

*integer3*

Specified as top of the hour (0), or in multiples of 5 minute increments after the hour: 0, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50 or 55

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

**UPDATE MINIMUM** *minchanges*

Specifies the minimum number of changes to text documents that must occur before the index is incrementally updated. Multiple changes to the same text document are treated as separate changes. If you do not specify the **UPDATE MINIMUM** option, the setting is left unchanged.

**INDEX CONFIGURATION** (*option-value*)

Starting with Version 9.7 Fix Pack 3 and later fix packs, this is an optional input argument of type VARCHAR(32K) that allows altering text index configuration settings. The following option is supported:

Table 9. Specifications for option-value

Option	Value	Data type	Description
SERIALUPDATE	<i>updatemode</i>	Integer	Specifies whether the update processing for a partitioned text search index should be run in parallel or in serial mode. In parallel mode the execution is distributed to the database partitions and executes independently on each node. In serial mode the execution is run without distribution and stops when a failure is encountered. Serial mode execution usually takes longer but requires significantly less resources. <ul style="list-style-type: none"><li>• 0 = parallel mode</li><li>• 1 = serial mode</li></ul>

Table 9. Specifications for option-value (continued)

Option	Value	Data type	Description
UPDATEAUTOCOMMIT	<i>commitcount_number</i>	Integer	<p>Specifies the number of index updates after which a commit is executed to preserve the previous work automatically for either initial or incremental updates.</p> <ul style="list-style-type: none"> <li>For initial updates, the index update will process batches of documents from a base table after the trigger to capture data updates is activated. After the amount of documents updated reaches the COMMITCOUNT number, the server will do an interim commit. Log entries generated by unprocessed documents will be removed from the staging table. Using the UPDATEAUTOCOMMIT option for an initial text index update will lead to a significant increase of execution time.</li> <li>For incremental updates, log entries which have been processed will be removed correspondingly from staging table with each interim commit. COMMITCOUNT counts the number of documents updated, not the number of staging table entries.</li> </ul>

#### *activation options*

Starting with Version 9.7 Fix Pack 3 and later fix packs, this input argument of type integer sets the status of a text index.

#### **ACTIVE**

Sets the text index status to active

#### **INACTIVE**

Sets the text index status to inactive

#### **UNILATERAL**

Specifies a unilateral change which only affects the status of DB2 Text Search indexes. If this argument is specified, only the status of a DB2 Text Search index is changed to active or inactive. Without the UNILATERAL argument, the activation status of the DB2 Text Search and DB2 Net Search Extender indexes is jointly switched so that only one of the text indexes is active.

#### **CONNECT TO** *database-name*

This clause specifies the database to which a connection will be established. The database must be on the local system. If specified, this clause takes precedence over the environment variable DB2DBDFT. This clause can be omitted if the following are all true:

- The DB2DBDFT environment variable is set to a valid database name.
- The user running the command has the required authorization to connect to the database server.

#### **USER** *username* **USING** *password*

This clause specifies the username and password that will be used to establish the connection.

### **Usage notes**

All limits and naming conventions, that apply to DB2 database objects and queries, also apply to DB2 Text Search features and queries. DB2 Text Search related

identifiers must conform to the DB2 naming conventions. In addition, there are some additional restrictions. For example, these identifiers can only be of the form:  
`[A-Za-z][A-Za-z0-9@#$_]*`

or

`"[A-Za-z ][A-Za-z0-9@#$_ ]*"`

Note that multiple commands cannot be executed concurrently on a text search index if they may conflict. If this command is issued while a conflicting command is running, an error will occur and the command will fail, after which you can try to run the command again. Some of the conflicting commands are:

- **ALTER INDEX**
- **CLEAR EVENTS FOR INDEX**
- **DROP INDEX**
- **UPDATE INDEX**
- **DISABLE DATABASE FOR TEXT**

Changes to the database: Updates the DB2 Text Search catalog information.

The result of activating indexes depends on the original index status. The following table describes the results.

*Table 10. Status changes without invalid index:*

Initial DB2 Text Search or Net Search Extender Status	Request Active	Request Active Unilateral	Request Inactive	Request Inactive Unilateral
Active / Inactive	No change	No change	Inactive / Active	Inactive / Inactive
Inactive / Active	Active / Inactive	Error	No change	No change
Inactive / Inactive	Active / Inactive	Active / Inactive	Inactive / Active	No change

SQL20427N and CIE0379E error messages are returned for active index conflicts.

---

## db2ts CLEANUP FOR TEXT

Cleans up DB2 Text Search collections within an instance or within a database.

When a cleanup operation is executed for a database, invalid text indexes and their associated collections are dropped. When a cleanup operation is executed for the instance, obsolete collections are removed. A collection can become obsolete if a database containing text search indexes is dropped before DB2 Text Search has been disabled for the database.

**Note:** While the commands operate on text search indexes, text search server tools operate on text search collections. A text search collection refers to the underlying representation of a text search index. The relationship between a text search index and its associated collections is 1:1 in a non-partitioned setup and 1:n in a partitioned setup, where n is the number of data partitions. Query the SYSIBMTS.TSCOLLECTIONNAMES catalog table to determine the text search collections for a text search index. For additional information, see the topic about Administration Tool for DB2 Text Search.

For execution, the command needs to be prefixed with **db2ts** at the command line.

### Authorization

To issue the command on instance level, you must be the owner of the text search server process. For the integrated text search server, this is the instance owner.

To issue the command on database level, the privileges held by the authorization ID of the statement must include the SYSTS\_ADM role and the DBADM authority.

### Required connection

This command must be issued from the DB2 database server.

### Command syntax

Instance level

▶▶—CLEANUP FOR TEXT—▶▶

Database level

▶▶—CLEANUP FOR TEXT—connection-options—▶▶

### Command parameters

None

---

## db2ts CLEAR COMMAND LOCKS

Removes all command locks for a specific text search index or for all text search indexes in the database. A command lock is created at the beginning of a text search index command, and is destroyed when it is done. It prevents undesirable conflict between different commands.

Use of this command is required in the rare case that locks remain in place due to an unexpected system behavior, and need to be cleaned up explicitly.

For execution, the command needs to be prefixed with **db2ts** at the command line.

### Authorization

The privileges held by the authorization ID of the statement used to clear locks on the index must include both of the following authorities:

- SYSTS\_MGR role
- DBADM authority or CONTROL privilege on the base table on which the index is defined

The privileges held by the authorization ID of the statement used to clear locks on the database connection must include the SYSTS\_ADM role.

### Required connection

Database

## Command syntax

```
► CLEAR COMMAND LOCKS ───────────┬── FOR TEXT ───────────►
                                └── FOR INDEX index-name ───┘
```

```
► | connection options | ───────────────────────────────────►
```

### connection options:

```
┌──────────────────────────────────────────────────────────┐
└── CONNECT TO database-name ───────────┬── USER username ─── USING password ───┘
```

## Command parameters

### FOR INDEX *index-name*

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

### CONNECT TO *database-name*

This clause specifies the database to which a connection will be established.

The database must be on the local system. If specified, this clause takes precedence over the environment variable **DB2DBDFT**. This clause can be omitted if the following are all true:

- The **DB2DBDFT** environment variable is set to a valid database name.
- The user running the command has the required authorization to connect to the database server.

### USER *username* USING *password*

This clause specifies the authorization name and password that will be used to establish the connection.

## Usage notes

You would invoke this command because the process owning the command lock is dead. In this case, the command (represented by the lock) may not have completed, and the index may not be operational. You need to take appropriate action. For example, the process executing the **DROP INDEX** command dies suddenly. It has deleted some index data, but not all the catalog and collection information. The command lock is left intact. After clearing the **DROP INDEX** command lock, you may want to re-execute the **DROP INDEX** command. In another example, the process executing the **UPDATE INDEX** command is interrupted. It has processed some documents, but not all, and the command lock is still in place. After reviewing the text search index status and clearing the **UPDATE INDEX** command lock, you can re-execute the **UPDATE INDEX** command.

When this command is issued, the content of the DB2 Text Search view **SYSIBMTS.TSLOCKS** is updated.

---

## db2ts CLEAR EVENTS FOR TEXT

This command deletes indexing events from an index's event table used for administration. The name of this table can be found in the view **SYSIBMTS.TSINDEXES** in column **EVENTVIEWNAME**.



Every index update operation that processes at least one document produces informational and, in some cases, error entries in the event table. For automatic updates, this table has to be regularly inspected. Document specific errors have to be corrected (by changing the document content). After correcting the errors, the events can be cleared (and should be, in order not to consume too much space).

For execution, the command needs to be prefixed with **db2ts** at the command line.

## Authorization

The privileges held by the authorization ID of the statement must include both of the following authorities:

- SYSTS\_MGR role
- DBADM with DATAACCESS authority or CONTROL privilege on the table on which the index is defined

## Required connection

Database

## Command syntax

```
►► CLEAR EVENTS FOR INDEX—index-name—FOR TEXT—| connection options |◄◄
```

### connection options:

```
|
|└─CONNECT TO—database-name—|
|                               └─USER—username—USING—password—|
|_____
```

## Command parameters

*index-name*

The name of the index as specified in the **CREATE INDEX** command. The index name must adhere to the naming restrictions for DB2 indexes.

**CONNECT TO** *database-name*

This clause specifies the database to which a connection will be established. The database must be on the local system. If specified, this clause takes precedence over the environment variable DB2DBDFT. This clause can be omitted if the following are all true:

- The DB2DBDFT environment variable is set to a valid database name.
- The user running the command has the required authorization to connect to the database server.

**USER** *username* **USING** *password*

This clause specifies the authorization name and password that will be used to establish the connection.

## Usage notes

All limits and naming conventions, that apply to DB2 database objects and queries, also apply to DB2 Text Search features and queries. DB2 Text Search related identifiers must conform to the DB2 naming conventions. In addition, there are some additional restrictions. For example, these identifiers can only be of the form:

```
[A-Za-z][A-Za-z0-9@#$_]*
```

or

```
"[A-Za-z ][A-Za-z0-9@#$_ ]*"
```

When regular updates are scheduled (see **UPDATE FREQUENCY** options in **CREATE INDEX** or **ALTER INDEX** commands), the event table should be regularly checked. To cleanup the DB2 Text Search event table for a text search index, use the **CLEAR EVENTS FOR INDEX** command after you have checked the reason for the event and removed the source of the error.

Be sure to make changes to all rows referenced in the event table. By changing the rows in the user table, you ensure that the next **UPDATE INDEX** attempt can be made to successfully re-index the once erroneous documents.

Note that multiple commands cannot be executed concurrently on a text search index if they may conflict. If this command is issued while a conflicting command is running, an error will occur and the command will fail, after which you can try to run the command again. Some of the conflicting commands are:

- **CLEAR EVENTS FOR INDEX**
- **UPDATE INDEX**
- **ALTER INDEX**
- **DROP INDEX**
- **DISABLE DATABASE FOR TEXT**

Changes to the database: The event table is cleared.

---

## db2ts CREATE INDEX

The **db2ts CREATE INDEX** command creates a text search index for a text column. You can then search the column data using text search functions.

The text search index does not contain any data until you run the text search **UPDATE INDEX** command, or the DB2 Administrative Task Scheduler runs the **UPDATE INDEX** command according to the defined update frequency for the index.

For execution, the command needs to be prefixed with **db2ts** at the command line.

### Authorization

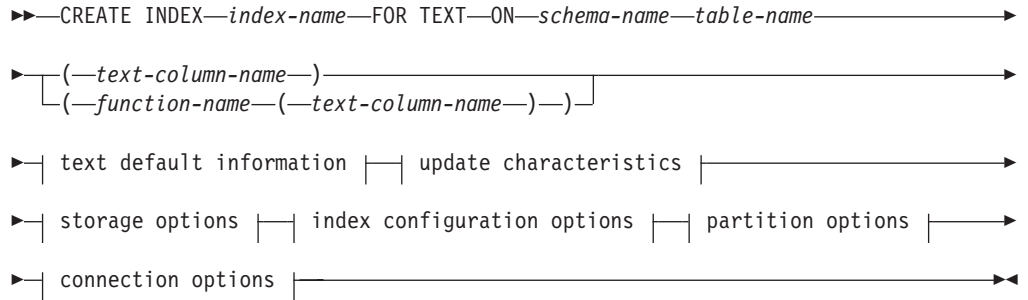
The authorization ID of the statement must include the SYSTS\_MGR role and CREATETAB authority on the database and one of the following privileges or authority :

- CONTROL privilege on the table on which the index will be defined
- INDEX privilege on the table on which the index will be defined and one of the following authorities:
  - IMPLICIT\_SCHEMA authority on the database, if the implicit or explicit schema name of the index does not exist
  - CREATEIN privilege on the schema, if the schema name of the index exists
- DBADM authority

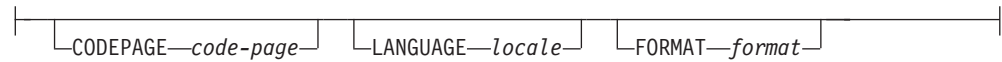
## Required connection

Database

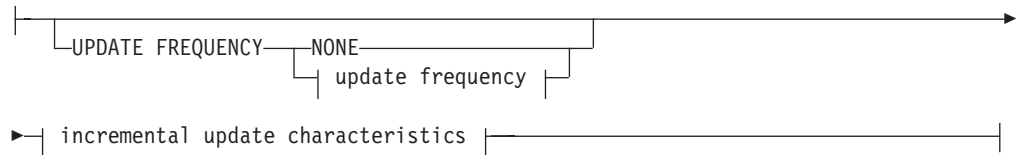
### Command syntax



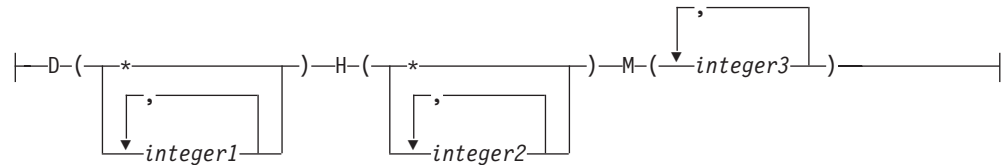
#### text default information:



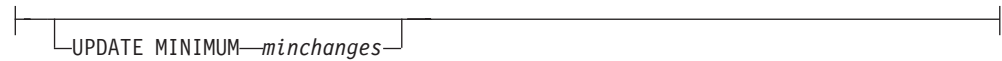
#### update characteristics:



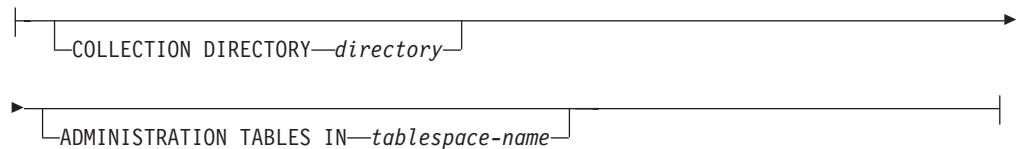
#### update frequency:



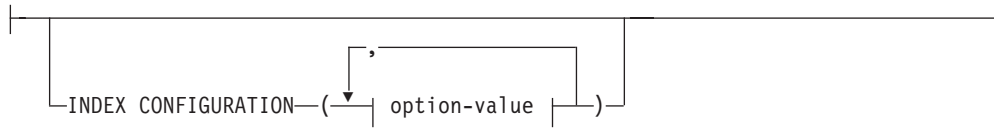
#### incremental update characteristics:



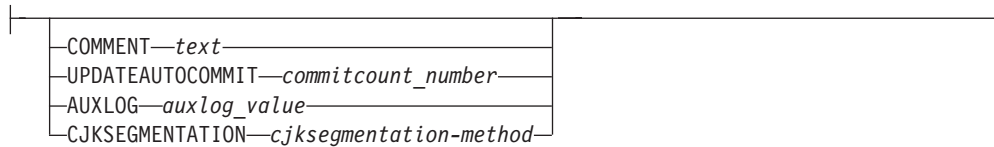
#### storage options:



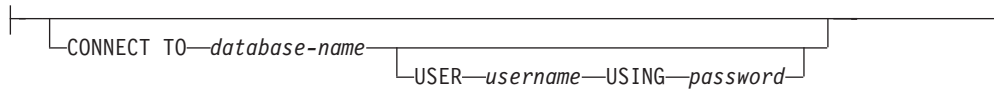
## index configuration options:



## option-value:



## connection options:



## Command parameters

### **CREATE INDEX** *index-name*

Specifies the name of the index (optionally schema qualified) to be created, that uniquely identifies the text search index within the database. The index name must adhere to the naming restrictions for DB2 indexes.

### **ON** *table-name*

The table name containing the text column. Text search indexes cannot be created on the following tables:

- federated tables
- materialized query tables
- views

### *text-column-name*

The column name of the column to be indexed. The column must be of one of the following data types: CHAR, VARCHAR, CLOB, DBCLOB, BLOB, GRAPHIC, VARGRAPHIC, or XML. If the data type of the column is not one of these data types, use a transformation function specified with *function-schema.function-name* to convert the column type to one of the valid types. Alternatively, you can specify a user-defined external function that accesses the text documents to be indexed. Only a single text search index can be created for a column.

### *function-name* (*text-column-name*)

Specifies the schema qualified name, conforming to DB2 naming conventions, of an external scalar function that accesses text documents in a column that is not of a supported type for text searching. Performs a data type conversion of that value and returns the value as one of the supported data types for text searching. Its task is to perform a column type conversion. This function must take only one parameter and return only one value.

### **CODEPAGE** *code-page*

Specifies the DB2 code page (CODEPAGE) to be used when indexing text documents. The default value is specified by the value in the view

SYSIBMTS.TSDEFAULTS, where DEFAULTNAME='CODEPAGE'; (which happens to be the database code page). This argument only applies to binary data types, such as, the column type or return type from a transformation function must be BLOB or character-type FOR BIT DATA.

**LANGUAGE** *locale*

Specifies the language to be used by DB2 Text Search for language-specific processing of a document during indexing. If you do not specify a locale, the database territory determines the default setting for **LANGUAGE**. To have your documents automatically scanned to determine the locale, specify *locale* as AUTO.

**FORMAT** *format*

Specifies the format of text documents in the column. The supported formats include: TEXT, XML, HTML, and INSO. DB2 Text Search needs this information when indexing documents. If the format is not specified, the default value is used. The default value is in the view SYSIBMTS.TSDEFAULTS, where DEFAULTNAME='FORMAT'; For columns of data type XML, the default format 'XML'; is used, regardless of the value of DEFAULTNAME. Rich text support must be installed to use the INSO format.

**UPDATE FREQUENCY**

Specifies the frequency with which index updates are made. The index is updated if the number of changes is at least the value set for **UPDATE MINIMUM**. The update frequency **NONE** indicates that no further index updates are made. An update frequency of **NONE** can be useful for a text column in a table with data that does not change. It is also useful when the user intends to manually update the index (using the **UPDATE INDEX** command). Automatic updates can only be done if the **START FOR TEXT** command has been run and the DB2 Text Search instance services are running.

The default frequency value is taken from the view SYSIBMTS.TSDEFAULTS, where DEFAULTNAME='UPDATEFREQUENCY';

**NONE**

No further index updates are made. The update has to be started manually.

**D** The days of the week when the index is updated.

- \* Every day of the week.

*integer1*

Specific days of the week, from Sunday to Saturday: 0 - 6

**H** The hours of the specified days when the index is updated.

- \* Every hour of the day.

*integer2*

Specific hours of the day, from midnight to 11 pm: 0 - 23

**M** The minutes of the specified hours when the index is updated.

*integer3*

Specified as top of the hour (0), or in multiples of 5 minute increments after the hour: 0, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50 or 55

**UPDATE MINIMUM** *minchanges*

Specifies the minimum number of changes to text documents before the index is updated incrementally at the time specified in **UPDATE FREQUENCY**. Positive integer values only are allowed. The default value is taken from the view SYSIBMTS.TSDEFAULTS, where DEFAULTNAME='UPDATEMINIMUM'.

**Note:** This value is ignored during an **UPDATE INDEX** command (unless the **USING UPDATE MINIMUM** option is used there). A small value increases consistency between the table column and the text search index. However, it also causes additional load for the system.

**COLLECTION DIRECTORY** *directory*

Specifies the directory in which the text search index collection is stored. You must specify the absolute path, where the maximum length of the absolute path name is 215 characters. The process owner of the Text Search server instance service must have read and write access on this directory.

The **COLLECTION DIRECTORY** parameter is only supported for an integrated text search server setup. Review the usage notes for additional information about collection locations.

**ADMINISTRATION TABLES IN** *tablespace-name*

Specifies the name of an existing non-temporary table space for the administration tables created for the index.

For a non-partitioned databases, if no table space is specified, the table space of the base table for which you are creating the index is used.

For a partitioned database, you must use the **ADMINISTRATION TABLES** parameter explicitly when creating text indexes. The table space must be in the same partition group as the table space of the base table to ensure that the staging tables for the text search index are distributed in the same manner as the corresponding base table.

**INDEX CONFIGURATION** (*option-value*)

Specifies additional index-related values as option value string pairs. The following values are supported:

Table 11. Specifications for option-value

Option	Value	Data type	Description
<b>COMMENT</b>	<i>text</i>	String value less than 512 bytes	Adds a string comment value to the REMARKS column in the DB2 Text Search catalog view TSINDEXES. It also adds the string comment value as the description of the collection.
<b>UPDATEAUTOCOMMIT</b>	<i>commitcount_number</i>	Integer	<p>This option specifies the number of index updates after which a commit is executed to automatically preserve the previous work for either initial or incremental updates.</p> <ul style="list-style-type: none"> <li>For initial updates, the index update will process batches of documents from a base table after the trigger to capture data updates is activated. After the number of documents updated reaches the COMMITCOUNT number, the server does an interim commit. Log entries generated by unprocessed documents are removed from the staging table. Using the <b>UPDATEAUTOCOMMIT</b> option for an initial text index update leads to a significant increase of execution time.</li> <li>For incremental updates, log entries which have been processed are removed correspondingly from the staging table with each interim commit. COMMITCOUNT counts the number of documents updated, not the number of staging table entries.</li> </ul>

Table 11. Specifications for option-value (continued)

Option	Value	Data type	Description
<b>AUXLOG</b>	<i>auxlog_value</i>	String	Controls the creation of the additional log infrastructure to capture changes that are not recognized by a trigger. The default setting for range partitioned tables is ON, otherwise it is OFF. The default value can be changed in the default table with a setting for AuxLogNorm for non-range-partitioned tables and AuxLogPart for range-partitioned tables.
<b>CJKSEGMENTATION</b>	<i>CJKSEGMENTATION_method</i>	String value less than 512 bytes	Specifies the segmentation method for documents in Chinese, Japanese, and Korean languages (zh_CN, zh_TW, ja_JP, ko_KR locale sets), including such documents when automatic language detection is enabled (LANGUAGE AUTO). If no option is specified, the value for <b>CJKSEGMENTATION</b> in the defaults table is applied. Supported values are: <ul style="list-style-type: none"> <li>• MORPHOLOGICAL</li> <li>• NGRAM</li> </ul> The specified segmentation method is added to the SYSIBMTS. TSCONFIGURATION administrative view and cannot be changed after the text index is created.

**Remember:** Non-numeric values, like comments, must be enclosed in single quotation marks. A single quote character within a string value must be represented by two consecutive single quotation marks.

**Example:**

```
INDEX CONFIGURATION (COMMENT 'Index on User''s Guide column')
```

**partition options**

Reserved for internal IBM use.

**CONNECT TO** *database-name*

This clause specifies the database to which a connection is established. The database must be on the local system. If specified, this clause takes precedence over the environment variable **DB2DBDFT**. This clause can be omitted if the following are all true:

- The **DB2DBDFT** environment variable is set to a valid database name.
- The user running the command has the required authorization to connect to the database server.

**USER** *username* **USING** *password*

This clause specifies the authorization name and password that are used to establish the connection.

**Usage notes**

All limits and naming conventions, that apply to DB2 database objects and queries, also apply to DB2 Text Search features and queries. DB2 Text related identifiers must conform to the DB2 naming conventions. In addition, there are some additional restrictions. For example, these identifiers can only be of the form:

```
[A-Za-z][A-Za-z0-9@#$_]*
```

or

```
"[A-Za-z ][A-Za-z0-9@#$_ ]*"
```

With the successful execution of the **CREATE INDEX** command:

- DB2 Text Search server data is updated. A collection of name *instance\_database-name\_index-identifier\_number* is created per database partition, as in the following example:

```
tigertail_MYTSDB_TS250517_0000
```

The collection name can be retrieved from the SYSIBMTS.TSCOLLECTIONNAMES view (column COLLECTIONNAME).

- DB2 Text Search catalog information is updated. An index staging table is created in the specified table space with appropriate DB2 indexes. In addition, an index event table is created in the specified table space. If the `auxlog` option is set, a second staging table is created to capture changes via integrity processing.
- When DB2 Text Search coexists with DB2 Net Search Extender and an active Net Search Extender index exists for the table column, the new text index is set to inactive.
- The newly created text search index is not automatically populated. The **UPDATE INDEX** command must be executed either manually or automatically (as a result of an update schedule having been defined for the index through the specification of the **UPDATE FREQUENCY** option) for the text search index to be populated.
- If a frequency is specified, a schedule task is created for the DB2 Administrative Scheduler.

Usage restrictions:

- A primary key must be defined for the table. In DB2 Text Search, a multi-column DB2 primary key can be used without type limitations. The number of primary key columns is limited to two columns less than the number of primary key columns allowed by DB2.
- The total length of all primary key columns for a table with DB2 Text Search indexes is limited to 15 bytes less than the maximum total primary key length allowed by DB2. Refer to the restrictions of the DB2 CREATE INDEX statement.

Multiple commands cannot be executed concurrently on a text search index if they might conflict. If this command is issued while a conflicting command is running, an error will occur and the command will fail, after which you can try to run the command again. A conflicting command is:

- **DISABLE DATABASE FOR TEXT**

The auxiliary log property for a text index cannot be changed after creating the index.

The following conditions must be satisfied in a partitioned database environment:

- Text search index-specific administration tables, such as staging tables, and text search indexes are distributed in a manner like the corresponding base table. Use the **ADMINISTRATION TABLES IN** clause while creating text search indexes so that the specified table space is in the same partition group as the table space of the base table.



The **CJKSEGMENTATION** option applies to zh\_CN, zh\_TW, ja\_JP and ko\_KR locale sets for Chinese, Japanese, and Korean languages. The MORPHOLOGICAL or NGRAM option specified for the segmentation method is added to the SYSIBMTS.TSCONFIGURATION administration view.

Creating an index with the **LANGUAGE** parameter set to the AUTO option allows **CJKSEGMENTATION** specification as an option. The specified segmentation method applies to Chinese, Japanese, and Korean language documents. You cannot change the value set for the *CJKSEGMENTATION\_method* value once index creation is complete.

If a text search index is created with the **LANGUAGE** parameter set to AUTO and **CJKSEGMENTATION** set to MORPHOLOGICAL, then searches for valid strings on a morphological index might not return the expected results. In such a case, use the QUERYLANGUAGE option of the **CONTAINS** function to obtain the results, as shown in the following sample statement:

```
select bookname from ngrambooks where contains (story, '△△', 'QUERYLANGUAGE=zh_CN') = 1
```

---

## db2ts DISABLE DATABASE FOR TEXT

This command reverses the changes (for example, drops the text-search related tables and view) made by the command **ENABLE DATABASE FOR TEXT**.

When issued, this command:

- Disables the DB2 Text Search feature for the database
- Drops text search catalog tables and views and related database objects
- If the **FORCE** option is specified, all text index information is removed from the database and all associated collections are deleted. See the “db2ts DROP INDEX command” for reference.

For execution, the command needs to be prefixed with **db2ts** at the command line.

### Authorization

The privileges held by the authorization ID of the statement must include both of the following authorities:

- DBADM with DATAACCESS authority.
- SYSTS\_ADM role

### Required connection

Database

### Command syntax

```
➤➤—DISABLE DATABASE FOR TEXT—FORCE | connection options | ➤➤
```

### connection options:

```
—CONNECT TO—database-name—USER—username—USING—password—
```

## Command parameters

### **FORCE**

Specifies that all text search indexes be forcibly dropped from the database.

If this option is not specified and text search indexes are defined for this database, the command will fail.

If this option is specified and DB2 Text Search service has not been started (the `db2ts START FOR TEXT` command has not been issued), the text search indexes (collections) are not dropped and need to be cleaned up manually with the `db2ts CLEANUP` command.

### **CONNECT TO** *database-name*

This clause specifies the database to which a connection will be established. The database must be on the local system. If specified, this clause takes precedence over the environment variable `DB2DBDFT`. This clause can be omitted if the following are all true:

- The `DB2DBDFT` environment variable is set to a valid database name.
- The user running the command has the required authorization to connect to the database server.

### **USER** *username* **USING** *password*

This clause specifies the authorization name and password that will be used to establish the connection.

## Usage notes

This command does not influence the DB2 Net Search Extender enablement status of the database. It deletes the DB2 Text Search catalog tables and views that are created by the **ENABLE FOR TEXT** command.

Before dropping a DB2 database that has text search index definitions, issue this command and make sure that the text indexes and collections have been removed successfully.

If some indexes could not be deleted using the **FORCE** option, the collection names are written to the `db2diag` log file.

**Note:** The user is discouraged from usage that results in orphaned collections, such as, collections that remain defined on the text search server but are not used by DB2. Here are some cases that cause orphaned collections:

- When a **DROP DATABASE CLP** command is executed without running a **DISABLE DATABASE FOR TEXT** command
- When a **DISABLE DATABASE FOR TEXT** command is executed using the **FORCE** option.
- Some other error conditions.

Multiple commands cannot be executed concurrently on a text search index if they may conflict. If this command is issued while a conflicting command is running, an error will occur and the command will fail, after which you can try to run the command again. Some of the conflicting commands are:

- **DROP INDEX**
- **UPDATE INDEX**
- **CLEAR EVENTS FOR INDEX**
- **ALTER INDEX**

## db2ts DROP INDEX

The **db2ts DROP INDEX** command drops an existing text search index.

For execution, the command needs to be prefixed with **db2ts** at the command line.

### Authorization

The privileges held by the authorization ID of the statement must include the SYSTS\_MGR role and one of the following privileges or authorities:

- CONTROL privilege on the table on which the index is defined
- DROPIN privilege on the schema on which the index is defined
- If the text search index has an existing schedule, the authorization ID must be the same as the index creator, or must have DBADM authority.

### Required connection

Database

### Command syntax

```

>> DROP INDEX index-name FOR TEXT
|-----|
| drop options | connection options |-----|
  
```

#### connection options:

```

|-----|
| CONNECT TO database-name |
| USER username USING password |
  
```

### Command parameters

#### **DROP INDEX** *index-name* **FOR TEXT**

The schema and name of the index as specified in the **CREATE INDEX** command. It uniquely identifies the text search index in a database.

#### *drop\_options*

Reserved for internal IBM use.

#### **CONNECT TO** *database-name*

This clause specifies the database to which a connection is established. The database must be on the local system. If specified, this clause takes precedence over the environment variable DB2DBDFT. This clause can be omitted if the following statements are all true:

- The DB2DBDFT environment variable is set to a valid database name.
- The user running the command has the required authorization to connect to the database server.

#### **USER** *username* **USING** *password*

This clause specifies the authorization name and password that are used to establish the connection.

## Usage notes

Multiple commands cannot be executed concurrently on a text search index if the command might conflict. If this command is issued while a conflicting command is running, an error occurs and the command fails, after which you can try to run the command again. The following commands are some common conflicting commands:

- **DROP INDEX**
- **UPDATE INDEX**
- **CLEAR EVENTS FOR INDEX**
- **ALTER INDEX**
- **DISABLE DATABASE FOR TEXT**

A **STOP FOR TEXT** command that runs in parallel with the **DROP** operation will not cause a conflicting command message, instead, if the text search server is shut down before **DROP** has removed the collection, an error will be returned that the text search server is not available.

After a text search index is dropped, text search is no longer possible on the corresponding text column. If you plan to create a new text search on the same text column, you must first disconnect from the database and then reconnect before creating the new text search index.

The **db2ts DROP INDEX FOR TEXT** command makes the following changes to the database:

- Updates the DB2 Text Search catalog information.
- Drops the index staging and event tables.
- Deletes triggers on the user text table.
- Destroys the collection associated with the DB2 Text Search index definition.

---

## db2ts ENABLE DATABASE FOR TEXT

The **db2ts ENABLE DATABASE FOR TEXT** command enables DB2 Text Search for the current database. It creates administrative tables and views, sets default values for parameters, and must run successfully before you can create text search indexes on columns in tables within the database. The command needs to be prefixed with **db2ts** at the command line.

After enabling the database, it is necessary to specify the connection information for the text search server in the **SYSIBMTS.TSSERVERS** view. The enable operation includes an attempt to populate the server data and will show a warning if the server configuration cannot be accessed. In any case, it is recommended to verify the connection information in the view. For details, see the topic about updating DB2 Text Search server information.

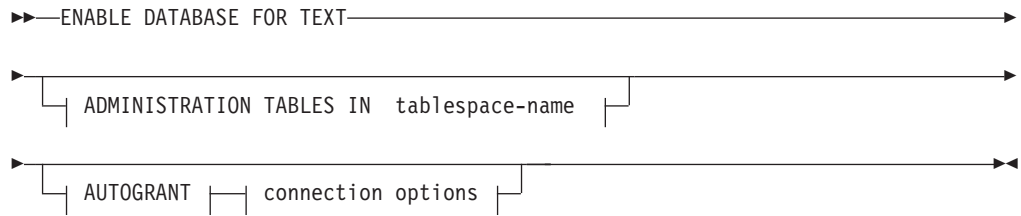
### Authorization

- The privileges held by the authorization ID of the statement must include the **SYSTS\_ADM** role and the **DBADM** authority.

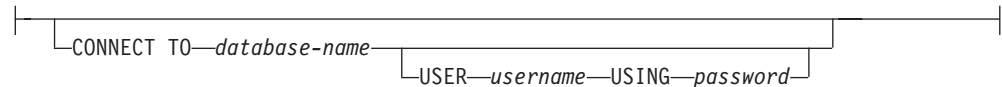
### Required connection

Database

## Command syntax



### connection options:



## Command parameters

### ADMINISTRATION TABLES IN *tablespace-name*

Specifies the name of an existing regular table space for administration tables created while enabling the database for DB2 Text Search. It is recommended that the table space is in the database partition group IBMCATGROUP. For a partitioned database, the bufferpool and table space should be defined with 32 KB page size.

If the clause is not specified, SYSTOOLSPACE is used as default table space. In this case, ensure that SYSTOOLSPACE already exists. If it does not exist, the SYSPROC.SYSINSTALLOBJECTS procedure may be used to create it.

**Note:** Use quotation marks to specify a case-sensitive table space name.

### AUTOGRANT

This option has been deprecated and does not grant privileges to the instance owner anymore. Its use is no longer suggested and might be removed in a future release.

### CONNECT TO *database-name*

This clause specifies the database to which a connection is established. The database must be on the local system. If specified, this clause takes precedence over the environment variable DB2DBDFT. This clause can be omitted if the following statements are all true:

- The DB2DBDFT environment variable is set to a valid database name.
- The user running the command has the required authorization to connect to the database server.

### USER *username* USING *password*

This clause specifies the authorization name and password used to establish the connection.

## Example

Example 1: Enable a database for DB2 Text Search creating administration tables in table space named tsspace and return any error messages in English.

```
CALL SYSPROC.SYSTS_ENABLE('ADMINISTRATION TABLES IN tsspace', 'en_US', ?)
```

The following is an example of output from this query.

```

Value of output parameters
-----
Parameter Name : MESSAGE
Parameter Value : Operation completed successfully.

Return Status = 0

```

## Usage notes

When executed successfully, this command does the following actions:

- Enables the DB2 Text Search feature for the database.
- Establishes DB2 Text Search database configuration default values in the view SYSIBMTS.TSDEFAULTS.
- Creates the following DB2 Text Search administrative views in the SYSIBMTS schema:
  - SYSIBMTS.TSDEFAULTS
  - SYSIBMTS.TSLOCKS
  - SYSIBMTS.TSINDEXES
  - SYSIBMTS.TSCONFIGURATION
  - SYSIBMTS.TSCOLLECTIONNAMES
  - SYSIBMTS.TSSERVERS

---

## db2ts HELP

**db2ts HELP** displays the list of available DB2 Text Search commands, or the syntax of an individual command.

Use the **db2ts HELP** command to get help on specific error messages as well.

For execution, the command needs to be prefixed with **db2ts** at the command line.

### Authorization

None.

### Command syntax



### Command parameters

#### HELP | ?

Provides help information for a command or a reason code.

#### *command*

The first keywords that identify a DB2 Text Search command:

- ALTER
- CLEANUP
- CLEAR (for both CLEAR COMMAND LOCKS and CLEAR EVENTS FOR INDEX)

- CREATE
- DISABLE
- DROP
- ENABLE
- RESET PENDING
- START
- STOP
- UPDATE

*sqlcode* SQLCODE for message returned by db2ts command (within or outside the administration stored procedure) or text search query.

*sqlstate* Sqlstate returned by command, administration stored procedure, or text search query.

*error-identifier*

An identifier is part of the *text-search-error-msg* that is embedded in error messages. This identifier starts with 'CIE' and is of the form CIE $nnnnn$  where  $nnnnn$  is a number. This identifier represents the specific error that is returned upon an error during text search. It may also be returned in an informational message upon completion of a text search command or in the message printed at the completion of a text search administration procedure. If the identifier does not start with 'CIE', then **db2ts help** cannot provide information about the *error-identifier*. For example, db2ts cannot provide help for a message with an *error-identifier* such as IQQR0012E.

## Usage notes

When using a UNIX shell, it might be necessary to supply the arguments to **db2ts** using double quotation marks, as in the following example:

```
db2ts "? CIE00323"
```

Without the quotation marks, the shell tries to match the wildcard with the contents of the working directory and it may give unexpected results.

If the first keyword of any db2ts command is specified, the syntax of the identified command is displayed. For the two db2ts commands that share the same first keyword (**CLEAR COMMAND LOCKS** and **CLEAR EVENTS FOR INDEX**), the syntax of both commands will be displayed when db2ts help clear is issued, but each command may be specifically displayed by adding the second keyword to distinguish them, for example db2ts help clear events. If a parameter is not specified after **?** or **HELP**, db2ts lists all available db2ts commands.

Specifying a *sqlcode*, *sqlstate*, or CIE *error-identifier* will return information about that code, state, or error identifier. For example,

```
db2ts help SQL20423
```

or

```
db2ts ? 38H10
```

or

```
db2ts ? CIE00323
```

---

## db2ts RESET PENDING command

Issues a SET INTEGRITY statement for all text-maintained staging tables that are associated with a particular table.

Certain commands cause the DB2 Text Search staging tables to go into pending mode, which blocks other database or text search operations. If you use the **db2ts RESET PENDING** command, you do not have to find all text indexes and associated staging tables and then issue a SET INTEGRITY statement for each table.

After detaching a range partition, you must issue the **RESET PENDING** command to update the staging-table content.

### Authorization

This command requires the SYSTS\_MGR role and at least one of the following authorities or privileges:

- DATAACCESS authority
- CONTROL on the base table on which the text index is created

**Note:** Currently ALL privileges are granted to the SYSTS\_MGR to allow for the creation or dropping of new index tables. However, if a dependent object like an index is implicitly created on the index table, then authorization is not propagated. To delete the dependent object, grant CONTROL privilege to the user.

### Required connection

You must issue this command from the DB2 database server.

### Command syntax

```
▶▶—RESET PENDING FOR TABLE— table-schema.table-name—FOR TEXT—▶▶  
▶  
└─|connection-options|─▶
```

### Connection-options:

```
└─CONNECT TO—database-name—  
└─USER—userid—USING—password—┘
```

### Command parameters

*table-name*

The name of the table for which the text-maintained staging infrastructure was added and for which integrity processing is required.

*table-schema*

The schema of the table for which a command was issued that resulted in a pending mode.



## Usage notes

Use the **RESET PENDING** command after issuing a command that causes the underlying tables to be put into pending mode, such as the **LOAD** command with the **INSERT** parameter, or after issuing a command that requires a **SET INTEGRITY** statement to refresh dependent tables, such as the **ALTER TABLE ... DETACH** statement.

---

## db2ts START FOR TEXT

The **db2ts START FOR TEXT** command starts the DB2 Text Search instance services that support other DB2 Text Search administration commands and the ability to reference text search indexes in SQL queries.

The **db2ts START FOR TEXT** command also includes starting processes for rich text support on the host machine running the DB2 Text Search server, if the server is configured for rich text support.

This command must be issued from the DB2 database server.

To start instance services in a partitioned database environment using an integrated text search setup, you must run the command on the integrated text search server host machine. By default, the integrated text search server host machine is the host of the lowest-numbered database partition server.

## Authorization

Instance owner. No database privilege is required

## Command syntax

```
▶▶ START FOR TEXT [STATUS] [VERIFY] ▶▶
```

## Command parameters

### STATUS

Verifies the status of the DB2 Text Search server. A verbose informational message is returned indicating the **STARTED** or **STOPPED** status of the server.

### VERIFY

Verifies the started status of the DB2 Text Search server and exits with a standard message and return code 0 indicating that the operation is successful. A non-zero code is returned for any other state of the text search server or if the status cannot be verified.

## Examples

- Check that the text search server is started.

```
Linux/UNIX:  
$ db2ts START FOR TEXT VERIFY  
CIE00001 Operation completed successfully.
```

```
$ echo $?  
0
```

```
Windows:  
C:\> db2ts START FOR TEXT VERIFY
```

```
CIE00001 Operation completed successfully.
```

```
C:\> echo %ERRORLEVEL%  
0
```

## Usage notes

- In a partitioned database environment, the **db2ts START FOR TEXT** command with the **STATUS** and **VERIFY** parameters can be issued on any one of the partition server hosts. To start the instance services, you must run the **db2ts START FOR TEXT** command on the integrated text search server host machine. The integrated text search server host machine is the host of the lowest-numbered database partition server. If custom collection directories are used, ensure that no lower numbered partitions are created later. This restriction is especially relevant for Linux and UNIX platforms. If you configure DB2 Text Search when creating an instance, the configuration initially determines the integrated text search server host. That configuration must always remain the host of the lowest-numbered database partition server.
- On Windows platforms, there is a Windows service associated with each DB2 instance for DB2 Text Search. The service name can be determined by issuing the following command:

```
DB2TS - <instance name>[-<partition number>]
```

. Apart from the using the **db2ts START FOR TEXT** command, you can also start the service using the Control Panel or the **NET START** command.

---

## db2ts STOP FOR TEXT

The **db2ts STOP FOR TEXT** command stops DB2 Text Search instance services. If the running services include processes for rich text support then those services are stopped as well.

This command must be issued from the DB2 database server.

When running this command from the command line, prefix the command with **db2ts** at the DB2 command line.

This command provides the convenience of stopping a stand-alone text search server which can also be achieved in its own installation environment using the provided script. If the instance services are already stopped, the command only checks and reports its status to the user.

### Authorization

Instance owner. No database privilege is required

### Command syntax

```
►►—STOP FOR TEXT—┬—STATUS—┬—  
                   └—VERIFY—┘
```

### Command parameters

#### STATUS

Verifies the status of the DB2 Text Search servers. A verbose informational message is returned indicating the **STARTED** or **STOPPED** status of the servers.

## VERIFY

Verifies the stopped status of the DB2 Text Search server. It exits with the standard message and return code 0 to indicate the command ran successfully. Otherwise, the text search server returns a non-zero code to indicate failure.

## Usage notes

- To avoid interrupting the execution of currently running commands, ensure no other administrative commands like the **db2ts UPDATE INDEX FOR TEXT** command are still active before issuing the **db2ts STOP FOR TEXT** command.
- In a partitioned database environment, the **db2ts START FOR TEXT** command with the **STATUS** and **VERIFY** parameters can be issued on any one of the partition server hosts.
- In a partitioned database environment on Windows platforms using an integrated text search server, stop the instance services by issuing the **db2ts STOP FOR TEXT** command on the integrated text search server host machine. By default, the integrated text search server host machine is the host of the lowest-numbered database partition server. Running the command on the integrated text search server host machine ensures that all processes and services are stopped. If the command is run on a different partition server host, the DB2TS service must be stopped separately using a command such as NET STOP.

---

## db2ts UPDATE INDEX

The **db2ts UPDATE INDEX** command updates the text search index (collection in DB2 Text Search) to reflect the current contents of the text column with which the index is associated. While the update is being performed, a search is possible. Until completion of the update, the search operates on a partially updated index.

For execution, the command needs to be prefixed with **db2ts** at the command line.

## Authorization

The privileges held by the authorization ID of the statement must include the SYSTS\_MGR role and at least one of the following authorities:

- DATAACCESS authority
- CONTROL privilege on the table on which the text index is defined
- INDEX with SELECT privilege on the base table on which the text index is defined

In addition, for an initial update the authorization requirements apply as outlined in the **CREATE TRIGGER** statement.

## Required connection

Database

## Command syntax

```
►► UPDATE INDEX index-name FOR TEXT UPDATE OPTIONS ►►  
► | connection options | ◄◄
```

## connection options:

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

## Command parameters

### UPDATE INDEX *index-name*

Specifies the name of the text search index to be updated. The index name must adhere to the naming restrictions for DB2 indexes.

### UPDATE OPTIONS

An input argument of type VARCHAR(32K) that specifies update options. If no options are specified the update is started unconditionally. The possible values are:

UPDATE OPTIONS value	Description
USING UPDATE MINIMUM	This option enforces the use of the UPDATE MINIMUM value that is defined for the text search index and processes updates if the specified minimum number of changes have occurred.
FOR DATA REDISTRIBUTION	This option specifies that a text search index in a partitioned database needs to be refreshed after data partitions were added or removed and a subsequent data redistribution operation completed. Search results may be inconsistent until the text search index was updated with the FOR DATA REDISTRIBUTION option.
ALLROWS	This option specifies that an initial update should be attempted unconditionally.
UPDATEAUTOCOMMIT	<p>This option specifies the number of index updates after which a commit is executed to preserve the previous work automatically for either initial or incremental updates.</p> <ul style="list-style-type: none"><li>• For initial updates, the index update will process batches of documents from a base table after the trigger to capture data updates is activated. After the number of documents updated reaches the COMMITCOUNT number, the server does an interim commit. Log entries generated by unprocessed documents are removed from the staging table. Using the UPDATEAUTOCOMMIT option for an initial text index update leads to a significant increase of execution time.</li><li>• For incremental updates, log entries which have been processed are removed correspondingly from staging table with each interim commit. COMMITCOUNT counts the number of documents updated, not the number of staging table entries.</li></ul>

**CONNECT TO** *database-name*

This clause specifies the database to which a connection is established. The database must be on the local system. If specified, this clause takes precedence over the environment variable DB2DBDFT. This clause can be omitted if the following statements are all true:

- The DB2DBDFT environment variable is set to a valid database name.
- The user running the command has the required authorization to connect to the database server.

**USER** *username* **USING** *password*

This clause specifies the authorization name and password that are used to establish the connection.

## Usage notes

All limits and naming conventions, that apply to DB2 database objects and queries, also apply to DB2 Text Search features and queries. DB2 Text Search related identifiers must conform to the DB2 naming conventions. In addition, there are some additional restrictions. For example, these identifiers can only be of the form:

[A-Za-z][A-Za-z0-9@#\$\_]\*

or

"[A-Za-z ][A-Za-z0-9@#\$\_ ]\*"

If synonym dictionaries have been created for a text index, executing the ALLROWS and FOR DATA REDISTRIBUTION update options removes dictionaries from existing collections. There can be new collections associated with the text index after database partitions are added. The synonym dictionaries for all associated collections have to be added again.

This command does not return until all index update processing is completed. The duration depends on the number of documents to be indexed and the number of documents already indexed. The collection name for the index can be retrieved from the SYSIBMTS.TSCOLLECTIONNAMES view (column COLLECTIONNAME).

Multiple commands cannot be executed concurrently on a text search index if they might conflict. If you run this command while a conflicting command is running, an error occurs and the command fails, after which you can try to run the command again. The following commands are some of the common conflicting commands:

- **UPDATE INDEX**
- **CLEAR EVENTS FOR INDEX**
- **ALTER INDEX**
- **DROP INDEX**
- **DISABLE DATABASE FOR TEXT**

**Note:** In cases of individual document errors, the documents must be corrected. The primary keys of the erroneous documents can be looked up in the event table for the index. The next **UPDATE INDEX** command reprocesses these documents if the corresponding rows in the user table are modified.

Changes to the database:

- Insert rows to the event table (including parser error information from DB2 Text Search).
- Delete from the index staging table in case of incremental updates.
- Before first update, create triggers on the user text table.
- The collection is updated.
- New or changed documents are parsed and indexed.
- Deleted documents are discarded from the index.

---

## Chapter 11. DB2 Text Search stored procedures

DB2 Text Search provides several administrative SQL routines for running commands and for returning the result messages of the commands that you run and the result message reason codes.

You can run the following **db2ts** commands using the administrative SQL routines:

- Enable a database - **SYSPROC.SYSTS\_ENABLE**
- Configure a database - **SYSPROC.SYSTS\_CONFIGURE**
- Disable a database - **SYSPROC.SYSTS\_DISABLE**
- Create a text index - **SYSPROC.SYSTS\_CREATE**
- Update a text index - **SYSPROC.SYSTS\_UPDATE**
- Alter a text index - **SYSPROC.SYSTS\_ALTER**
- Drop a text index - **SYSPROC.SYSTS\_DROP**
- Clear events for a text index - **SYSPROC.SYSTS\_CLEAR\_EVENTS**
- Clear command locks - **SYSPROC.SYSTS\_CLEAR\_COMMANDLOCKS**
- Reset pending status - **SYSPROC.SYSTS\_ADMIN\_CMD**
- Cleanup inactive indexes - **SYSPROC.SYSTS\_CLEANUP**

---

### **SYSTS\_ADMIN\_CMD procedure - Run text search administration commands**

The **SYSTS\_ADMIN\_CMD** procedure is used by applications to run text search administrative commands using the SQL CALL statement.

#### **Authorization**

EXECUTE privilege on the SYSTS\_ADMIN\_CMD procedure and the required authorization as listed for the requested operation.

#### **Default PUBLIC privilege**

In a non-restrictive database, EXECUTE privilege is granted to PUBLIC when the procedure is automatically created.

#### **Syntax**

►►SYSTS\_ADMIN\_CMD(—*command-string*—,—*message-locale*—,—*message*—)◄◄

The schema is SYSPROC.

#### **Procedure parameters**

##### *command-string*

An input argument of type VARCHAR (32 K) that specifies a single DB2 Text Search index administration command to be executed. The command syntax is the same as the DB2 Text Search command except for the connection options, which are not supported through this procedure. Commands that you issue through this procedure use the current connection.

The procedure supports the following DB2 Text Search commands:

- ALTER INDEX
- CLEAR COMMAND LOCKS
- CLEAR EVENTS
- CREATE INDEX
- DISABLE DATABASE
- DROP INDEX
- ENABLE DATABASE
- RESET PENDING
- UPDATE INDEX

*message-locale*

An input argument of type VARCHAR(33) that specifies the language for any error message text that is returned. If the argument is null or an empty string or the message files for the specified locale are not available on the server, 'en\_US' is used.

*message*

An output argument of type VARCHAR(32K) that specifies a warning or informational message for an operation that is considered successful.

## Example

The following example updates the MYTEXTINDEX text search index in schema DB2TS and returns any error messages in English:

```
CALL SYSPROC.SYSTS_ADMIN_CMD  
('UPDATE INDEX DB2TS.MYTEXTINDEX FOR TEXT', 'en_US', ?);
```

Sample output is as follows:

```
Value of output parameters  
-----  
Parameter Name : MESSAGE  
Parameter Value : CIE00001 Operation completed successfully.  
  
Return Status = 0
```

## Usage notes

If the command does not run successfully, SQLCODE -20427 and SQLSTATE 38H14 are returned with the text-search-specific error message. For example, if index MYTEXTINDEX exists and the following statement is issued:

```
CALL SYSPROC.SYSTS_ADMIN_CMD ('CREATE INDEX MYTEXTINDEX FOR TEXT  
ON DB2TS.TEXTBOOKS (STORY)', 'en_US', ?)
```

Index creation fails with the following error message:

```
SQL20427N An error occurred during a text search administration  
procedure or command. The error message is "CIE00201 Text search  
index "DB2TS"."MYTEXTINDEX" already exists.". SQLSTATE=38H14
```

If an SQLCODE is returned by the procedure, the message might be truncated. Full message information is in the **db2diag** log files.



## SYSTS\_ALTER procedure - Change the update characteristics of an index

This procedure changes the update characteristics of an index.

The procedure issues an **ALTER INDEX** text search administration command on the database server.

### Authorization

The privileges held by the authorization ID of the statement must include the SYSTS\_MGR role and at least one of the following authorities:

- DBADM authority
- ALTERIN privilege on base schema
- CONTROL or ALTER privilege on the base table on which the text search index is defined

To change an existing schedule, the authorization ID must be the same as the index creator or must have DBADM authority.

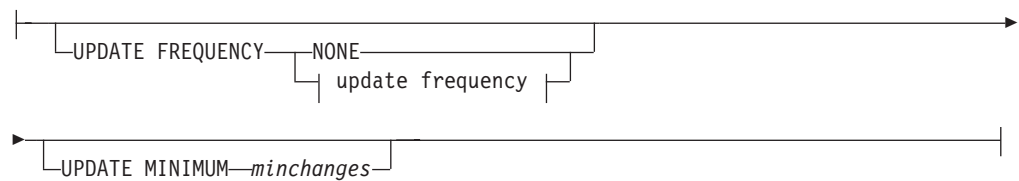
### Default PUBLIC privilege

None

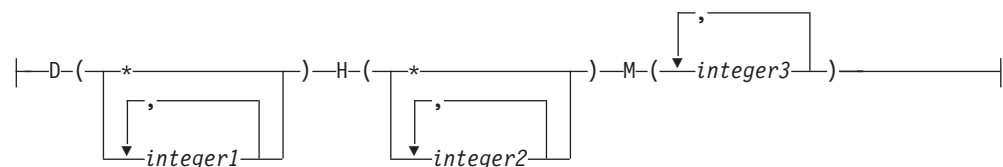
### Syntax

```
►► SYSTS_ALTER (—index_schema—, —index_name—, — update characteristics |  
► | options |, —message_locale—, —message—) ►►
```

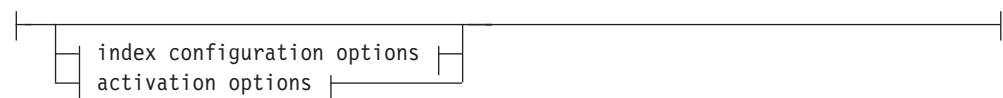
#### update characteristics:



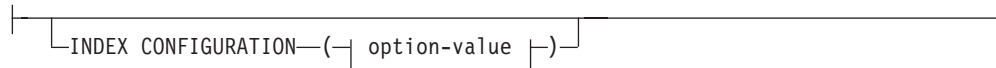
#### update frequency:



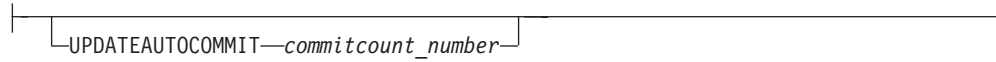
#### options:



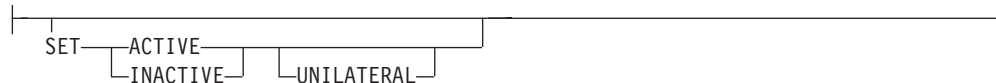
## index configuration options:



## option-value:



## activation options:



The schema is SYSPROC.

## Procedure parameters

### *index\_schema*

An input argument of type VARCHAR(128) that specifies the schema of the text search index. The *index\_schema* must follow the naming restriction for DB2 schema names. If the argument is null or an empty string, the value of CURRENT SCHEMA is used. The *index\_schema* is case-sensitive.

### *index\_name*

An input argument of type VARCHAR(128) that specifies the name of the index. Together with *index\_schema*, it uniquely identifies a text search index in a database. The *index\_name* is case-sensitive.

### update characteristics

An input argument of type VARCHAR(32K) that specifies the alter options. The alter options allowed are as follows:

#### UPDATE FREQUENCY

Specifies the frequency with which index updates are made. The index will be updated, if the number of changes is at least the value set for **UPDATE MINIMUM**. The update frequency **NONE** indicates that no further index updates will be made. This can be useful for a text column in a table with data that will not change. It is also useful when the user intends to manually update the index (using the **UPDATE INDEX** command). Automatic updates can only be done if the DB2\_ATS\_ENABLE registry variable is set and the **START FOR TEXT** command is issued.

The default frequency value is taken from the view SYSIBMTS.TSDEFAULTS, where DEFAULTNAME='UPDATEFREQUENCY'.

#### **NONE**

No automatic updates will be applied to the text index. Any further index update will have to be started manually.

**D** The day(s) of the week when the index is updated.

\* Every day of the week.

*integer1*

Specific days of the week, from Sunday to Saturday: 0 to 6

**H** The hour(s) of the specified day(s) when the index is updated.

\* Every hour of the day.

*integer2*

Specific hours of the day, from midnight to 11 pm: 0 to 23

**M** The minute(s) of the specified hour(s) when the index is updated.

*integer3*

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

**UPDATE MINIMUM** *minchanges*

Specifies the minimum number of changes to text documents that must occur before the index is incrementally updated. Multiple changes to the same text document are treated as separate changes. If you do not specify the **UPDATE MINIMUM** option, the setting is left unchanged.

**INDEX CONFIGURATION** (*option-value*)

Starting with Version 9.7 Fix Pack 3 and later fix packs, this is an optional input argument of type VARCHAR(32K) that allows altering text index configuration settings. The following option is supported:

Table 12. Specifications for option-value

Option	Value	Data type	Description
SERIALUPDATE	<i>updatemode</i>	Integer	Specifies whether the update processing for a partitioned text search index should be run in parallel or in serial mode. In parallel mode the execution is distributed to the database partitions and executes independently on each node. In serial mode the execution is run without distribution and stops when a failure is encountered. Serial mode execution usually takes longer but requires significantly less resources. <ul style="list-style-type: none"><li>• 0 = parallel mode</li><li>• 1 = serial mode</li></ul>

Table 12. Specifications for option-value (continued)

Option	Value	Data type	Description
UPDATEAUTO COMMIT	<i>commitcount</i> <i>_number</i>	Integer	<p>Specifies the number of index updates after which a commit is executed to preserve the previous work automatically for either initial or incremental updates.</p> <ul style="list-style-type: none"> <li>• For initial updates, the index update will process batches of documents from a base table after the trigger to capture data updates is activated. After the amount of documents updated reaches the COMMITCOUNT number, the server will do an interim commit. Log entries generated by unprocessed documents will be removed from staging table. Using the UPDATEAUTOCOMMIT option for an initial text index update will lead to a significant increase of execution time.</li> <li>• For incremental updates, log entries which have been processed will be removed correspondingly from staging table with each interim commit. COMMITCOUNT counts the number of documents updated, not the number of staging table entries.</li> </ul>

*activation options*

Starting with Version 9.7 Fix Pack 3 and later fix packs, this input argument of type integer sets the status of a text index.

**ACTIVE**

Sets the text index status to active

**INACTIVE**

Sets the text index status to inactive

## UNILATERAL

Specifies a unilateral change which only affects the status of DB2 Text Search indexes. If this argument is specified, only the status of a DB2 Text Search index is changed to active or inactive. Without the UNILATERAL argument, the activation status of the DB2 Text Search and DB2 Net Search Extender indexes is jointly switched so that only one of the text indexes is active.

### *message\_locale*

An input argument of type VARCHAR(33) that specifies the locale to be used for any error message returned. If the argument is null or an empty string, or the message files for the specified locale are not available on the server, 'en\_US' is used.

### *message*

An output argument of type VARCHAR(32K) that specifies a warning or informational message for a successfully completed operation.

## Examples

*Example 1:* In the following example, the update characteristics of a text search index are being altered. This index was originally created with *index\_schema* 'db2ts' and *index\_name* 'myTextIndex'. By using 'UPDATE FREQUENCY NONE', the intention is to make no further updates to the text search index as possibly no changes are expected for the associated table column. Any error messages are requested to be returned in English. When the procedure succeeds, the output parameter message indicative of the successful operation is returned to the caller.

```
CALL SYSPROC.SYSTS_ALTER('db2ts', 'myTextIndex',  
'UPDATE FREQUENCY NONE', 'en_US', ?)
```

The following is an example of output from this query.

```
Value of output parameters  
-----  
Parameter Name : MESSAGE  
Parameter Value : Operation completed successfully.  
  
Return Status = 0
```

*Example 2:* In the following example, the SYSTS\_ALTER stored procedure is called to alter the update-characteristics for a text search index with *index\_schema* 'db2ts' and *index\_name* 'myTextIndex2'. The intention is to ensure that updates to the index occur every hour on the hour. However, this index does not exist and results in an error.

```
CALL SYSPROC.SYSTS_ALTER('db2ts', 'myTextIndex2',  
'update frequency D(*) H(*) M(0)', 'en_US', ?)
```

The following is an example of output from this query.

```
SQL20427N An error occurred during a text search administration  
procedure or command. The error message is "CIE00316 Text search  
index "db2ts"."myTextIndex2" does not exist. ". SQLSTATE 38H14
```

## Usage notes

- Text search administration procedures use an existing connection to the database. It is recommended to commit all transaction changes before executing a text search administration procedure to avoid any unexpected impact from a commit or rollback in the procedure. One way to achieve this is to turn on AUTOCOMMIT.

- Multiple procedures or commands cannot be run concurrently on a text search index if they might conflict. Some of the conflicting procedures and commands are:
  - SYSTS\_ALTER procedure or ALTER INDEX db2ts command
  - SYSTS\_CLEAR\_EVENTS procedure or CLEAR EVENTS FOR INDEX db2ts command
  - SYSTS\_DISABLE procedure or DISABLE DATABASE FOR TEXT db2ts command
  - SYSTS\_DROP procedure or DROP INDEX db2ts command
  - SYSTS\_UPDATE procedure or UPDATE INDEX db2ts command
  - SYSTS\_CONFIGURE procedure

If there is a conflict, the procedure returns an SQLCODE -20426 and SQLSTATE 38H13.

- When this procedure is run to change the frequency, a schedule task is created, updated or dropped for the text index.
- The result of activating indexes depends on the original index status. The following table describes the results.

Table 13. Status changes without invalid index:

Initial DB2 Text Search or Net Search Extender Status	Request Active	Request Active Unilateral	Request Inactive	Request Inactive Unilateral
Active / Inactive	No change	No change	Inactive / Active	Inactive / Inactive
Inactive / Active	Active / Inactive	Error	No change	No change
Inactive / Inactive	Active / Inactive	Active / Inactive	Inactive / Active	No change

SQL20427N and CIE0379E error messages are returned for active index conflicts.

---

## SYSTS\_CLEANUP procedure - Remove invalid text search indexes

This procedure removes invalid text search indexes and their associated collections from the database. An index can become invalid when database operations are executed that affect all of the table content, for example, truncate.

The procedure serves as an alternative to the **db2ts CLEANUP FOR TEXT** command for the database scope. To remove obsolete collections on the instance level, the command must be used.

### Authorization

The privileges held by the authorization ID of the statement must include the SYSTS\_ADM role with DBADM and DATAACCESS authority.

### Default PUBLIC privilege

None

## Syntax

```
►►SYSTS_CLEANUP(—options—,—message_locale—,—message—)◄◄
```

The schema is SYSPROC.

## Procedure parameters

### *options*

An input argument of type VARCHAR(32K). Only a NULL or an empty parameter is accepted. Reserved for internal IBM use.

### *message\_locale*

An input argument of type VARCHAR(33) that specifies the locale to be used for any error message returned. If the argument is null or an empty string, or the message files for the specified locale are not available on the server, 'en\_US' is used.

### *message*

An output argument of type VARCHAR(32K) that specifies a warning or informational message for a successfully completed operation.

## Example

*Example 1:* In the following example, all invalid text search indexes in the current database are dropped. The message language is set to English, and when the procedure succeeds, the output parameter message indicative of the successful operation is returned to the caller.

```
CALL SYSPROC.SYSTS_CLEANUP('', 'en_US', ?)
```

Value of output parameters

-----

Parameter Name : MESSAGE

Parameter Value : Operation completed successfully.

Return Status = 0

*Example 2:* Output where an error is returned:

```
SQL0462W Command or routine "SYSTS_CLEANUP" (specific name "*N") has returned a warning SQLSTATE, with diagnostic text "CIE00212W 2 of 20 collections could not be deleted. Check db2diag.log for details.". SQLSTATE=01H14
```

---

## SYSTS\_CLEAR\_COMMANDLOCKS procedure - Remove command locks for text search indexes

This procedure removes all command locks for a specific text search index or for all text search indexes in the database.

### Authorization

The privileges held by the authorization ID of the statement used to clear locks on the index must include both of the following authorities:

- SYSTS\_MGR role
- DBADM authority or CONTROL privilege on the base table on which the index is defined

The privileges held by the authorization ID of the statement used to clear locks on the database connection must include the SYSTS\_ADM role.

## Default PUBLIC privilege

None

A command lock is created at the beginning of a text search index command, and is destroyed when the command has completed. It prevents undesirable conflict between different commands. Use of this procedure is required in the rare case that locks remain in place due to an unexpected system behavior, and need to be cleaned up explicitly.

This procedure issues the **CLEAR COMMAND LOCKS** text search administration command on the database server.

## Syntax

```
►► SYSTS_CLEAR_COMMANDLOCKS(—index_schema—,—index_name—,——————►  
►—message_locale—,—message—)—————►◄
```

The schema is SYSPROC.

## Procedure parameters

### *index\_schema*

An input argument of type VARCHAR(128) that specifies the schema of the text index. The *index\_schema* must follow the naming restriction for DB2 schema names. If the argument is null or an empty string, the value of CURRENT SCHEMA is used. The *index\_schema* is case-sensitive.

### *index\_name*

An input argument of type VARCHAR(128) that specifies the name of the index. Together with *index\_schema*, it uniquely identifies a text search index in a database. If the argument is null or an empty string, the procedure deletes command locks for all text search indexes in the database. The *index\_name* is case-sensitive.

### *message\_locale*

An input argument of type VARCHAR(33) that specifies the locale to be used for any error message returned. If the argument is null or an empty string, or the message files for the specified locale are not available on the server, 'en\_US' is used.

### *message*

An output argument of type VARCHAR(32K) that specifies a warning or informational message for a successfully completed operation.

## Examples

*Example 1:* In the following example, SYSTS\_CLEAR\_COMMANDLOCKS is issued for a text search index with *index\_schema* 'db2ts' and *index\_name* 'myTextIndex'. Error messages are requested to be returned in English. When the procedure succeeds, the output parameter message indicative of the successful operation is returned to the caller.

```
CALL SYSPROC.SYSTS_CLEAR_COMMANDLOCKS('db2ts', 'myTextIndex', 'en_US', ?)
```



The following is an example of output from this query.

```
Value of output parameters
-----
Parameter Name : MESSAGE
Parameter Value : Operation completed successfully.

Return Status = 0
```

*Example 2:* In the following example, SYSTS\_CLEAR\_COMMANDLOCKS is called to clear the command locks for a text search index with *index\_schema* 'db2ts' and *index\_name* 'myTextIndex2'. This index does not exist and the procedure returns an error message.

```
CALL SYSPROC.SYSTS_CLEAR_COMMANDLOCKS('db2ts', 'myTextIndex2', 'en_US', ?)
```

The following is an example of output from this query.

```
SQL20427N An error occurred during a text search administration
procedure or command. The error message is "CIE00316 Text search
index "db2ts"."myTextIndex2" does not exist. ". SQLSTATE 38H14
```

## Usage notes

- Text search administration procedures use an existing connection to the database. It is recommended to commit all transaction changes before executing a text search administration procedure to avoid any unexpected impact from a commit or rollback in the procedure. One way to achieve this is to turn on AUTOCOMMIT.
- You would invoke this procedure because the process owning the command lock is dead. In this case, the command (represented by the lock) may not have completed, and the index may not be operational. You need to take appropriate action. For example, the process executing the DROP INDEX command dies suddenly. It has deleted some index data, but not all the catalog and collection information. The command lock is left intact. After clearing the DROP INDEX command lock, you may want to re-execute the SYSTS\_DROP procedure. In another example, the process executing the UPDATE INDEX command is interrupted. It has processed some documents, but not all, and the command lock is still in place. After reviewing the text search index status and clearing the UPDATE INDEX command lock, you can re-execute the UPDATE INDEX command.
- When this procedure is run, the content of the DB2 Text Search view SYSIBMTS.TSLOCKS is updated.

---

## SYSTS\_CLEAR\_EVENTS procedure - Delete indexing events from an index's event table

This procedure deletes indexing events from an index's event table used for administration.

The name of the event table can be found in the view SYSIBMTS.TSINDEXES in column EVENTVIEWNAME. Every index update operation that processes at least one document produces informational and, in some cases, error entries in the event table. For automatic updates, the event table has to be regularly inspected. Document specific errors must be corrected by changing the document content. After correcting the errors, the events can be cleared (and should be, in order not to consume too much space).

The procedure issues a **CLEAR EVENTS FOR INDEX** text search administration command on the database server.

## Authorization

The privileges held by the authorization ID of the statement must include both of the following authorities:

- SYSTS\_MGR role
- DBADM with DATAACCESS authority or CONTROL privilege on the table on which the index is defined

## Default PUBLIC privilege

None

## Syntax

```
►►—SYSTS_CLEAR_EVENTS—(—index_schema—,—index_name—,——————►  
►—message_locale—,—message—)—————►—————►
```

The schema is SYSPROC.

## Procedure parameters

### *index\_schema*

An input argument of type VARCHAR(128) that specifies the schema of the text search index. The *index\_schema* must follow the naming restriction for DB2 schema names. If the argument is null or an empty string, the value of CURRENT SCHEMA is used. The *index\_schema* is case-sensitive.

### *index\_name*

An input argument of type VARCHAR(128) that specifies the name of the index. Together with *index\_schema*, it uniquely identifies a text search index in a database. The *index\_name* is case-sensitive.

### *message\_locale*

An input argument of type VARCHAR(33) that specifies the locale to be used for any error message returned. If the argument is null or an empty string, or the message files for the specified locale are not available on the server, 'en\_US' is used.

### *message*

An output argument of type VARCHAR(32K) that specifies a warning or informational message for a successfully completed operation.

## Examples

*Example 1:* In the following example, SYSTS\_CLEAR\_EVENTS is being called for a text search index that was created with *index\_schema* 'db2ts' and *index\_name* 'myTextIndex'. Any error messages are requested to be returned in English. When the procedure succeeds, the output parameter message indicative of the successful operation is returned to the caller.

```
CALL SYSPROC.SYSTS_CLEAR_EVENTS('db2ts', 'myTextIndex', 'en_US', ?)
```

The following is an example of output from this query.

```

Value of output parameters
-----
Parameter Name : MESSAGE
Parameter Value : Operation completed successfully.

Return Status = 0

```

*Example 2:* In the following example, SYSTS\_CLEAR\_EVENTS is called to clear the event table entries for a text search index with *index\_schema* 'db2ts' and *index\_name* 'myTextIndex2'. This index does not exist and results in an error.

```
CALL SYSPROC.SYSTS_CLEAR_EVENTS('db2ts', 'myTextIndex2', 'en_US', ?)
```

The following is an example of output from this query.

```
SQL20427N An error occurred during a text search administration
procedure or command. The error message is "CIE00316 Text search
index "db2ts"."myTextIndex2" does not exist. ". SQLSTATE 38H14
```

## Usage notes

- Text search administration procedures use an existing connection to the database. The current transaction might be committed or rolled back depending on the completion of the procedures. As such, you might want to commit all transaction changes to avoid any unexpected impact from such a commit or rollback. One way to achieve this is to turn on AUTOCOMMIT.
- Multiple procedures or commands cannot be run concurrently on a text search index if they might conflict. Some of the conflicting procedures and commands are:
  - SYSTS\_ALTER procedure or ALTER INDEX db2ts command
  - SYSTS\_DISABLE procedure or DISABLE DATABASE FOR TEXT db2ts command
  - SYSTS\_DROP procedure or DROP INDEX db2ts command
  - SYSTS\_CONFIGURE procedure
  - SYSTS\_UPDATE procedure or UPDATE INDEX db2ts command

If there is a conflict, the procedure returns an SQLCODE -20426 and SQLSTATE 38H13.

- When regular updates are scheduled (see UPDATE FREQUENCY options in SYSTS\_CREATE or SYSTS\_ALTER procedures), the event table should be checked regularly.
- To clean up the DB2 Text Search event table for a text search index, use the SYSTS\_CLEAR\_EVENTS procedure or **CLEAR EVENTS FOR INDEX** db2ts command after you have checked the reason for the event and removed the source of the error.
- Ensure that changes have been made to all rows referenced in the event table. By changing the rows in the user table, you ensure that when you run the SYSTS\_UPDATE procedure or **UPDATE INDEX** db2ts command again, an attempt is made to index the erroneous documents again.
- When this command is issued, the event table is cleared.

---

## SYSTS\_CONFIGURE procedure - Configure current database for text search

The **SYSTS\_CONFIGURE** procedure applies text search server connection information to the text search catalog.

Certain text search server properties have to be reflected in the text search administration tables (text search catalog). These properties are associated with a text index when the index is created. When parameters such as the token are updated periodically, the change has to be reflected in the database and the properties for all the indexes have to be updated as well.

This procedure is required initially for:

- Incomplete enablement
- Stand-alone text search server setups
- Partitioned databases
- and further on, following any updates to text search server connection information

For subsequent updates, ensure that no text search administrative operation is active and shut down the currently configured text search server.

During database enablement the SYSIBMTS.TSSERVER administrative view is updated with connection information for the integrated text search server. Review and update the text server information in the SYSIBMTS.TSSERVER view with the relevant text search server data and run the SYSTS\_CONFIGURE procedure to apply the updated information. For multiple databases in the instance, configure each database with the information for the same text search server.

Generally the sequence of operations is as follows:

1. Configure a text search server. Integrated text search servers can be configured during installation or when a DB2 instance is created. A stand-alone text search server is configured separately.
2. Enable a database for text search by using the **db2ts ENABLE** command, or the **SYSTS\_ENABLE** or **SYSTS\_ADMIN\_CMD** procedures with the ENABLE option.
3. Determine parameters for the text search server as needed for the SYSIBMTS.TSSERVERS view.
4. Update the SYSIBMTS.TSSERVERS administrative view with the parameters for the text search server.
  - If the view is empty then use an INSERT statement. For example:

```
INSERT INTO SYSIBMTS.TSSERVERS (HOST, PORT, TOKEN, SERVERSTATUS)
VALUES ('localhost', 55000, '9kfsjg48=', 0);
```
  - If the view already contains a row then use a SQL UPDATE statement. For example:

```
UPDATE SYSIBMTS.TSSERVERS SET (HOST, PORT, TOKEN) =
('tsmach1.ibm.com', 55002, 'k3j4fjk9u=')
```
5. Execute the SYSTS\_CONFIGURE procedure.

## Authorization

The privileges held by the authorization ID of the statement must include the SYSTS\_ADM role.

## Default PUBLIC privilege

None

## Syntax

► SYSTS\_CONFIGURE (—options—, —message-locale—, —message—) ◄

The schema is SYSPROC.

### Procedure parameter

#### *options*

An input argument of type VARCHAR(32K) that specifies the options to be used. If no options are needed, the argument can be null or an empty string.

#### *message-locale*

An input argument of type VARCHAR(33) that specifies the locale to be used for any returned error message. If the argument is null, an empty string or the message files for the specified locale are not available on the server, en\_US is used.

#### *message*

An output argument of type VARCHAR(32K) that specifies a warning or informational message for an operation that is considered successful.

## Example

Example 1: Prepare a database for text search services and return any messages in English.

```
CALL SYSTS_ENABLE('', 'en_US', ?)"

INSERT INTO SYSIBMTS.TSSERVERS
(HOST,PORT,TOKEN,KEY,LOCALE,SERVERTYPE,SERVERSTATUS)
VALUES ('tsmach1.ibm.com', 55000, '9kfsjg48=', 'en_US', 0);

CALL SYSPROC.SYSTS_CONFIGURE('', 'en_US', ?)
```

An example of output from this query:

Value of output parameters

-----

Parameter Name : MESSAGE

Parameter Value : Operation completed successfully.

Return Status = 0

## Usage notes

- A SYSIBMTS.TSSERVERS view is created when a database is enabled for text search. This view is updated with information about the integrated text search server during database enablement. If the text search server configuration cannot be obtained, the enable operation will end with an 'incomplete enablement' warning.
- The SYSTS\_CONFIGURE procedure must be issued anytime a row is inserted or updated into SYSIBMTS.TSSERVERS. Make sure that no text search administration operation is active and shut down the text search server before updating SYSIBMTS.TSSERVERS.
- When updating SYSIBMTS.TSSERVERS in a database, all text search enabled databases should be updated with the same parameters. Only a single text search server is supported with a given DB2 instance.

- Running the **SYSTS\_CONFIGURE** procedure for a database registers the use of the configured text search server for the instance. Not running the procedure does not result in a severe error but the response to some commands can be unexpected.
- Multiple procedures or commands cannot be executed concurrently on a text search index if they might conflict. Some of the conflicting procedures and commands are:
  - SYSTS\_ALTER procedure
  - SYSTS\_DISABLE procedure
  - SYSTS\_CONFIGURE procedure
  - SYSTS\_UPDATE procedure

If there is a conflict, the procedure returns an SQLCODE -20426 and SQLSTATE 38H13.

- Certain aspects relating to the text search installation and DB2 instance configuration for text search have to be updated. They include:
  - An indication whether the search server utilized by the DB2 instance is integrated (configured by DB2 as part of the DB2 instance), or if it is a separate stand-alone installation of the ECMTS server.
  - An indication if the text search setup is enabled for rich text support.

---

## **SYSTS\_CREATE procedure - Create a text search index on a column**

The SYSTS\_CREATE procedure creates a text search index for a text column by issuing the DB2 Text Search **CREATE INDEX** command on the database server. After you create and update a text search index, you can search the column data by using text search functions.

The index does not contain any data until an index update operation is processed. The update operation can be started by using the stored procedure interface or the command line interface. For the latter, you can explicitly issue the DB2 Text Search **UPDATE INDEX** command, or the command is implicitly issued by the DB2 Administrative Task Scheduler, according to the update frequency defined for the index

### **Authorization**

The authorization ID of the statement must include the SYSTS\_MGR role and CREATETAB authority on the database and one of the following privileges or authority:

- CONTROL privilege on the table on which the index will be defined
- INDEX privilege on the table on which the index will be defined and one of the following authorities:
  - IMPLICIT\_SCHEMA authority on the database, if the implicit or explicit schema name of the index does not exist
  - CREATEIN privilege on the schema, if the schema name of the index exists
- DBADM authority

### **Default PUBLIC privilege**

None

## Syntax

► SYSTS\_CREATE (—*index\_schema*—, —*index\_name*—, —| text source |—, —►  
 ► | options |—, —*message\_locale*—, —*message*—) —►

### text source:

| —| *table-name*— (—| text column name |—), —►  
 | —| *table-schema*— |

### text column name:

| —| *column-name*— |  
 | —| *function-name* (*column-name*)— |

### options:

text default information
update characteristics
storage options
index configuration options
partition options

### text default information:

| —| CODEPAGE—*code-page*— | —| LANGUAGE—*language*— | —| FORMAT—*format*— |

### update characteristics:

| —| UPDATE FREQUENCY—NONE— | —| update frequency | —►  
 ► | incremental update characteristics | —►

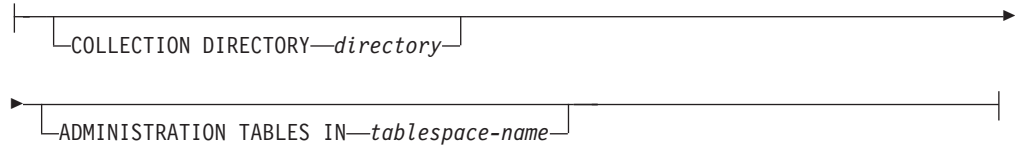
### update frequency:

| —D— (—| \*— |—) —| —H— (—| \*— |—) —| —M— (—| *integer3*— |—) —►  
 | —| *integer1*— | —| *integer2*— |

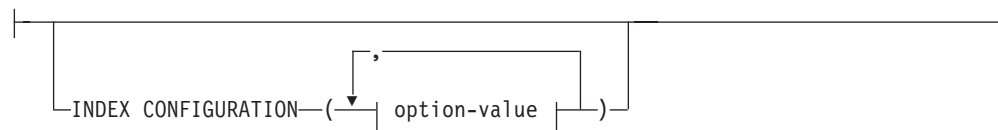
### incremental update characteristics:



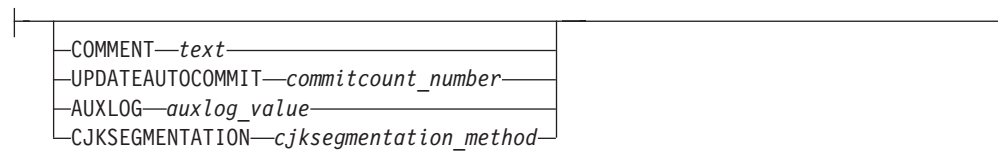
### storage options:



### index configuration options:



### option-value:



The schema is SYSPROC.

## Procedure parameters

### *index\_schema*

An input argument of type VARCHAR(128) that specifies the schema of the text search index. The *index\_schema* must comply with the naming restrictions for DB2 schema names. If the argument is null or an empty string, the default value of schema is used. The *index\_schema* is case-sensitive.

### *index\_name*

An input argument of type VARCHAR(128) that specifies the name of the index. Together with *index\_schema*, *index\_name* uniquely identifies a text search index in a database. The *index\_name* is case-sensitive.

### text source

An input argument of type VARCHAR(1024) that specifies the name of the column to be indexed. The options are as follows:

#### *table-schema*

The schema of the table for which the text search index is to be created.

#### *table-name*

Specifies the name of the table containing the text column. You cannot create text search indexes on federated tables, materialized query tables, or views. The *table-name* is case-sensitive.



**text column name**

Specifies the name of the column to be indexed.

*column-name*

Specifies the name of the column to be indexed. The column must be of one of the following data types: CHAR, VARCHAR, CLOB, DBCLOB, BLOB, GRAPHIC, VARGRAPHIC, or XML. If the data type of the column is not one of these data types, you can use a transformation function with the name *function-schema.function-name* to convert the column type to one of the valid types. (For details, see the *function-name (column-name)* parameter.) Alternatively, you can specify a user-defined external function that accesses the text documents to be indexed. You can create only a single text search index for a column. The *column-name* is case-sensitive.

*function-name (column-name)*

Specifies the schema-qualified name of an external scalar function that converts a column data type that is unsupported for text searching into a data type that is supported for text searching. This function must take only one parameter and return only one value. The name of the function must conform to DB2 naming conventions. The *function-name(column-name)* parameter is case-sensitive.

**options**

An input argument of type VARCHAR(32) that specifies the options to be used. If you do not require any options, the parameter can be null or an empty string. The options are as follows:

**CODEPAGE** *code-page*

Specifies the DB2 code page to be used when indexing text documents. The default code page value is taken from the SYSIBMTS.TSDEFAULTS view, from the row with the DEFAULTNAME column value of CODEPAGE. The initial default code page for new indexes is the database code page. This parameter applies only to binary data types that is, the column type or return type from a transformation function must be BLOB or character-type FOR BIT DATA.

**LANGUAGE** *language*

Specifies the language to be used by DB2 Text Search for language-specific processing of a document during indexing. If you do not specify a locale, the database territory is used to determine the default setting for the **LANGUAGE** parameter. To have your documents scanned to determine the locale, specify *locale* as AUTO.

**FORMAT** *format*

Specifies the format of the text documents in the column. The supported formats are TEXT, XML, HTML, and INSO. If the column data type is not XML, the default format is taken from the SYSIBMTS.TSDEFAULTS view, from the row with the DEFAULTNAME column value of FORMAT. If the column data type is XML, the default format is always XML. If you want to use the INSO format, rich text support must be installed.

**UPDATE FREQUENCY**

Specifies the frequency of index updates. The index is updated if the number of changes is at least the value set for the **UPDATE MINIMUM** parameter. Automatic updates can only be done if the DB2\_ATS\_ENABLE registry variable is set and the **START FOR TEXT** command is issued.

The default format is taken from the SYSIBMTS.TSDEFAULTS view, from the row with the DEFAULTNAME column value of UPDATEFREQUENCY.

**NONE**

No further index updates are made. This value can be useful for a text column in a table with data that will not change. It is also useful if you intend to manually update the index by using the **UPDATE INDEX** command.

**D** The days of the week when the index is updated.

\* Every day of the week.

*integer1*

Specific days of the week, from Sunday - Saturday: 0 - 6.

**H** The hours of the specified days when the index is updated.

\* Every hour of the day.

*integer2*

Specific hours of the day, from midnight to 11 p.m.: 0 - 23.

**M** The minutes of the specified hours when the index is updated.

*integer3*

**UPDATE MINIMUM** *minchanges*

Specifies the minimum number of changes to text documents before the index is updated incrementally at the time specified by the **UPDATE FREQUENCY** parameter. Only positive integer values are allowed. The default value is taken from the SYSIBMTS.TSDEFAULTS view, from the row with the DEFAULTNAME column value of UPDATEMINIMUM. A small value increases consistency between the table column and the text search index but also causes additional load on the system. The **UPDATE INDEX** command ignores the value of this parameter unless you specify the **USING UPDATE MINIMUM** parameter for that command.

**COLLECTION DIRECTORY** *directory*

Specifies the directory in which the text search index collection is stored. You must specify the absolute path, where the maximum length of the absolute path name is 215 characters. The process owner of the Text Search server instance service must have read and write access on this directory.

The **COLLECTION DIRECTORY** parameter is only supported for an integrated text search server setup. Review the usage notes for additional information about collection locations.

**ADMINISTRATION TABLES IN** *table-space-name*

Specifies the name of an existing non-temporary table space for the administration tables created for the index. By default the table space of the base table for which you are creating the index is used.

This argument is required when creating a text index for a partitioned table or for tables in partitioned databases. For partitioned databases, the table space must be in the same partition group as the table space of the base table. The text index administration tables are distributed in the same manner as the corresponding base table.

**INDEX CONFIGURATION** (*option-value*)

Specifies additional index-related options as option-value pairs. The following options are supported.

Table 14. Option-value pairs for the INDEX CONFIGURATION parameter

Option	Value	Data type	Description
COMMENT	<i>text</i>	String value of fewer than 512 bytes	Adds a string comment to the REMARKS column in the DB2 Text Search SYSIBMTS.TSINDEXES catalog view. The comment is also used as the description of the collection.
UPDATEAUTOCOMMIT	<i>commitcount_number</i>	Integer	<p>Specifies the number of index updates after which a commit is executed to preserve the previous work, for initial or incremental updates:</p> <ul style="list-style-type: none"> <li>• For initial updates, the index update processes batches of documents from a base table after the trigger to capture data updates is activated. After the number of updated documents reaches the <i>commitcount_number</i> number, the server does an interim commit. Log entries that are generated by unprocessed documents are removed from the staging table. Using the UPDATEAUTOCOMMIT option for an initial text index update significantly increases execution time.</li> <li>• For incremental updates, log entries that were processed are removed from the staging table with each interim commit. The <i>commitcount_number</i> number counts the number of documents that are updated, not the number of staging table entries.</li> </ul>

Table 14. Option-value pairs for the INDEX CONFIGURATION parameter (continued)

Option	Value	Data type	Description
AUXLOG	<i>auxlog_value</i>	String	Controls the creation of the additional log infrastructure to capture changes that are not recognized by a trigger. The default setting for range-partitioned tables is ON. You can change the default value in the default table by setting AuxLogNorm for non-range-partitioned tables and AuxLogPart for range-partitioned tables.  You cannot change the auxiliary log infrastructure property for a text index after creating the index.
CJKSEGMENTATION	<i>CJKSEGMENTATION_method</i>	String	The segmentation method is applicable for documents in Chinese, Japanese, and Korean languages (zh_CN, zh_TW, ja_JP, ko_KR locale sets), including such documents when automatic language detection is enabled (LANGUAGE AUTO). If no option is specified, the value for CJKSEGMENTATION in the defaults table is applied. Supported values are: <ul style="list-style-type: none"> <li>• MORPHOLOGICAL</li> <li>• NGRAM</li> </ul> The specified segmentation method is added to the SYSIBMTS. TSCONFIGURATION administrative view and cannot be changed after the text index is created.

**Remember:** Non-numeric values, such as comments, must be enclosed in single quotation marks. A single quote character within a string value must be represented by two consecutive single quotation marks.

**Example:**

```
INDEX CONFIGURATION (COMMENT 'Index on User''s Guide column')
```

*message\_locale*

An input argument of type VARCHAR(33) that specifies the locale to be used for any error message returned. If the argument is null or an empty string or the message files for the specified locale are not available on the server, 'en\_US' is used.

*message*

An output argument of type VARCHAR(32) that specifies a warning or informational message for a successfully completed operation.

## Examples

*Example 1:* In the following example, the SYSTS\_CREATE procedure is called to create a text search index called myTextIndex. The **UPDATE MINIMUM 10** parameter specifies that 10 changes must be made to the text documents that are associated with the index before an incremental update of the index is performed. Any error messages are returned in English. When the underlying text search command runs successfully, the **message** output parameter is set to indicate the status of the command execution.

```
CALL SYSPROC.SYSTS_CREATE('db2ts', 'myTextIndex',
    'myUserSchema.myBaseTable (myTextColumn)', 'UPDATE MINIMUM 10',
    'en_US', ?)
```

Sample output is as follows:

```
Value of output parameters
-----
Parameter Name   : MESSAGE
Parameter Value  : Operation completed successfully.
Return Status    = 0
```

*Example 2:* In the following example, the SYSTS\_CREATE procedure is called to create a text search index called myTextIndex2. No options are specified. In this example, the index exists, which results in an error message being returned to the caller.

```
CALL SYSPROC.SYSTS_CREATE('db2ts', 'myTextIndex2',
    'myUserSchema.myBaseTable (myTextColumn)', '', 'en_US', ?)
```

Sample output is as follows:

```
SQL20427N An error occurred during a text search administration
procedure or command. The error message is "CIE00201 Text search
index "db2ts"."myTextIndex2" already exists. "
```

## Usage notes

- Text search administration procedures use an existing connection to the database. If a previous transaction is not committed before executing a text search administration operation, you might run into a deadlock, as the same database objects may be affected and the operation waits for the previous step to commit. This may occur, for example, when AUTOCOMMIT is turned off and a table is created followed by creating a text index without an explicit commit between the two transactions.
- Without the COLLECTION DIRECTORY clause the collection will be placed in a subdirectory named after the system-generated index identifier in the path defined via the *defaultDataDirectory* parameter in the Text Server configuration. The configTool utility can be used to explicitly configure this parameter. If the *defaultDataDirectory* parameter is not explicitly configured, then the collection subdirectory is located in <configPath>/config/collections directory. It is strongly recommended to use a location other than the home directory or the database instance path as *defaultDataDirectory* or COLLECTION DIRECTORY. Make sure the location has adequate storage space and is local (not NFS mounted).
- Configure a value for the **COLLECTION DIRECTORY** parameter which enables use of a striped RAID device.

The following key-related rules apply:

- You must define a primary key for the table.
- The number of primary key columns is limited to two columns fewer than the number of primary key columns that are allowed in other DB2 environments.
- The total length of all primary key columns for a table with DB2 Text Search indexes is limited to 15 bytes fewer than the maximum total primary key length that is allowed in other DB2 environments. See the restrictions for the DB2 CREATE INDEX statement.

Creating an index with the **LANGUAGE** parameter set to the AUTO option allows CJKSEGMENTATION specification as an option. The specified segmentation method applies to Chinese, Japanese, and Korean language documents. You cannot change the value set for the *CJKSEGMENTATION\_method* value once index creation is complete.

Certain procedures or commands cannot be executed concurrently on a text search index because they might result in an error message depending on the timing of the conflicting operation.

- SYSTS\_DISABLE procedure or **db2ts DISABLE DATABASE FOR TEXT** command
- SYSTS\_CONFIGURE procedure

Successful execution of the **CREATE INDEX** command has the following effects:

- DB2 Text Search server data is updated. A collection with the name *instance\_database-name\_index-identifier\_number* is created, as in the following example:

```
tigertail_MYTSDB_TS250517_0000
```

For partitioned databases, a collection is created for each partition. You can retrieve the collection name(s) from the SYSIBMTS.TSCOLLECTIONNAMES view COLLECTIONNAME column.

- An index event table is created in the specified table space. Also, an index staging table is created in the specified table space with appropriate DB2 indexes. If the AUXLOG ON option was set for the **INDEX CONFIGURATION** parameter, a second staging table is created to capture changes through integrity processing.
- DB2 Text Search catalog information is updated. However, the newly created text search index is not automatically populated. To populate the text search index, the **SYSTS\_UPDATE** procedure or **UPDATE INDEX** command must be issued either manually or automatically (as a result of defining an update schedule for the index by using the **UPDATE FREQUENCY** parameter).
- If DB2 Text Search coexists with DB2 Net Search Extender and an active Net Search Extender index exists for the table column, the new DB2 Text Search index is set to inactive.
- If a text search index is created with parameters **LANGUAGE** set to AUTO and **CJKSEGMENTATION** set to MORPHOLOGICAL, then searches for valid strings on a morphological index returns zero rows. To obtain the results, add the QUERYLANGUAGE option to the **CONTAINS** function, as seen in the following query:

```
select bookname from morphobooks
where contains (story, '△', 'QUERYLANGUAGE=zh_CN') = 1
```

---

## **SYSTS\_DISABLE procedure - Disable current database for text search**

The procedure disables DB2 Text Search for the current database.

Once the Text Search feature has been disabled, text search indexes and commands are no longer available for use with the database.

The procedure issues a **DISABLE DATABASE FOR TEXT** text search administration command on the database server.

## Authorization

The privileges held by the authorization ID of the statement must include the following authorities:

- DBADM with DATAACCESS authority.
- SYSTS\_ADM role

## Default PUBLIC privilege

None

## Syntax

```
►►—SYSTS_DISABLE—(—options—,—message_locale—,—message—)—————►►
```

The schema is SYSPROC.

## Procedure parameters

### *options*

An input argument of type VARCHAR(128) that specifies the options to be used when disabling the database. The argument can be set to FORCE. When this value is specified, all indexes are dropped and the Text Search feature is disabled by force. No text search indexes are preserved and no error message or warning is returned. If the argument is null or an empty string, an attempt is made to disable the Text Search feature for the database.

### *message\_locale*

An input argument of type VARCHAR(33) that specifies the locale to be used for any error message returned. If the argument is null or an empty string, or the message files for the specified locale are not available on the server, 'en\_US' is used.

### *message*

An output argument of type VARCHAR(32K) that specifies a warning or informational message for a successfully completed operation.

## Examples

*Example 1:* In the following example, Text Search is disabled for a database using the SYSTS\_DISABLE procedure. The FORCE option is specified to ensure that the feature is disabled even if text search indexes still exist on tables in the database. Error messages are specified to be returned in English. The *message* output parameter is set to an informational message string.

```
CALL SYSPROC.SYSTS_DISABLE('FORCE', 'en_US', ?)
```

The following output is an example of sample output from this query.



```

Value of output parameters
-----
Parameter Name : MESSAGE
Parameter Value : Operation completed successfully.

Return Status = 0

```

*Example 2:* In the following example, Text Search is disabled for a database with existing text search indexes using the SYSTS\_DISABLE procedure without specifying the FORCE option. This results in an error message to the caller. It is preferable to drop all existing text search indexes before disabling the Text Search feature or alternatively to specify the FORCE option for the *options* input parameter value.

```
CALL SYSPROC.SYSTS_DISABLE('', 'en_US', ?)
```

The following output is an example of sample output from this query.

```
SQL20427N An error occurred during a text search administration
procedure or command. The error message is "CIE00326 Text search
index active in specified or default database. ". SQLSTATE 38H14
```

## Usage notes

- Text search administration procedures use an existing connection to the database. It is recommended to commit all transaction changes before executing a text search administration procedure to avoid any unexpected impact from a commit or rollback in the procedure. One way to achieve this is to turn on AUTOCOMMIT.
- Multiple procedures or commands cannot be executed concurrently on a text search index if they might conflict. Some of the conflicting procedures are:
  - SYSTS\_ALTER procedure
  - SYSTS\_CLEAR\_EVENTS procedure
  - SYSTS\_DISABLE procedure
  - SYSTS\_CONFIGURE procedure
  - SYSTS\_UPDATE procedure

If there is a conflict, the procedure returns an SQLCODE -20426 and SQLSTATE 38H13.

**Note:** The lock is set at the database level for the **SYSTS\_DISABLE** procedure.

- When this procedure is run,
  - the DB2 Text Search catalog information is updated. The index log and event tables are dropped. Triggers on the user text table are deleted.
  - if the **FORCE** option is specified, all text index information is removed from the database and all associated collections are deleted. See the "db2ts DROP INDEX command" or "SYSTS\_DROP procedure" for reference.
- This procedure does not influence the DB2 Net Search Extender enablement status of the database. It deletes the DB2 Text Search catalog tables and views that are created by the SYSTS\_ENABLE procedure or the **ENABLE FOR TEXT** command.
- Before dropping a DB2 database that has text search index definitions, run this procedure and make sure that the text indexes and collections have been removed successfully.
- If some indexes could not be deleted using the **FORCE** option, the collection names are written to the **db2diag** log files.



**Note:** The user is discouraged from usage that results in orphaned collections, that is, collections that remain defined on the text search server but are not used by DB2. Here are some cases that may cause orphaned collections:

- When a DROP DATABASE CLP command is executed without running a DISABLE DATABASE FOR TEXT command
- When the SYSTS\_DISABLE procedure is run or a **DISABLE DATABASE FOR TEXT** command is executed using the **FORCE** option.

---

## SYSTS\_DROP procedure - Drop text search index

The SYSTS\_DROP procedure drops an existing text search index associated with any table column.

After this procedure runs successfully, text search queries cannot be run on the column for which the text search index was dropped.

The procedure issues a **DROP INDEX** text search administration command on the database server.

### Authorization

The privileges held by the authorization ID of the statement must include the SYSTS\_MGR role and at least one of the following privileges or authorities:

- CONTROL privilege on the table on which the index is defined
- DROPIN privilege on the schema on which the index is defined
- If the text search index has an existing schedule, the authorization ID must be the same as the index creator, or must have DBADM authority

### Default PUBLIC privilege

None

### Syntax

```
►► SYSTS_DROP (—index_schema—, —index_name—, —options—) ►►  
►—message_locale—, —message—) ►►
```

The schema is SYSPROC.

### Procedure parameters

#### *index\_schema*

An input argument of type VARCHAR(128) that specifies the schema of the text search index. The *index\_schema* must follow the naming restriction for DB2 schema names. If the argument is null or an empty string, the value of CURRENT SCHEMA is used. The *index\_schema* is case-sensitive.

#### *index\_name*

An input argument of type VARCHAR(128) that specifies the name of the index. Together with *index\_schema*, it uniquely identifies a text search index in a database. The *index\_name* is case-sensitive.

### *options*

An input argument of type VARCHAR(32000) that specifies the options to be used. If no options are needed, the argument can be null or an empty string.

### *message\_locale*

An input argument of type VARCHAR(33) that specifies the locale to be used for any error message returned. If the argument is null or an empty string, or the message files for the specified locale are not available on the server, 'en\_US' is used.

### *message*

An output argument of type VARCHAR(32K) that specifies a warning or informational message for a successfully completed operation.

## Examples

*Example 1:* In the following example, the text search index that was created with *index\_schema* 'db2ts' and *index\_name* 'myTextIndex' is being dropped. Any error messages are requested to be returned in English. When the procedure succeeds, the output parameter message indicative of the successful operation is returned to the caller.

```
CALL SYSPROC.SYSTS_DROP('db2ts', 'myTextIndex', '', 'en_US', ?)
```

As in previous releases, the SYSTS\_DROP procedure is supported without the **options** argument, for example :

```
CALL SYSPROC.SYSTS_DROP('db2ts', 'myTextIndex', 'en_US', ?)
```

The following is an example of output from this query.

```
Value of output parameters
-----
Parameter Name   : MESSAGE
Parameter Value  : Operation completed successfully.

Return Status = 0
```

*Example 2:* In the following example, SYSTS\_DROP is called to drop a text search index with *index\_schema* 'db2ts' and *index\_name* 'myTextIndex2'. This index does not exist and results in an error.

```
CALL SYSPROC.SYSTS_DROP('db2ts', 'myTextIndex2', 'en_US', ?)
```

The following is an example of output from this query.

```
SQL20427N An error occurred during a text search administration
procedure or command. The error message is "CIE00316 Text search
index "db2ts"."myTextIndex2" does not exist. ". SQLSTATE 38H14
```

## Usage notes

- Multiple procedures or commands cannot be executed concurrently on a text search index if they might conflict. Some of the conflicting procedures are:
  - SYSTS\_ALTER procedure
  - SYSTS\_CLEAR\_EVENTS procedure
  - SYSTS\_DISABLE procedure
  - SYSTS\_DROP procedure
  - SYSTS\_UPDATE procedure
  - SYSTS\_CONFIGURE procedure

A STOP FOR TEXT command that runs in parallel with the DROP operation will not cause a conflicting command message, instead, if the text search server is shut down before DROP has removed the collection, an error will be returned that the text search server is not available.

- After a text search index is dropped, text search is no longer possible on the corresponding text column. If you plan to create a new text search on the same text column, you must first disconnect from the database and then reconnect before creating the new text search index.
- The db2ts DROP INDEX command makes the following changes to the database:
  - Updates the DB2 Text Search catalog information.
  - Drops the index staging and event tables.
  - Deletes triggers on the user text table.
  - Destroys the collection associated with the DB2 Text Search index definition.

## SYSTS\_ENABLE procedure - Enable current database for text search

The SYSTS\_ENABLE procedure enables DB2 Text Search for the current database.

This procedure must be issued successfully before text search indexes on columns within the database can be created.

This procedure issues the **ENABLE DATABASE FOR TEXT** text search administration command on the database server.

### Authorization

The privileges held by the authorization ID of the procedure must include the SYSTS\_ADM role and the DBADM authority.

### Default PUBLIC privilege

None

### Syntax

```
►► SYSTS_ENABLE (—options—, —message_locale—, —message—) ◀◀
```

#### options:

```
►► ┌──────────────────────────────────────────────────────────────────────────┐ ◀◀  
    | ADMINISTRATION TABLES IN table-space-name |  
└──────────────────────────────────────────────────────────────────────────┘
```

The schema is SYSPROC.

### Procedure parameters

#### options

An input argument of type VARCHAR(32K) that specifies the options to be used. If no options are needed, the argument can be null or an empty string. Supported values are:

#### **ADMINISTRATION TABLES IN** *table-space-name*

Specifies the name of an existing regular table space for administration tables created while enabling the database for DB2 Text Search. If this

clause is not specified, SYSTOOLSPACE is used as the table space. Regardless of whether SYSTOOLSPACE or an explicitly specified table space name is used, the table space has to meet certain requirements. It has to be:

- A regular table space
- For partitioned databases it would be recommended to define a bufferpool and table space with 32KB page size

To use a case-sensitive tablespace-name use a delimited identifier by enclosing the name in double quotation marks. By default it is treated as an ordinary identifier and converted to uppercase.

#### *message\_locale*

An input argument of type VARCHAR(33) that specifies the locale to be used for any error message returned. If the argument is null or an empty string, or the message files for the specified locale are not available on the server, 'en\_US' is used.

#### *message*

An output argument of type VARCHAR(32K) that specifies a warning or informational message for a successfully completed operation.

## Examples

*Example 1:* Enable the database for text search by creating administration tables in a table space with any output messages in English.

```
CALL SYSPROC.SYSTS_ENABLE ('ADMINISTRATION TABLES IN TSPACE', 'en_US', ?)
```

Enable the database for text search by creating administration tables in a table space with any output messages in French.

```
CALL SYSPROC.SYSTS_ENABLE ('ADMINISTRATION TABLES IN "tbs32k" ', 'fr_FR', ?)
```

As in previous releases, the SYSTS\_ENABLE procedure is supported without the **options** argument, for example:

```
CALL SYSPROC.SYSTS_ENABLE ('en_US', ?)
```

An example of output for this query:

```
Value of output parameters
-----
Parameter Name  : MESSAGE
Parameter Value : Operation completed successfully.
```

```
Return Status = 0
```

*Example 2:* In the following example, SYSTS\_ENABLE is called on a database that is already enabled for text search. This results in an error message to the caller.

```
CALL SYSPROC.SYSTS_ENABLE('en_US', ?)
```

An example of output for this query:

```
SQL20427N An error occurred during a text search administration
procedure or command. The error message from the text search
product is "CIE00322 Specified or default database already
enabled for text. ". SQLSTATE 38H14
```

## Usage notes

- Text search administration procedures use an existing connection to the database. It is recommended to commit all transaction changes before executing

a text search administration procedure to avoid any unexpected impact from a commit or rollback in the procedure. One way to achieve this is to turn on AUTOCOMMIT.

- When this procedure is run, the following events occur:
  - This procedure creates database objects, such as text search administration catalog tables and views, in the schema SYSIBMTS.
  - The established database defaults for text search index are available in view SYSIBMTS.TSDEFAULTS.
  - When the command has successfully completed, the text search catalog tables and views are created and are available.
- When executing this procedure, if you do not have sufficient execution and file access privileges to retrieve the Text Search server configuration, the procedure will create the text search catalog with an 'Incomplete enablement' warning. In this case, the Text Search server connection information will need to be updated manually.

---

## SYSTS\_UPDATE procedure - Update the text search index

The **SYSTS\_UPDATE** procedure updates the text search index to reflect the current contents of the text column with which the index is associated.

While the update is being performed, a search is possible. Until completion of the update, the search operates on a partially updated index.

The procedure issues an **UPDATE INDEX** text search administration command on the database server.

### Authorization

The privileges held by the authorization ID of the statement must include the SYSTS\_MGR role and at least one of the following authorities:

- DATAACCESS authority
- CONTROL privilege on the table on which the text index is defined
- INDEX with SELECT privilege on the base table on which the text index is defined

In addition, for an initial update the authorization requirements apply as outlined in the **CREATE TRIGGER** statement.

### Default PUBLIC privilege

None

### Syntax

```
►►SYSTS_UPDATE(—index_schema—,—index_name—,——————►  
►—update_options—,—message_locale—,—message—)—————►◄
```

The schema is SYSPROC.

## Procedure parameters

### *index\_schema*

An input argument of type VARCHAR(128) that specifies the schema of the text search index. The *index\_schema* must follow the naming restriction for DB2 schema names. If the argument is null or an empty string, the value of CURRENT SCHEMA is used. The *index\_schema* is case-sensitive.

### *index\_name*

An input argument of type VARCHAR(128) that specifies the name of the index. Together with *index\_schema*, it uniquely identifies a text search index in a database. The *index\_name* is case-sensitive.

### *update\_options*

An input argument of type VARCHAR(32K) that specifies update options. If no options are needed the argument can be null or an empty string. The possible values are:

- USING UPDATE MINIMUM: This value respects the UPDATE MINIMUM settings from the **CREATE INDEX** text search administration command and the SYSTS\_CREATE procedure.
- FOR DATA REDISTRIBUTION: Specifies that a text search index in a partitioned database needs to be refreshed after data partitions were added or removed and a subsequent data redistribution operation completed. Search results may be inconsistent until the text search index was updated with the FOR DATA REDISTRIBUTION option.
- ALLROWS: Specifies that an initial update should be attempted unconditionally.
- NULL or an empty string (""): Specifies that the update is unconditionally started when the procedure is called

### *message\_locale*

An input argument of type VARCHAR(33) that specifies the locale to be used for any error message returned. If the argument is null or an empty string, or the message files for the specified locale are not available on the server, 'en\_US' is used.

### *message*

An output argument of type VARCHAR(32K) that specifies a warning or informational message for a successfully completed operation.

## Examples

*Example 1:* In the following example, the text search index that was created with *index\_schema* 'db2ts' and *index\_name* 'myTextIndex' is being updated. A NULL value in the place of the *update\_options* means that an update is unconditionally started when the stored procedure is called. Any error messages are requested to be returned in English. When the procedure succeeds, the output parameter message indicative of the successful operation is returned to the caller.

```
CALL SYSPROC.SYSTS_UPDATE
('db2ts', 'myTextIndex', '', 'en_US', ?)
```

An example of output from this query:

```
Value of output parameters
-----
Parameter Name : MESSAGE
Parameter Value : Operation completed successfully.
```

*Example 2:* Update a text index after an operation was executed on the partition group associated with the base table, and return any error messages in English.

```
CALL SYSPROC.SYSTS_UPDATE  
( 'db2ts', 'myTextIndex2', 'FOR DATA REDISTRIBUTION', 'en_US', ?)
```

Value of output parameters

-----

Parameter Name : MESSAGE

Parameter Value : Operation completed successfully.

Return Status = 0

*Example 3:* In the following example, SYSTS\_UPDATE is called to update a text search index with *index\_schema* 'db2ts' and *index\_name* 'myTextIndex3'. This index does not exist and results in an error.

```
CALL SYSPROC.SYSTS_UPDATE('db2ts', 'myTextIndex3', 'USING UPDATE MINIMUM',  
'en_US', ?)
```

```
SQL20427N An error occurred during a text search administration  
procedure or command. The error message is "CIE00316 Text search  
index "db2ts"."myTextIndex3" does not exist. ". SQLSTATE 38H14
```

## Usage notes

- Text search administration procedures use an existing connection to the database. The current transaction might be committed or rolled back depending on the completion of the procedures. To avoid any unexpected impact from such a commit or rollback, you might want to commit all transaction changes. Turning on AUTOCOMMIT is one way to commit all transaction changes.
  - Certain procedures or commands cannot be executed concurrently on a text search index because the timing of the conflicting operation might cause an error. Some of the conflicting procedures and commands are:
    - SYSTS\_ALTER procedure or **db2ts ALTER INDEX** command
    - SYSTS\_CLEAR\_EVENTS procedure or **db2ts CLEAR EVENTS FOR INDEX** command
    - SYSTS\_DISABLE procedure or **db2ts DISABLE DATABASE FOR TEXT** command
    - SYSTS\_UPDATE procedure or **db2ts UPDATE INDEX** command
- If there is a conflict, the procedure returns an SQLCODE -20426 and SQLSTATE 38H13.
- This procedure does not return until all index update processing is completed. The duration depends on the number of documents to be indexed and the number of documents already indexed. The collection name for the index can be retrieved from the SYSIBMTS.TSCOLLECTIONNAMES view COLLECTIONNAME column.
  - When there are individual document errors, the documents must be corrected. The primary keys of the erroneous documents can be looked up in the event table for the index. By changing the corresponding rows in the user table, the next call to SYSTS\_UPDATE reprocesses these documents.
  - When the SYSTS\_UPDATE procedure is run, the following events occur:
    - Rows are inserted into the event table, including parser error information. Information is deleted from the index staging table in case of incremental updates. Before the first update, the SYSTS\_UPDATE procedure creates triggers on the user table.
    - The collection is updated:
      - New or changed documents are parsed and indexed.
      - Deleted documents are discarded from the index.

- If a synonym dictionary has been associated with a text index, executing the update with the ALLROWS or FOR DATA REDISTRIBUTION options removes the association by dropping and recreating the collections for the text index. The synonym dictionary must be associated with the new text index collections.

---

## SYSTS\_UPGRADE\_CATALOG procedure - Upgrade the text search catalog

This procedure upgrades the DB2 Text Search catalog, including the administrative tables and administrative views, to the latest product version.

This procedure creates new catalog tables and views which are used in the latest version of the product and also updates the existing catalog tables and views. It removes obsolete catalog tables and views.

### Authorization

The privileges held by the authorization ID of the procedure must include the SYSTS\_ADM role and the DBADM authority.

### Default PUBLIC privilege

None

### Syntax

```
►►—SYSTS_UPGRADE_CATALOG—(—message_locale—,—message—)—————►◄
```

The schema is SYSPROC.

### Procedure parameters

#### *message\_locale*

An input argument of type VARCHAR(33) that specifies the locale to be used for any error message returned. If the argument is null or an empty string, or the message files for the specified locale are not available on the server, 'en\_US' is used.

#### *message*

An output argument of type VARCHAR(32K) that specifies a warning or informational message for a successfully completed operation.

### Example

*Example 1:* In the following example, the database was enabled for text search in an older release. Calling the **SYSTS\_UPGRADE\_CATALOG** procedure upgrades the text search catalog tables, and if the procedure succeeds, the output parameter message indicating successful operation is returned to the caller.

```
CALL SYSPROC.SYSTS_UPGRADE_CATALOG('en_US',?)
```

```
Value of output parameters
```

```
-----
Parameter Name : MESSAGE
Parameter Value : CIE0213W The DB2 Text Search Catalog has been upgraded to
the current version. You will now need to update the text search index using
```



the SYSPROC.SYSTS\_UPGRADE\_INDEX stored procedure.  
index also needs to be updated by calling "SYSPROC.SYSTS\_UPGRADE\_INDEX".

Return Status = 0

*Example 2:* In the following example, the database was not enabled for DB2 Text Search in an older release before calling the procedure. A NULL value in the place *message\_locale* means the a default locale of 'en\_US' will be used. All error messages are returned in English.

```
CALL SYSPROC.SYSTS_UPGRADE_CATALOG('','?')
SQL20427N An error occurred during a text search administration procedure
or command. The error message is "CIE0323E Specified or default database
not enabled for text. ".
SQLSTATE=38H14
```

*Example 3:* In the following example, the text search catalog and the text search indexes are already upgraded to the current version.

```
CALL SYSPROC.SYSTS_UPGRADE_CATALOG('en_US',?)
Value of output parameters
-----
Parameter Name : MESSAGE
Parameter Value : CIE0002I The DB2 Text Search release level is current
for the database. The system has not been upgraded.
```

Return Status = 0

## Usage notes

The **SYSTS\_UPGRADE\_CATALOG** procedure is integrated into the **DB2 UPGRADE DATABASE** command to perform the text search specific catalog upgrade. If the database upgrade fails to upgrade the text search catalog, the **SYSTS\_UPGRADE\_CATALOG** procedure must be executed separately by the user to complete the database upgrade.

To upgrade the text search index catalog, proceed as follows:

1. Make sure that the DB2 Text Search instance service is stopped.
2. Execute the **SYSTS\_UPGRADE\_CATALOG** procedure.

**Note:** DB2 Text Search administrative procedures use an existing connection to the database. The current transaction might be committed or rolled back depending on the completion of the procedures. As such, you might want to commit all transaction changes to avoid any unexpected impact from such a commit or rollback. One way to achieve this is to turn on **AUTOCOMMIT**.

3. The **SYSTS\_UPGRADE\_CATALOG** procedure attempts to populate the Text Search server information in the catalog. Review the SYSIBMTS.TSSERVER content and update the Text Search server information as necessary.
4. Upgrade the text search indexes by following the procedure outlined for the **SYSTS\_UPGRADE\_INDEX** procedure. The version value in the SYSIBMTS.TSDEFAULTS administrative view will not be updated until the **SYSTS\_UPGRADE\_INDEX** procedure is executed successfully.

---

## SYSTS\_UPGRADE\_INDEX - Upgrade text search indexes

This procedure updates DB2 Text Search index information in DB2 catalog tables and text search catalog tables.

Text search index collections are managed by the Text Search server. See Upgrading DB2 Text Search for more information.

## Authorization

The privileges held by the authorization ID of the procedure must include the SYSTS\_ADM role and the DBADM authority.

## Default PUBLIC privilege

None

## Syntax

```
►►—SYSTS_UPGRADE_INDEX—(—message_locale—,—message—)—————►►
```

The schema is SYSPROC.

## Procedure parameters

### *message\_locale*

An input argument of type VARCHAR(33) that specifies the locale to be used for any error message returned. If the argument is null or an empty string, or the message files for the specified locale are not available on the server, 'en\_US' is used.

### *message*

An output argument of type VARCHAR(32K) that specifies a warning or informational message for a successfully completed operation.

## Example

Example 1: In the following example, the database was enabled for text search in an older release and the procedure **SYSTS\_UPGRADE\_CATALOG** has already been completed successfully. The procedure **SYSTS\_UPGRADE\_INDEX** will complete the upgrade for the text index metadata in the database catalog. When the procedure succeeds, the output parameter message indicative of the successful operation is returned to the caller.

```
CALL SYSPROC.SYSTS_UPGRADE_INDEX('en_US', ?)
Parameter Name : MESSAGE
Parameter Value : CIE00001 Operation completed successfully.
```

Return Status = 0

Example 2: In the following example, the database was not enabled for text search in the older release of the product. If a NULL value is set for *message\_locale*, it means the system locale will be used. If the system locale is not available then the default locale 'en\_US' will be used.

```
CALL SYSPROC.SYSTS_UPGRADE_INDEX('', ?)
SQL20427N An error occurred during a text search administration procedure
or command. The error message is "CIE0323E Specified or default database
not enabled for text. ".
SQLSTATE=38H14
```

Example 3: In the following example, the DB2 Text Search catalog and the text search indexes were already upgraded to the current version.

```
CALL SYSPROC.SYSTS_UPGRADE_INDEX('en_US', ?)
```

Value of output parameters

-----

Parameter Name : MESSAGE

Parameter Value : CIE0002I The DB2 Text Search release level is current for the database. The system has not been upgraded.

Return Status = 0

Example 4: In the following example, the DB2 Text Search catalog was not upgraded.

```
CALL SYSPROC.SYSTS_UPGRADE_INDEX('en_US', ?)
```

CIE0409E The DB2 Text Search catalog has not been upgraded to the current version.

## Usage notes

- The **SYSTS\_UPGRADE\_INDEX** procedure is integrated into the **DB2 UPGRADE DATABASE** command to perform upgrades for text search indexes. When issuing the **DB2 UPGRADE DATABASE** command, this procedure will be executed as well. If the upgrade of text search indexes fails, the procedure must be run manually.
- The **SYSTS\_UPGRADE\_CATALOG** procedure should be executed before **SYSTS\_UPGRADE\_INDEX** procedure. The version value in the **SYSIBMTS.TSDEFAULTS** administrative view will not be updated until the **SYSTS\_UPGRADE\_INDEX** procedure is executed successfully.
- DB2 Text search administrative procedures use an existing connection to the database. The current transaction might be committed or rolled back depending on the completion of the procedures. As such, you might want to commit all transaction changes to avoid any unexpected impact from such a commit or rollback. One way to achieve this is to turn on **AUTOCOMMIT**.



---

## Chapter 12. Text search administrative views

DB2 Text Search creates and maintains several administrative views that describe the text search indexes in a database and their properties.

Do not update any of these views unless specifically instructed to do so.

The following views reflect the current configuration of your system:

- Database-level views:
  - SYSIBMTS.TSDEFAULTS
  - SYSIBMTS.TSLOCKS
  - SYSIBMTS.TSSERVERS
- Index-level views:
  - SYSIBMTS.TSINDEXES
  - SYSIBMTS.TSCONFIGURATION
  - SYSIBMTS.TSCOLLECTIONNAMES
  - SYSIBMTS.TSEVENT\_#####
  - SYSIBMTS.TSSTAGING\_#####

---

### SYSIBMTS.TSDEFAULTS view

SYSIBMTS.TSDEFAULTS displays all the default values for all text search indexes in a database.

The default values are available as attribute-value pairs in this view.

Table 15. SYSIBMTS.TSDEFAULTS view

Column name	Data type	Nullable?	Description
DEFAULTNAME	VARCHAR (30)	NO	Database default parameters for text search
DEFAULTVALUE	VARCHAR (512)	NO	Values for database default parameters for text search

The following values are used as defaults for the db2ts **CREATE INDEX**, **ALTER INDEX**, **UPDATE INDEX**, and **CLEAR EVENTS FOR INDEX** commands:

- **AUXLOGNORM**: The staging infrastructure can be enabled for a text search index with explicit index configuration **AUXLOG ON**. Do not enable the extended text-maintained staging infrastructure for non-partitioned tables by default.
- **AUXLOGPART**: The staging infrastructure can be disabled for a text index with explicit index configuration **AUXLOG OFF**. By default, enable the extended text-maintained staging infrastructure for range-partitioned tables.
- **CJKSEGMENTATION**: Specifies the segmentation method to use when indexing documents for Chinese, Japanese and Korean languages. The supported value includes: **MORPHOLOGICAL** and **NGRAM**. The default value is **NGRAM**.
- **CODEPAGE**: The initial default code page for new indexes is the database code page.
- **DOCUMENTRESULTQUEUE SIZE**: This value is used to limit how much database memory is reserved per update operation for a collection. The default value is 30,000 while the range is 100 - 100,000. Note that on a multi-partition

setup, a single text index update that is configured for parallel execution will reserve memory space for each collection that needs an update.

- **FORMAT:** The initial default for the document format is plain text.
- **LANGUAGE:** The initial default for document indexing is en\_US.
- **MAXCONCURRENTUPDATES:** Controls the number of collection updates that can be executed in parallel at any given time. For multiple partition setups, the number of collections for each text index is determined according to the table distribution. However, only active partition updates count. The default is 8.
- **MAXCONCURRENTCOLLECTIONS:** Controls the number of collections that can be created. For a single-node database, the number of collections equals the number of text indexes, for multi-partition setups, the number of collections per text index matches the table distribution. The default is 160.
- **MAXDOCUMENTSIZEINMB:** Controls the size of documents that are accepted for processing. A text that exceeds the limit will result in a warning message in the event table. The value is 100.
- **UPDATEFREQUENCY:** The initial default for the update schedule for new indexes is **NONE**.
- **UPDATEMINIMUM:** The initial default for updating new indexes is 1, meaning that incremental updates can be done after every change.
- **UPDATEAUTOCOMMIT:** The initial default for updating new indexes is 0, meaning that there will be no intermediate commits when documents are read from DB2 text columns. This value is reserved, and you cannot change it.

You cannot use **db2ts** commands to change the default values at the database level.

---

## SYSIBMTS.TSLOCKS view

You can view command lock information at the database and index level using SYSIBMTS.TSLOCKS.

Table 16. SYSIBMTS.TSLOCKS view

Column name	Data type	Nullable?	Description
COMMAND	VARCHAR(30)	NO	Name of the command that created the lock. Possible values are: CREATE INDEX, ALTER INDEX, DROP INDEX, UPDATE INDEX, CLEAR EVENTS, DISABLE DATABASE, CONFIGURE, CLEANUP
LOCKSCOPE	VARCHAR(30)	NO	Scope of the lock. Possible values are: DATABASE or INDEX.
INDSCHEMA	VARCHAR(128)	NO	Schema name of the text search index (only for LOCKSCOPE = INDEX)
INDNAME	VARCHAR(128)	NO	Unqualified name of the text search index (only for LOCKSCOPE = INDEX)
PARTITION	INTEGER	NO	Partition number on which the text search lock is created
LOCKCREATETIME	TIMESTAMP	NO	Time stamp when the lock was granted

There are three distinct scenarios to be aware of for locking strategies:

- **An operation is started and no applicable lock is encountered:** The procedure sets the lock and continues execution. For both successful and failed execution, the lock is removed.
- **An operation is started and encounters an applicable lock:** The request is returned with a conflicting command message.

- An operation is started and encounters an applicable lock, even though no associated operation is currently running: A failure occurred for an earlier operation that prevented proper removal of the lock. This can occur in extreme situations like disk failures or crashes. In such a case the locks need to be removed by issuing a **CLEAR COMMAND LOCKS** operation at the index or database level as appropriate, after the cause of failure is addressed and system consistency is verified.

---

## SYSIBMTS.TSSERVERS view

Each row represents of the SYSIBMTS.TSSERVERS view displays information about a DB2 Text Search server configured for the database.

You can query the view to obtain information about the text search server that is marked as the one to be used:

```
db2 "SELECT SERVERID, HOST from SYSIBMTS.TSSERVERS where SERVERSTATUS = 0"
```

Table 17. SYSIBMTS.TSSERVERS view

Column name	Data type	Nullable?	Description
SERVERID	INTEGER	NO	Unique ID generated for the text search server.
HOST	VARCHAR(256)	NO	Host name or IP address of the text search server. For partitioned databases, stand-alone text search server deployments or when administrative operations are executed from remote clients, make sure to use the actual host name or IP address, not 'localhost'.
PORT	INTEGER	NO	Port number for the text search server. (ADMIN/SEARCH)
TOKEN	VARCHAR(256)	NO	Authentication token for the text search server.
KEY	VARCHAR(128)	NO	The server key for the text search server.
DEFAULTLOCALE	VARCHAR(33)	NO	Default client locale assumed for messages from text search server
SERVERTYPE	INTEGER	NO	The value indicates the type for each text search server. <ul style="list-style-type: none"> <li>• 0 = the default (integrated) text search server</li> <li>• non-zero value = a stand-alone text search server <ul style="list-style-type: none"> <li>- 1 = a local stand-alone text search server</li> <li>- 2 = a remote stand-alone text search server</li> </ul> </li> </ul>
SERVERSTATUS	INTEGER	NO	Indicates whether the text search server can be used to create new text search indexes. The default value is 0, indicating that the server is active and usable.

---

## SYSIBMTS.TSINDEXES view

The current text search index properties are shown in the SYSIBMTS.TSINDEXES view.

The following example uses the index schema and name:

```
db2 "SELECT COLNAME from SYSIBMTS.TSINDEXES where INDSHEMA=schema-name
and INDNAME=index-name"
```

The SYSIBMTS.TSINDEXES view is described in the following table.

Table 18. SYSIBMTS.TSINDEXES view

Column name	Data type	Nullable?	Description
INDSCHEMA	VARCHAR(128)	NO	Schema name for the text search index.
INDNAME	VARCHAR(128)	NO	Unqualified name of the text search index.
TABSCHEMA	VARCHAR(128)	NO	Schema name of the base table.
TABNAME	VARCHAR(128)	NO	Unqualified name of the base table.
COLNAME	VARCHAR(128)	NO	Column that the text search index was created on.
CODEPAGE	INTEGER	NO	Document code page for the text search index.
LANGUAGE	VARCHAR(5)	NO	Document language for the text search index.
FORMAT	VARCHAR(30)	YES	Document format.
FUNCTIONSCHEMA	VARCHAR(128)	YES	Schema for the column type.
FUNCTIONNAME	VARCHAR(18)	YES	Name of the column-type conversion function.
COLLECTIONDIRECTORY	VARCHAR(512)	YES	Directory for the text search index files.
UPDATEFREQUENCY	VARCHAR(300)	NO	Trigger criterion for applying updates to the index.
UPDATEMINIMUM	INTEGER	YES	Minimum number of entries in the log table before an incremental update is performed. A lower value means better consistency between the table column and the text search index. However, a lower value also increases the resources required for text search indexing.
EVENTVIEWSHEMA	VARCHAR(128)	NO	Schema for the event view that is created for the text search index (always SYSIBMTS).
EVENTVIEWNAME	VARCHAR(128)	NO	Name of the event view that is created for the text search index.
STAGINGVIEWSHEMA	VARCHAR(128)	YES	Schema for the log view that is created for the text search index (always SYSIBMTS).
STAGINGVIEWNAME	VARCHAR(128)	YES	Name of the log view that is created for the text search index.
REORGAUTOMATIC	INTEGER	YES	Reserved (not supported in this release). The value is always 1.
RECREATEONUPDATE	INTEGER	NO	Reserved (not supported in this release). The value is always 0.
ATTRIBUTES	VARCHAR(32000)	YES	Reserved (not supported in this release).
COLLECTIONNAMEPREFIX	VARCHAR(128)	NO	Prefix of the collection name on the text search server.



Table 18. SYSIBMTS.TSINDEXES view (continued)

Column name	Data type	Nullable?	Description
COMMENT	VARCHAR(512)	YES	Comment that is specified for a parameter that is related to index properties of the <b>CREATE INDEX</b> command.
AUXSTAGINGSHEMA	VARCHAR(48)	YES	Schema of the text-maintained staging table.
AUXSTAGINGNAME	VARCHAR(48)	YES	Name of the text-maintained staging table.
INDSTATUS	VARCHAR(10)	NO	Index status: <ul style="list-style-type: none"> <li>• ACTIVE indicates an active index.</li> <li>• INACTIVE indicates an inactive index. (This value is not used for DB2 Text Search.)</li> <li>• INVALID indicates an invalidated index, usually a side effect of a DB2 operation.</li> </ul>

## SYSIBMTS.TSCONFIGURATION view

Information about index configuration parameters is available in the SYSIBMTS.TSCONFIGURATION view.

Each row represents a configuration parameter of the text search index.

Following is an example of a query against the view that uses the index name:

```
db2 "SELECT VALUE from SYSIBMTS.TSCONFIGURATION where INDSHEMA=schema-name
and INDNAME=ind-name and PARAMETER ='parameter'"
```

Table 19. SYSIBMTS.TSCONFIGURATION view

Column name	Data type	Nullable?	Description
INDSCHEMA	VARCHAR(128)	NO	Schema name of the text search index
INDNAME	VARCHAR(128)	NO	Unqualified name of the text search index
PARAMETER	VARCHAR(30)	NO	Name of a configuration parameter
VALUE	VARCHAR(512)	NO	Value of the parameter

The PARAMETER column contains the names of the text search index configuration parameters specified with the CREATE INDEX statement and the names of some of the parameters from the SYSIBMTS.TSDEFAULTS view.

## SYSIBMTS.TSCOLLECTIONNAMES view

The SYSIBMTS.TSCOLLECTIONNAMES view displays the names of collections.

Each row represents a collection for a text search index.

Table 20. SYSIBMTS.TSCOLLECTIONNAMES view

Column name	Data type	Nullable?	Description
INDSCHEMA	VARCHAR(128)	NO	Schema name of the text search index

Table 20. SYSIBMTS.TSCOLLECTIONNAMES view (continued)

Column name	Data type	Nullable?	Description
INDNAME	VARCHAR(128)	NO	Unqualified name of the text search index
COLLECTIONNAME	VARCHAR(132)	NO	Name of the associated collection on the text search server. In partitioned database systems, each text index partition is represented as a collection. The collection name includes the partition number as suffix.

## SYSIBMTS.TSEVENT view

The event view provides information about indexing status and error events.

A database might have multiple views with the prefix SYSIBMTS.TSEVENT. Each view is differentiated by the *nnnnnn* value, an internal identifier that points to the corresponding text index that the view is associated with. To determine the text search index associated with a particular view, query the view SYSIBMTS.TSINDEXES, searching for the schema name and view name in the columns EVENTVIEWSHEMA and EVENTVIEWNAME. The query returns a single row that describes the text search index and user table in question.

The number of columns in this view depends on the number of primary key columns in the user table. The columns PK1..PK*n* match the primary key columns of the user table and have corresponding data type and lengths definitions. The data type of each of the columns in the view exactly corresponds to the data type of the corresponding primary key column.

Each row in this view represents a message from an **UPDATE INDEX** command on the text search index. For instance, a row might indicate that an **UPDATE INDEX** command has started or has completed. Alternatively, a row might describe a problem that occurred when a text document was being indexed. You can identify the text document by retrieving the primary key column values from the row in this view and looking them up in the user table.

You can clear events by using the db2ts **CLEAR EVENTS FOR INDEX** command.

Table 21. Event view

Column name	Data type	Nullable?	Description
OPERATION	INTEGER	YES	The operation (insert, update, or delete) on the base table to be reflected in the text search index
TIME	TIMESTAMP	YES	Time stamp of event entry creation
SEVERITY	INTEGER	YES	If the message corresponds to a single document, one of the following values: <ul style="list-style-type: none"> <li>• 1 = Informational</li> <li>• 4 = Parts of the document were indexed but there was a warning, as indicated by the message</li> <li>• 8 = The document was not indexed, as indicated by the message</li> <li>• 0= Otherwise</li> </ul>
SQLCODE	INTEGER	YES	SQLCODE for the associated error, if any
MESSAGE	VARCHAR(1024)	YES	Text information about the specific error
PARTITION	INTEGER	YES	Reserved for internal IBM use.

Table 21. Event view (continued)

Column name	Data type	Nullable?	Description
PK01	Data type of the first primary key column of the base table	YES	Value of the first primary key column of the base table of the text search index for the row being processed when the event occurred
...	...	...	...
PK $nn$	Data type of the last primary key column of the base table	YES	Value of the last primary key column of the base table of the text search index for the row being processed when the event occurred

Informational events, such as starting, committing, and finishing update processing are also available in this view. In this case, PK01, PK $nn$  and OPERATION all have NULL values. The code page and the locale of MESSAGE correspond to the database settings.

---

## SYSIBMTS.TSSTAGING view

The staging table stores the change operations on the user table that requires synchronization with the text search index.

Triggers are created on the user table to feed change information into the staging table. In addition, with the auxiliary log option, integrity processing detects changes to the user table. The **UPDATE INDEX** command reads the entries and deletes them after successful synchronization.

The database might have multiple views with the prefix SYSIBMTS.TSSTAGING\_. Each view is differentiated by the  $nnnnnn$  value, an internal identifier that points to the corresponding text index that the view is associated with. To determine the text search index associated with a particular view, query the view SYSIBMTS.TSINDEXES, searching for the schema name and view name in the columns STAGINGVIEWSHEMA and STAGINGVIEWNAME. The query returns a single row that describes the text search index and user table in question.

The number of columns in this view depends on the number of primary key columns in the user table. The columns PK1..PK $nn$  match the primary key columns of the user table and have corresponding data type and lengths definitions. The data type of each of the columns in the view exactly corresponds to the data type of the corresponding primary key column.

Each row in this view represents an insert, a delete, or an update operation on a user table row or text document. You can identify the text document by retrieving the primary key column values from the row in this view and looking them up in the user table.

You can use the following query to obtain information about the view:

```
db2 "SELECT STAGINGVIEWSHEMA, STAGINGVIEWNAME from SYSIBMTS.TSINDEXES
where INDSHEMA=schema-name and INDNAME=index-name"
```

Table 22. SYSIBMTS.TSSTAGING view

Column Name	Data type	Nullable?	Description
OPERATION	INTEGER	NO	The operation (insert, update, or delete) on the base table to be reflected on the text search index.

Table 22. SYSIBMTS.TSSTAGING view (continued)

Column Name	Data type	Nullable?	Description
TIME	TIMESTAMP	NO	Sequence ID of a row (when an insert, an update, or a delete trigger is fired). This is a time stamp but does not exactly represent the time of the operation.
STATUS	INTEGER	NO	Processing status of the row: -1 unprocessed
PK01	Data type of the first primary key column of the base table	YES	First primary key column of the base table.
...	...	...	...
PK $n$	Data type of the last primary key column of the base table	YES	Last primary key column of the base table.

## Appendix A. DB2 Text Search and Net Search Extender comparison

You should be aware of the differences in syntax, semantics, and results sets for full-text search queries that look similar in both solutions before migrating from Net Search Extender (NSE) to DB2 Text Search.

Review Table 23 and Table 24 on page 206 to help you to determine whether you can port from NSE to DB2 Text Search.

DB2 Text Search is supported on all operating systems that NSE is supported, except for Linux on System z<sup>®</sup> (64-bit) operating systems. The following table provides a list of install functions available in NSE and DB2 Text Search:

Table 23. Install functions available in NSE and DB2 Text Search

Function	NSE	DB2 Text Search	Comments and links to additional information
Local Install for Text Engine	Yes	Yes	
Remote Install for Text Engine	No	Yes	“DB2 Text Search server deployment scenarios” at <a href="http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/topic/com.ibm.db2.luw.admin.ts.doc/doc/c0058598.html">http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/topic/com.ibm.db2.luw.admin.ts.doc/doc/c0058598.html</a>
Database partitioning	Yes	Yes	“DB2 Text Search in a partitioned database environment” at <a href="http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/topic/com.ibm.db2.luw.admin.ts.doc/doc/c0058524.html">http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/topic/com.ibm.db2.luw.admin.ts.doc/doc/c0058524.html</a>
Index on non-partitioned base tables	Yes	Yes	Text search index creation, updates, and property alterations at <a href="http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/topic/com.ibm.db2.luw.admin.ts.doc/doc/c_textindexcreation.html">http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/topic/com.ibm.db2.luw.admin.ts.doc/doc/c_textindexcreation.html</a>
Index on partitioned base tables (Range-partitioned)	Yes	Yes	Extended text-maintained staging infrastructure for text search index incremental updates at <a href="http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/topic/com.ibm.db2.luw.admin.ts.doc/doc/c0057426.html">http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/topic/com.ibm.db2.luw.admin.ts.doc/doc/c0057426.html</a>

Table 23. Install functions available in NSE and DB2 Text Search (continued)

Function	NSE	DB2 Text Search	Comments and links to additional information
Index on Nicknames (with Replication)	Deprecated	No	Deprecated in Version 9.7
Index on Views	Yes	No	

DB2 Text Search provides similar functionality to NSE functionality. The following table shows the functionality available in NSE and DB2 Text Search:

Table 24. Functionality available in NSE and DB2 Text Search

Functional Items	NSE	DB2 Text Search	Comments and links to additional information
Recreate on update	Yes	Yes	
Custom transformation functions	Yes	Yes	
Caching	No	No	
Multiple Indexes	Yes	No	
Pre-sorted indexes	No	No	
Synonym dictionary	Yes	Yes	"Synonym dictionaries for DB2 Text Search" at <a href="http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/topic/com.ibm.db2.luw.admin.ts.doc/doc/c0052652.html">http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/topic/com.ibm.db2.luw.admin.ts.doc/doc/c0052652.html</a>
Thesaurus (associative, hierarchical, user-defined)	No	No	
Text, HTML, XML	Yes	Yes	"Document formats supported for DB2 Text Search" at <a href="http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/topic/com.ibm.db2.luw.admin.ts.doc/doc/r0053096.html">http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/topic/com.ibm.db2.luw.admin.ts.doc/doc/r0053096.html</a>
INSO	Yes	Yes	DB2 Text Search supports INSO using the DB2 accessories suite package. See "Rich text document support" at <a href="http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/topic/com.ibm.db2.luw.admin.ts.doc/doc/c0054766.html">http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/topic/com.ibm.db2.luw.admin.ts.doc/doc/c0054766.html</a> for details.
GPP	Yes	No	You can create a function in DB2 Text Search to support GPP
Document Models	Yes	No	

Table 24. Functionality available in NSE and DB2 Text Search (continued)

Functional Items	NSE	DB2 Text Search	Comments and links to additional information
Linguistic processing	Yes	Yes <sup>+</sup>	NSE linguistic process is limited to simple stemming (English only).  DB2 Text Search supports linguistic processing for 20 languages, including both morphological and n-gram segmentation support for Chinese, Japanese, and Korean. See "Linguistic processing for DB2 Text Search" for details.
CONTAINS function	Yes	Yes	"CONTAINS function at <a href="http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/topic/com.ibm.db2.luw.admin.ts.doc/doc/r_contains.html">http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/topic/com.ibm.db2.luw.admin.ts.doc/doc/r_contains.html</a> "
SCORE function	Yes	Yes	DB2 Text Search uses a different algorithm that might return different results. See "SCORE function at <a href="http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/topic/com.ibm.db2.luw.admin.ts.doc/doc/r_score.html">http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/topic/com.ibm.db2.luw.admin.ts.doc/doc/r_score.html</a> " for details.
Number of matches	No	No	
Highlights	No	No	
Stop-word processing	Yes	Yes	"Stop-word tool for DB2 Text Search syntax at <a href="http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/topic/com.ibm.db2.luw.admin.ts.doc/doc/r0058492.html">http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/topic/com.ibm.db2.luw.admin.ts.doc/doc/r0058492.html</a> "
Result limit	Yes	Yes	The CONTAINS and SCORE functions have a <b>RESULTLIMIT</b> parameter to indicate the maximum number of results to be returned.
Character normalization	Yes	Yes	
Escape characters	Yes	Yes	Customization is not available in DB2 Text Search.
Boolean search	Yes	Yes	"Text search argument syntax at <a href="http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/topic/com.ibm.db2.luw.admin.ts.doc/doc/r0052651.html">http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/topic/com.ibm.db2.luw.admin.ts.doc/doc/r0052651.html</a> "
Wildcard characters	Yes	Yes	"Text search argument syntax at <a href="http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/topic/com.ibm.db2.luw.admin.ts.doc/doc/r0052651.html">http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/topic/com.ibm.db2.luw.admin.ts.doc/doc/r0052651.html</a> "

Table 24. Functionality available in NSE and DB2 Text Search (continued)

Functional Items	NSE	DB2 Text Search	Comments and links to additional information
Stemmed search	Yes	Yes	Stemmed search is the default for DB2 Text Search
Precise search	Yes	Yes	DB2 Text Search is not case-sensitive. See “Precise search at <a href="http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/topic/com.ibm.db2.luw.admin.ts.doc/doc/t_searchingwiththetextindex.html">http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/topic/com.ibm.db2.luw.admin.ts.doc/doc/t_searchingwiththetextindex.html</a> ” for details.
Fuzzy search	Yes	Yes	“Fuzzy search at <a href="http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/topic/com.ibm.db2.luw.admin.ts.doc/doc/c0058557.html">http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/topic/com.ibm.db2.luw.admin.ts.doc/doc/c0058557.html</a> ”
Proximity search	Yes	Yes	“Proximity search at <a href="http://www.ibm.com/support/.ibm.com/infocenter/db2luw/v10r1/topic/com.ibm.db2.luw.admin.ts.doc/doc/c0058673.html">http://www.ibm.com/support/.ibm.com/infocenter/db2luw/v10r1/topic/com.ibm.db2.luw.admin.ts.doc/doc/c0058673.html</a> ”
Range search	Yes	Yes, for XML	DB2 Text Search relies on XPath expressions in XML for range search. Net Search Extender supports range search via the document model.
Freetext search	Yes	No	
Fielded search	Yes	Yes, for XML	DB2 Text Search support uses XPath expressions in XML. NSE support uses the document model. See “XML search configuration for DB2 Text Search at <a href="http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/topic/com.ibm.db2.luw.admin.ts.doc/doc/c0052709.html">http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/topic/com.ibm.db2.luw.admin.ts.doc/doc/c0052709.html</a> ” and “Searching XML documents using DB2 Text Search at <a href="http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/topic/com.ibm.db2.luw.admin.ts.doc/doc/c0052708.html">http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/topic/com.ibm.db2.luw.admin.ts.doc/doc/c0052708.html</a> ” for details.
Attribute search	Yes	No	



Table 24. Functionality available in NSE and DB2 Text Search (continued)

Functional Items	NSE	DB2 Text Search	Comments and links to additional information
Weights/boosting	Yes	Yes	DB2 Text Search and NSE have different algorithms. See "Searching text search indexes using SCORE at <a href="http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/topic/com.ibm.db2.luw.admin.ts.doc/doc/t_searchingandreturningscore.html">http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/topic/com.ibm.db2.luw.admin.ts.doc/doc/t_searchingandreturningscore.html</a> " for details.



---

## Appendix B. Locales supported for DB2 Text Search

The following table lists the locales that DB2 Text Search supports for document processing.

*Table 25. Supported locales*

Locale code	Language	Territory
ar_AA	Arabic	Arabic countries or regions
cs_CZ	Czech	Czech Republic
da_DK	Danish	Denmark
de_CH	German	Switzerland
de_DE	German	Germany
el_GR	Greek	Greece
en_AU	English	Australia
en_GB	English	United Kingdom
en_US	English	United States
es_ES	Spanish	Spain
fi_FI	Finnish	Finland
fr_CA	French	Canada
fr_FR	French	France
it_IT	Italian	Italy
ja_JP	Japanese	Japan
ko_KR	Korean	Korea, Republic of
nb_NO	Norwegian Bokmål	Norway
nl_NL	Dutch	Netherlands
nn_NO	Norwegian Nynorsk	Norway
pl_PL	Polish	Poland
pt_BR	Portuguese	Brazil
pt_PT	Portuguese	Portugal
ru_RU	Russian	Russia
sv_SE	Swedish	Sweden
zh_CN	Chinese	China
zh_TW	Chinese	Taiwan



---

## Appendix C. DB2 commands

---

### db2iupgrade - Upgrade instance

Upgrades an instance to a DB2 copy of the current release from a DB2 copy of a previous release. The DB2 copy from where you are running the **db2iupgrade** command must support instance upgrade from the DB2 copy that you want to upgrade.

On Linux and UNIX operating systems, this command is located in the *DB2DIR/instance* directory, where *DB2DIR* represents the installation location where the new release of the DB2 database system is installed. This command does not support instance upgrade for a non-root installation.

On Windows operating systems, this command is located in the **DB2PATH\bin** directory, where **DB2PATH** is the location where the DB2 copy is installed. To move your instance profile from its current location to another location, use the **/p** option and specify the instance profile path. Otherwise, the instance profile will stay in its original location after the upgrade.

#### Authorization

Root user authority on Linux and UNIX operating systems or Local Administrator on Windows operating systems.

#### Command syntax

##### For Linux and UNIX operating systems

```
db2iupgrade [-d] [-k]
             [-j "TEXT_SEARCH [,servicename] [,portnumber]" ]
             [-a AuthType] [-u FencedID] InstName
```

##### For Windows operating systems

```
db2iupgrade InstName /u: username,password
             [/p: instance-profile-path] [/q] [/a: authType]
             [/j "TEXT_SEARCH [,servicename] [,portnumber]" ] [/?]
```

#### Command parameters

##### For Linux and UNIX operating systems

- d Turns debug mode on. Use this option only when instructed by DB2 database support.
- k Keeps the pre-upgrade instance type if it is supported in the DB2 copy from where you are running the **db2iupgrade** command. If this parameter is not specified, the instance type is upgraded to the default instance type supported.

**-j "TEXT\_SEARCH"**

Configures the DB2 Text Search server using generated default values for service name and TCP/IP port number. This parameter cannot be used if the instance type is client.

**-j "TEXT\_SEARCH, servicename"**

Configures the DB2 Text Search server using the provided service name and an automatically generated port number. If the service name has a port number assigned in the services file, it uses the assigned port number.

**-j "TEXT\_SEARCH, servicename, portnumber"**

Configures the DB2 Text Search server using the provided service name and port number.

**-j "TEXT\_SEARCH, portnumber"**

Configures the DB2 Text Search server using a default service name and the provided port number. Valid port numbers must be within the 1024 - 65535 range.

**-a AuthType**

Specifies the authentication type (SERVER, CLIENT or SERVER\_ENCRYPT) for the instance. The default is SERVER.

**-u FencedID**

Specifies the name of the user ID under which fenced user-defined functions and fenced stored procedures will run. This option is required when upgrading a DB2 client instance to a DB2 server instance.

*InstName*

Specifies the name of the instance.

**For Windows operating systems**

*InstName*

Specifies the name of the instance.

**/u:username,password**

Specifies the account name and password for the DB2 service. This option is required when upgrading a partitioned instance.

**/p:instance-profile-path**

Specifies the new instance profile path for the upgraded instance.

**/q** Issues the **db2iupgrade** command in quiet mode.

**/a:authType**

Specifies the authentication type (SERVER, CLIENT, or SERVER\_ENCRYPT) for the instance.

**/j "TEXT\_SEARCH"**

Configures the DB2 Text Search server using generated default

values for service name and TCP/IP port number. This parameter cannot be used if the instance type is client.

`/j "TEXT_SEARCH, servicename"`

Configures the DB2 Text Search server using the provided service name and an automatically generated port number. If the service name has a port number assigned in the services file, it uses the assigned port number.

`/j "TEXT_SEARCH, servicename, portnumber"`

Configures the DB2 Text Search server using the provided service name and port number.

`/j "TEXT_SEARCH, portnumber"`

Configures the DB2 Text Search server using a default service name and the provided port number. Valid port numbers must be within the 1024 - 65535 range.

`/?` Displays usage information for the **db2iupgrade** command.

## Usage notes

Only DB2 Enterprise Server Edition instances (instance type *ese*) can be upgraded using the **db2iupgrade** command.

The **db2iupgrade** command calls the **db2ckupgrade** command with the **-not1** parameter, and specifies `update.log` as the log file for **db2ckupgrade**. The default log file created for **db2iupgrade** is `/tmp/db2ckupgrade.log.processID`. Verify that local databases are ready for upgrade before upgrading the instance. The **-not1** parameter disables the check for type-1 indexes. The log file is created in the instance home directory for Linux and UNIX operating systems or in the current directory for Windows operating systems. The instance upgrade will not continue if the **db2ckupgrade** command returns any errors.

For partitioned database environments, run the **db2ckupgrade** command before you issue the **db2iupgrade** command. The **db2ckupgrade** command will check all partitions and will return errors found in any partition. If you do not check whether all database partitions are ready for upgrade, subsequent database upgrades could fail even though the instance upgrade was successful. See **db2ckupgrade** for details.

### For Linux and UNIX operating systems

- If you use the **db2iupgrade** command to upgrade a DB2 instance from a previous version to the current version of a DB2 database system, the DB2 Global Profile Variables defined in an old DB2 database installation path will not be upgraded over to the new installation location. The DB2 Instance Profile Variables specific to the instance to be upgraded will be carried over after the instance is upgraded.
- If you are using the **su** command instead of the **login** command to become the root user, you must issue the **su** command with the **-** option to indicate that the process environment is to be set as if you had logged in to the system using the **login** command.
- You must not source the DB2 instance environment for the root user. Running the **db2iupgrade** command when you sourced the DB2 instance environment is not supported.

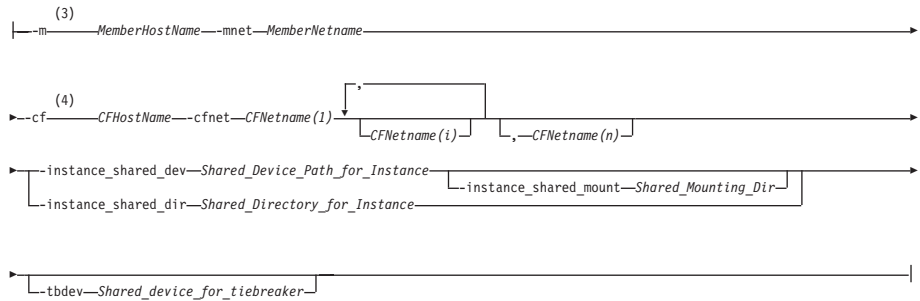




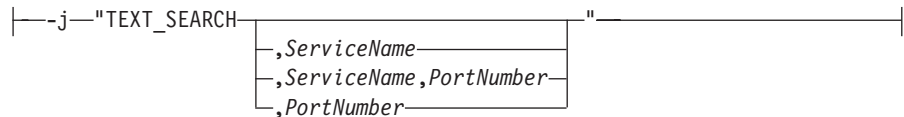
### InstType:



### DB2 pureScale options:



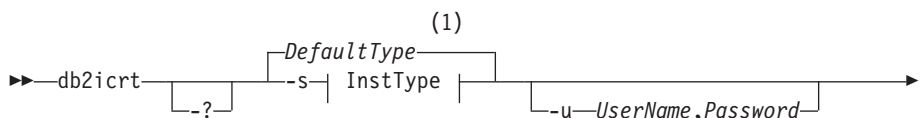
### DB2 Text Search options:

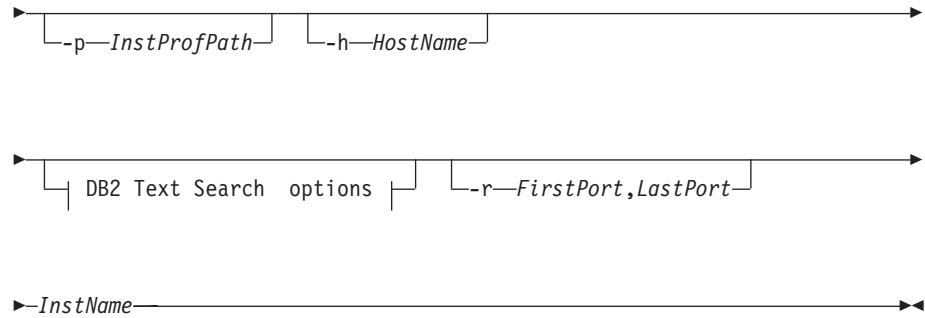


### Notes:

- 1 If the instance type is not specified with `-s`, the default instance type created is the highest type listed in *InstType* that DB2 copy is licensed to create.
- 2 When creating client instances, `-u FencedID` is not a valid option.
- 3 The `MemberHostName:MemberNetname` format has been deprecated for the `-m` option, and might be discontinued in the future. The new format, with both `-m` and `-mnet` options, is required for IPv6 support with DB2 pureScale Feature.
- 4 The `CFHostName:CFNetames` format has been deprecated for the `-cf` option, and might be discontinued in the future. The new format, with both `-cf` and `-cfnet` options, is required for IPv6 support with DB2 pureScale Feature.

### For Windows operating systems

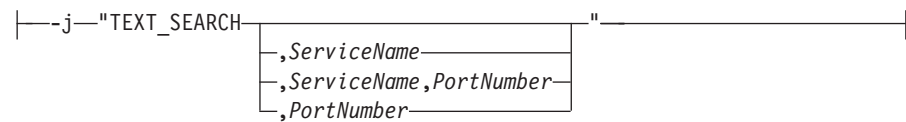




### InstType:



### DB2 Text Search options:



### Notes:

- 1 If the instance type is not specified with `-s`, the default instance type created is the highest type listed in *InstType* that DB2 copy is licensed to create.

## Command parameters

### For Linux and UNIX operating systems

- `-?` Displays the usage information.
- `-h` Displays the usage information.
- `-d` Turns on debug mode. Saves the trace file with default name in `/tmp` as `db2icrt.trc.ProcessID`. Use this option only when instructed by DB2 database support
- `-a AuthType`  
Specifies the authentication type (SERVER, CLIENT, or SERVER\_ENCRYPT) for the instance. The default is SERVER.
- `-j "TEXT_SEARCH"`  
Configures the DB2 Text Search server with generated default values for service name and TCP/IP port number. This parameter cannot be used if the instance type is `client`.
- `-j "TEXT_SEARCH, servicename"`  
Configures the DB2 Text Search server with the provided service name and an automatically generated port number.

If the service name has a port number assigned in the services file, it uses the assigned port number.

**-j "TEXT\_SEARCH, servicename, portnumber"**

Configures the DB2 Text Search server with the provided service name and port number.

**-j "TEXT\_SEARCH, portnumber"**

Configures the DB2 Text Search server with a default service name and the provided port number. Valid port numbers must be within the 1024 - 65535 range.

**-p <TCP/IP PortName>**

Specifies the TCP/IP port name or number used by the instance. This option also configures the database manager configuration parameter **SVCENAME** for the DB2 instance.

**-m MemberHostName:NetName1**

Specifies the host to set up as a DB2 member during instance creation. This parameter is mandatory in a DB2 pureScale environment. Only one DB2 member can be set up by the **db2icrt** command. Additional DB2 members can be added with the **db2iupdt -add** command. The *NetName1* syntax is deprecated and might be discontinued in a future release. Use the **-mnet** parameter instead.

The *MemberHostName* should be the canonical host name (for example, the output of 'hostname' command run on a local host). The *NetName1* value specified here must belong to the same subnet as specified in the **-cf** parameter.

**-mnet MemberNetName**

This parameter replaces the deprecated *:NetName1* syntax of the **-m MemberHostName:NetName1** parameter. Specifies the cluster interconnect netname, which is the hostname of the interconnect used for high speed communication between members and cluster caching facilities (also referred to as CF) in a DB2 pureScale instance.

**-cf CFHostName:NetName2**

Specifies the host to set up as a cluster caching facility (also referred to as CF) during instance creation. This parameter is mandatory in a DB2 pureScale environment. Only one CF can be set up by the **db2icrt** command. Additional CFs can be added by using the **db2iupdt -add** command. The *NetName2* syntax is deprecated and might be discontinued in a future release. Use the **-cfnet** parameter instead.

**-cfnet CFNetName**

This parameter replaces the deprecated *:NetName2* syntax of the **-cf CFHostName:NetName2** parameter. Specifies the cluster interconnect netname, which is the hostname of the interconnect used for high speed communication between members and CFs in a DB2 pureScale instance.

The *CFNetName* must belong to the same subnet as specified in the **-m** parameter, and must correspond to a cluster interconnect netname (for example, *db2\_<hostname\_ib0>*).

**-instance\_shared\_dev** *Shared\_Device\_Path\_for\_Instance*

Specifies a shared disk device path required to set up a DB2 pureScale instance to hold instance shared files and default database path. For example, /dev/hdisk1. The shared directory must be accessible on all the hosts for the DB2 pureScale instance. The value of this option cannot have the same value as the **-tbdev** option.

When the **-instance\_shared\_dev** parameter is specified, the DB2 installer creates a DB2 cluster file system.

The **-instance\_shared\_dev** parameter and the **-instance\_shared\_dir** parameter are mutually exclusive.

**-instance\_shared\_mount** *Shared\_Mounting\_Dir*

Specifies the mount point for a new IBM General Parallel File System (GPFS™) file system. The specified path must be a new and empty path that is not nested inside an existing GPFS file system.

**-instance\_shared\_dir** *Shared\_Directory\_for\_Instance*

Specifies a directory in a shared file system (GPFS) required to set up a DB2 pureScale instance to hold instance shared files and default database path. For example, /sharedfs. The disk must be accessible on all the hosts for the DB2 pureScale instance. The value of this option cannot have the same value as the **-tbdev** option or the installation path.

When the **-instance\_shared\_dir** parameter is specified, the DB2 installer uses a user-managed file system. The user-managed file system must be available on all hosts, and must be a GPFS file system.

The **-instance\_shared\_dir** parameter and the **-instance\_shared\_dev** parameter are mutually exclusive.

**-tbdev** *Shared\_device\_for\_tiebreaker*

Specifies a shared device path for a device that will act as a tiebreaker in the DB2 pureScale environment to ensure that the integrity of the data is maintained. The value of this option cannot have the same value as either the **-instance\_shared\_dev** option or the **-instance\_shared\_dir** option. This option is required when the DB2 cluster services tiebreaker is created for the first time. The disk device should not have any file system associated with it. This option is invalid if a DB2 cluster services Peer Domain already exists.

**-s** *InstType*

Specifies the type of instance to create. Use the **-s** option only when you are creating an instance other than the default associated with the installed product from which you are running **db2icrt**. Valid values are:

**dsf** Used to create a DB2 pureScale instance for a DB2 database server with local and remote clients. This option is the default instance type for the IBM DB2 pureScale Feature.

**ese** Used to create an instance for a database server with local and remote clients. This option is the default instance type for DB2 Enterprise Server Edition or DB2 Advanced Enterprise Server Edition.

**wse** Used to create an instance for a database server with local and remote clients. This option is the default instance type for DB2 Workgroup Server Edition, DB2 Express Edition or DB2 Express-C, and DB2 Connect™ Enterprise Edition.

**standalone**

Used to create an instance for a database server with local clients. It is the default instance type for DB2 Personal Edition.

**client** Used to create an instance for a client. This option is the default instance type for IBM Data Server Client, IBM Data Server Runtime Client, and DB2 Connect Personal Edition.

DB2 database products support their default instance types and the instance types lower than their default ones. For instance, DB2 Enterprise Server Edition supports the instance types of **ese**, **wse**, **standalone**, and **client**.

**-u Fenced ID**

Specifies the name of the user ID under which fenced user-defined functions and fenced stored procedures will run. The **-u** option is required if you are not creating a client instance.

*InstName*

Specifies the name of the instance which is also the name of an existing user in the operating system. The instance name must be the last argument of the **db2icrt** command.

**For Windows operating systems**

*InstName*

Specifies the name of the instance.

**-s InstType**

Specifies the type of instance to create. Currently, there are four kinds of DB2 instance types. Valid values are:

**client** Used to create an instance for a client. This option is the default instance type for IBM Data Server Client, IBM Data Server Runtime Client, and DB2 Connect Personal Edition.

**standalone**

Used to create an instance for a database server with local clients. It is the default instance type for DB2 Personal Edition.

**ese** Used to create an instance for a database server with local and remote clients with partitioned database environment support. The

*-s ese -u Username, Password*

options have to be used with **db2icrt** to create the ESE instance type and a partitioned database environment instance.

**wse** Used to create an instance for a database server with local and remote clients. This option is the default instance type for DB2 Workgroup Server Edition, DB2 Express Edition or DB2 Express-C, and DB2 Connect Enterprise Edition.

DB2 database products support their default instance types and the instance types lower than their default ones. For instance, DB2 Enterprise Server Edition supports the instance types of `ese`, `wse`, `standalone`, and `client`.

**-u** *Username, Password*

Specifies the account name and password for the DB2 service. This option is required when creating a partitioned database instance.

**-p** *InstProfPath*

Specifies the instance profile path.

**-h** *HostName*

Overrides the default TCP/IP host name if there is more than one for the current machine. The TCP/IP host name is used when creating the default database partition (database partition 0). This option is only valid for partitioned database instances.

**-r** *PortRange*

Specifies a range of TCP/IP ports to be used by the partitioned database instance when running in MPP mode. For example, `-r 50000,50007`. The services file of the local machine will be updated with the following entries if this option is specified:

```
DB2_InstName      baseport/tcp
DB2_InstName_END  endpoint/tcp
```

**/j** *"TEXT\_SEARCH"*

Configures the DB2 Text Search server with generated default values for service name and TCP/IP port number. This parameter cannot be used if the instance type is `client`.

**/j** *"TEXT\_SEARCH, servicename"*

Configures the DB2 Text Search server with the provided service name and an automatically generated port number. If the service name has a port number assigned in the services file, it uses the assigned port number.

**/j** *"TEXT\_SEARCH, servicename, portnumber"*

Configures the DB2 Text Search server with the provided service name and port number.

**/j** *"TEXT\_SEARCH, portnumber"*

Configures the DB2 Text Search server with a default service name and the provided port number. Valid port numbers must be within the 1024 - 65535 range.

**-?** Displays usage information.

## Examples

1. To create a DB2 pureScale instance for the instance owner `db2sdin1` and fenced user `db2sdfel`, run the following command:

```
DB2DIR/instance/db2icrt
-cf host1.domain.com -cfnet host1.domain.com-ib0
-m host2.domain.com -mnet host2.domain.com-ib0
-instance_shared_dev /dev/hdisk1
-tbdev /dev/hdisk2
-u db2sdfel
db2sdin1
```

where *DB2DIR* represents the installation location of your DB2 copy. The DB2 pureScale instance *db2sdin1* will have a CF on *host1*, and a member on *host2*. This command also uses */dev/hdisk1* to create a shared file system to store instance shared files and sets up */dev/hdisk2* as the shared device path for the tiebreaker.

2. To create a DB2 Enterprise Server Edition instance for the user ID *db2inst1*, run the following command:

```
DB2DIR/instance/db2icrt -s ese -u db2fenc1 db2inst1
```

where *DB2DIR* represents the installation location of your DB2 copy.

3. To create a DB2 pureScale instance that uses an existing file system (GPFS) managed by the DB2 product for the instance owner *db2sdin1* and the fenced user *db2sdfel*, run the following command:

```
DB2DIR/instance/db2icrt
-cf host1.domain.com -cfnet host1.domain.com-ib0
-m host2.domain.com -mnet host2.domain.com-ib0
-tbdev /dev/hdisk2
-u db2sdfel
db2sdin1
```

where *DB2DIR* represents the installation location of your DB2 copy.

4. To create a DB2 pureScale instance with an existing user-managed GPFS file system (*/gpfs\_shared\_dir*) for the instance owner *db2sdin1* and the fenced user *db2sdfel*, run the following command:

```
DB2DIR/instance/db2icrt
-cf host1.domain.com -cfnet host1.domain.com-ib0
-m host2.domain.com -mnet host2.domain.com-ib0
-instance_shared_dir /gpfs_shared_dir
-tbdev /dev/hdisk2
-u db2sdfel
db2sdin1
```

where *DB2DIR* represents the installation location of your DB2 copy.

5. On an AIX machine, to create an instance for the user ID *db2inst1*, issue the following command:

On a client machine:

```
DB2DIR/instance/db2icrt db2inst1
```

On a server machine:

```
DB2DIR/instance/db2icrt -u db2fenc1 db2inst1
```

where *db2fenc1* is the user ID under which fenced user-defined functions and fenced stored procedures will run.

## Usage notes

- The instance user must exist on all hosts with the same UID, GID, group name, and home directory path. The same rule applies for the fenced user. After the **db2icrt** command is successfully run, the DB2 installer will set up SSH for the instance user across hosts.
- When using the **db2icrt** command, the name of the instance must match the name of an existing user.
- You can have only one instance per DB2 pureScale environment.
- When creating DB2 instances, consider the following restrictions:
  - If existing IDs are used to create DB2 instances, make sure that the IDs are not locked and do not have passwords expired.

- Group ID (GID) must be within range 100-65535.
- You can also use the **db2isetup** command to create and update DB2 instances and add multiple hosts with a graphical interface.
- If you are using the **su** command instead of the **login** command to become the root user, you must issue the **su** command with the **-** option to indicate that the process environment is to be set as if you had logged in to the system with the **login** command.
- You must not source the DB2 instance environment for the root user. Running **db2icrt** when you sourced the DB2 instance environment is not supported.
- If you have previously created a DB2 pureScale instance and have dropped it, you cannot re-create it using the **-instance\_shared\_dev** parameter specification since the DB2 cluster file system might already have been created. To specify the previously created shared file system:
  - If the existing GPFS shared file system was created and managed by DB2 pureScale Feature, the **-instance\_shared\_dev** parameter and the **-instance\_shared\_dir** parameter should not be used.
  - If the existing GPFS shared file system was not created and managed by DB2 pureScale Feature, use the **-instance\_shared\_dir** parameter.
- On AIX 6.1 (or higher), when running this command from a shared DB2 copy in a system workload partition (WPAR) global environment, this command must be run as the root user. WPAR is not supported in a DB2 pureScale environment.

---

## db2idrop - Remove instance

Removes a DB2 instance that was created by **db2icrt**.

You can only drop instances that are listed by the **db2ilist** command for the same DB2 copy where you are issuing the **db2idrop** command from. You can also use the **db2idrop** command to drop a DB2 pureScale instance.

On Linux and UNIX operating systems, this utility is located in the *DB2DIR/instance* directory, where *DB2DIR* represents the installation location where the current version of the DB2 database system is installed. On Windows operating systems, this utility is located under the **DB2PATH\bin** directory where **DB2PATH** is the location where the DB2 copy is installed.

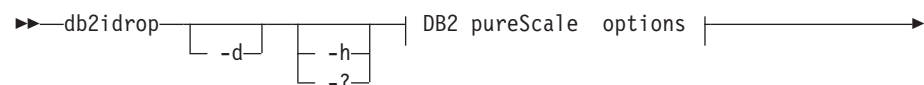
**Note:** A non-root-installed DB2 instance, on Linux and UNIX operating systems, cannot be dropped using this command. The only option is to uninstall the non-root DB2 copy. See the following *Usage notes* section for more details.

### Authorization

Root user authority on Linux and UNIX operating systems or Local Administrator on Windows operating systems.

### Command syntax

For Linux and UNIX operating systems





► | Outside Of DB2 pureScale options | *InstName* →

**DB2 pureScale options:**

| [ -g ] |

**Outside Of DB2 pureScale options:**

| [ -f ] |

**For Windows operating systems**

►► db2idrop [ -f ] [ -h ] *InstName* →

**Command parameters**

**For Linux and UNIX operating systems**

- d** Enters debug mode, for use by DB2 database support.
  - f** This parameter is deprecated.  
Specifies the force applications flag. If this flag is specified all the applications using the instance will be forced to terminate. This parameter is not supported on a DB2 pureScale environment.
  - g** This parameter is required when **db2idrop** is used with a DB2 pureScale instance. Specifies that you want to drop the DB2 pureScale instance on all hosts. This parameter requires all DB2 members and all cluster caching facilities are stopped on all the hosts in the DB2 pureScale instance. This option will be ignored for dropping any other instance type
  - h | -?** Displays the usage information.
- InstName*  
Specifies the name of the instance.

**For Windows operating systems**

- InstName*  
Specifies the name of the instance.
- f** Specifies the force applications flag. If this flag is specified all the applications using the instance will be forced to terminate.
  - h** Displays usage information.

**Example**

If you created db2inst1 on a Linux or UNIX operating system by issuing the following command:

```
/opt/IBM/db2/copy1/instance/db2icrt -u db2fenc1 db2inst1
```

To drop `db2inst1`, you must run the following command:

```
/opt/IBM/db2/copy1/instance/db2idrop db2inst1
```

## Usage notes

- Before an instance is dropped, ensure that the DB2 database manager has been stopped on all hosts and that DB2 database applications accessing the instance are disconnected and terminated. DB2 databases associated with the instance can be backed up, and configuration data saved for future reference if needed.
- The **db2idrop** command does not remove any databases. Remove the databases first if they are no longer required. If the databases are not removed, they can always be catalogued under another DB2 copy of the same release and continued to be used.
- If you want to save DB2 Text Search configurations and plan to reuse instance databases, you need to take the extra step of saving the config directory (on UNIX: `instance_home/sql1lib/db2tss/config` and on Windows: `instance_profile_path\instance_name\db2tss\config`) or config directory contents before issuing the **db2idrop** command. After the new instance is created, the config directory can be restored. However, restoring the config directory is only applicable if the new instance created is of the same release and fix pack level.
- A non-root-installed instance cannot be dropped on Linux and UNIX operating systems. To remove this DB2 instance, the only option available to the user is to uninstall the non-root copy of DB2 by running **db2\_deinstall -a**.
- On Linux and UNIX operating systems, if you are using the **su** command instead of the **login** command to become the root user, you must issue the **su** command with the **-** option to indicate that the process environment is to be set as if you had logged in to the system using the **login** command.
- On Linux and UNIX operating systems, you must not source the DB2 instance environment for the root user. Running **db2idrop** when you sourced the DB2 instance environment is not supported.
- In a DB2 pureScale environment, the **-g** parameter is mandatory. In this case, the instance is dropped on all hosts. However, the IBM General Parallel File System (GPFS) on the installation-initiating host (IIH) is not deleted, nor is the GPFS file system. You must manually remove the file system and uninstall GPFS.
- On Windows operating systems, if an instance is clustered with Microsoft Cluster Service (MSCS), then you can uncluster that instance by issuing the **db2mcs** or **db2iclus** command before dropping the instance.
- On AIX 6.1 (or higher), when running this command from a shared DB2 copy in a system workload partition (WPAR) global environment, this command must be run as the root user. WPAR is not supported in a DB2 pureScale environment.

---

## db2iupdt - Update instances

The **db2iupdt** command can be used to update an instance to a higher level within a release, update an instance other than a DB2 pureScale instance to a DB2 pureScale instance, or scale a DB2 pureScale instance.

Before running the **db2iupdt** command, you must first stop the instance and all processes that are running for the instance. When using this command to update an instance other than a DB2 pureScale instance, add a member or a cluster caching facility, ensure that you have reviewed the prerequisites and the pre-installation checklist so that your instance and hosts are compliant.

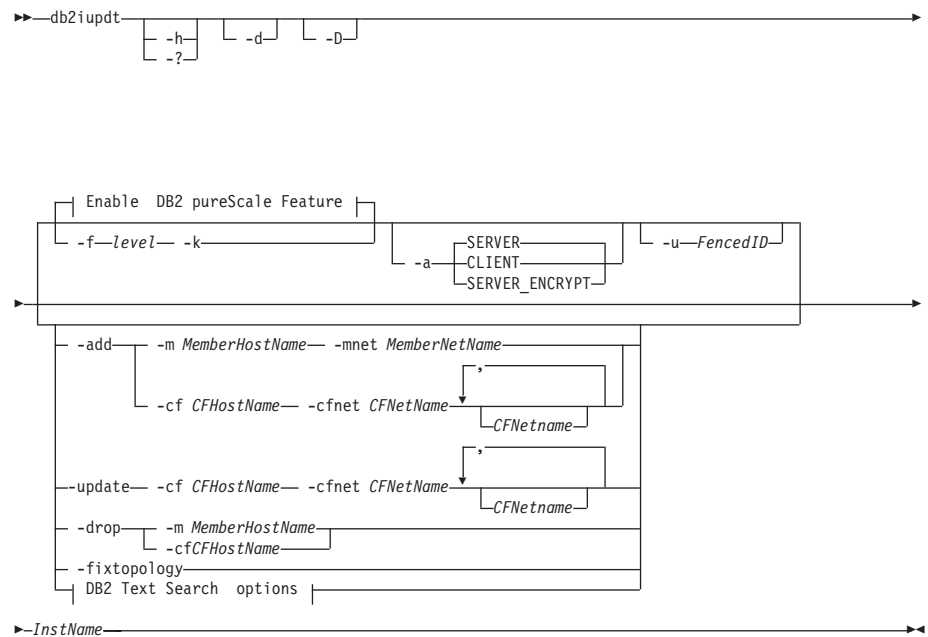
**Note:** In a DB2 pureScale instance, you cannot make changes to the resource model without having a *configurational quorum*, meaning that there is an absolute majority nodes online. In a two-host setup, you cannot use the **db2iupdt** command if one of the hosts is offline.

## Authorization

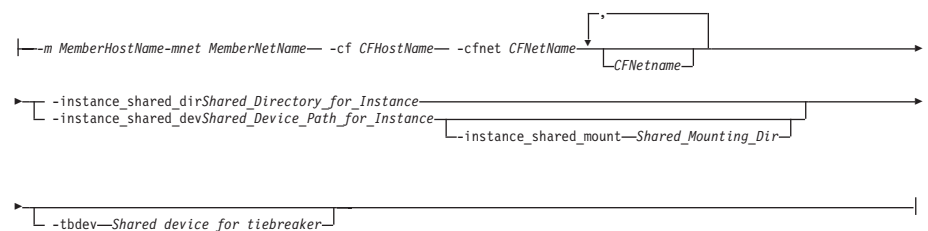
Root user authority on UNIX and Linux operating systems or Local Administrator on Windows operating systems.

## Command syntax

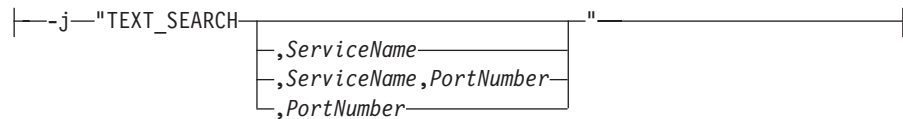
For UNIX and Linux operating systems



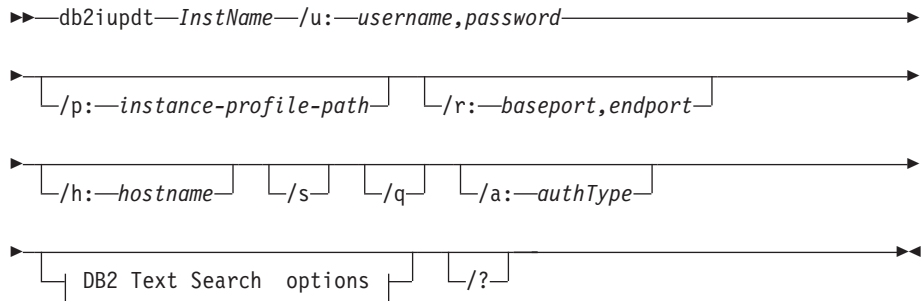
### Enable DB2 pureScale Feature:



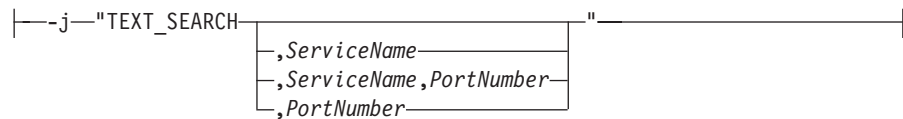
## DB2 Text Search options:



## For Windows operating systems



## DB2 Text Search options:



## Command parameters

### For UNIX and Linux operating systems

**-h | -?**

Displays the usage information.

**-a AuthType**

Specifies the authentication type (SERVER, SERVER\_ENCRYPT or CLIENT) for the instance. The default is SERVER.

**-d** Turns on debug mode.

**-k** Keeps the current instance type during the update.

**-D** This parameter has been deprecated and might be removed in a future release. This parameter is replaced by the **-f level** parameter. Moves an instance from a higher code level on one path to a lower code level installed on another path.

**-f level**

Updates an instance from a higher DB2 version instance type to a lower DB2 version instance type for compatibility.

**-add -m MemberHostName:Netname1 | -cf CFHostName:Netname2**

Specifies the host (member or cluster caching facility) to be added to a DB2 pureScale instance.

To specify which type of host is to be added, use **-m** for member, or **-cf** for cluster caching facility. You must select one, not both.

The **db2iupdt -add** command must be run from a host that is already part of the DB2 pureScale instance.

The `-m MemberHostName:NetName1` and `-cf CFHostName:NetName2` syntax is deprecated and might be discontinued in a future release. Use the syntax `-add -m MemberHostName -mnet MemberNetName` or `-add -cf CFHostName -cfnet CFNetName` syntax instead.

**-add -m MemberHostName -mnet MemberNetName | -add -cf CFHostName -cfnet CFNetName**

Specifies the cluster interconnect netname, which is the hostname of the interconnect used for high speed communication between DB2 members and cluster caching facility. The Netname of each host you want to add has to be on the same subnet as any existing members or CFs in the instance.

To specify which type of host is to be added, use `-m MemberHostName -mnet MemberNetName` for a member, or `-cf CFHostName -cfnet CFNetName` for a cluster caching facility. You must select one, not both.

The `db2iupdt -add` command must be run from a host that is already part of the DB2 pureScale instance.

**-add** Specifies the hostname and cluster interconnect netname or netnames of the host to be added to the DB2 pureScale Feature instance. The `db2iupdt -add` command must be run from a host that is already part of the DB2 pureScale instance.

**-m MemberHostName -mnet MemberNetName**

The host with hostname *MemberHostName* is added to the DB2 pureScale Feature instance with the cluster interconnect netname *MemberNetName*.

**-cf CFHostName -cfnet CFNetName**

The host with hostname *CFHostName* is added to the DB2 pureScale Feature instance as a cluster caching facility with the cluster interconnect netname *CFNetName*. If *CFHostName* has multiple cluster interconnect network adapter ports, you can supply a comma delimited list for *CFNetName* to separate each cluster interconnect netname.

**-drop -m MemberHostName | -cf CFHostName**

Specifies the host (member or cluster caching facility) to be dropped from a DB2 pureScale instance. To specify which type of host to be dropped, use the `-m` option for a member, or `-cf` option for a cluster caching facility. You must select one, not both. This parameter cannot be used to drop the last member and the last CF from a DB2 pureScale instance. This parameter should not be used with the `-add` parameter.

**-instance\_shared\_dev Shared\_Device\_Path\_for\_Instance**

Specifies a shared disk device path required to set up a DB2 pureScale instance to hold instance shared files and default database path. For example, the device path `/dev/hdisk1`. The shared directory must be accessible on all the hosts for the DB2 pureScale instance. The value of this parameter cannot have the same value as the `-tbdev` parameter. This parameter and `-instance_shared_dir` are mutually exclusive.

This parameter is only required if you are updating an instance other than a DB2 pureScale instance to a DB2 pureScale instance.

**-instance\_shared\_mount** *Shared\_Mounting\_Dir*  
Specifies the mount point for a new IBM General Parallel File System ( GPFS) file system. The specified path must be a new and empty path that is not nested inside an existing GPFS file system.

**-instance\_shared\_dir** *Shared\_Directory\_for\_Instance*  
Specifies the directory in a shared file system (GPFS) required to set up a DB2 pureScale instance to hold instance shared files and default database path. For example, /sharedfs. The disk must be accessible on all the hosts for the DB2 pureScale instance. The value of this parameter cannot have the same value as the **-tbdev** parameter. This parameter and **-instance\_shared\_dev** are mutually exclusive.

This parameter is only required if you are updating an instance other than a DB2 pureScale instance to a DB2 pureScale instance.

**-tbdev** *Shared\_device\_for\_tiebreaker*  
Specifies a shared device path that will act as a tiebreaker in the DB2 pureScale environment to ensure that the integrity of the data is maintained. The value of this parameter cannot have the same value as either the **-instance\_shared\_dev** parameter or the **-instance\_shared\_dir** parameter. This parameter is required when the DB2 cluster services tiebreaker is created for the first time, or if updating an instance other than a DB2 pureScale instance to a DB2 pureScale instance. This parameter is invalid if a DB2 cluster services Peer Domain exists.

**-j "TEXT\_SEARCH"**  
Configures the DB2 Text Search server with generated default values for service name and TCP/IP port number. This parameter cannot be used if the instance type is client or dsf.

**-j "TEXT\_SEARCH, servicename"**

Configures the DB2 Text Search server with the provided service name and an automatically generated port number. If the service name has a port number assigned in the services file, it uses the assigned port number.

**-j "TEXT\_SEARCH, servicename, portnumber"**

Configures the DB2 Text Search server with the provided service name and port number.

**-j "TEXT\_SEARCH, portnumber"**

Configures the DB2 Text Search server with a default service name and the provided port number. Valid port numbers must be within the 1024 - 65535 range.

**-u** *Fenced ID*  
Specifies the name of the user ID under which fenced user-defined functions and fenced stored procedures will run. This parameter is only needed when converting an instance from a client instance to a non-client instance type. To determine the current instance type, refer to the node type parameter in the output from a **GET DBM CFG** command. If an instance is already a non-client instance, or if an instance is a client instance and is staying as a client instance (for

example, by using the **-k** parameter), the **-u** parameter is not needed. The **-u** parameter can change the fenced user for an existing instance.

**-fixtopology**

Used to manually correct a failed add or drop operation. For an add operation, this parameter will roll back any changes to return to the previous topology. For a drop operation, this parameter will complete the drop operation. This parameter cannot be used in combination with any other parameters, except **-d**.

*InstName*

Specifies the name of the instance.

**For Windows operating systems**

*InstName*

Specifies the name of the instance.

*/u:username,password*

Specifies the account name and password for the DB2 service.

*/p:instance-profile-path*

Specifies the new instance profile path for the updated instance.

*/r:baseport,endport*

Specifies the range of TCP/IP ports to be used by the partitioned database instance when running in MPP mode. When this option is specified, the services file on the local machine will be updated with the following entries:

```
DB2_InstName      baseport/tcp
DB2_InstName_END  endport/tcp
```

*/h:hostname*

Overrides the default TCP/IP host name if there are more than one TCP/IP host names for the current machine.

*/s* Updates the instance to a partitioned instance.

*/q* Issues the **db2iupdt** command in quiet mode.

*/a:authType*

Specifies *authType*, the authentication type (SERVER, CLIENT, or SERVER\_ENCRYPT) for the instance.

*/j "TEXT\_SEARCH"*

Configures the DB2 Text Search server with generated default values for service name and TCP/IP port number. This parameter cannot be used if the instance type is client.

*/j "TEXT\_SEARCH, servicename"*

Configures the DB2 Text Search server with the provided service name and an automatically generated port number. If the service name has a port number assigned in the services file, it uses the assigned port number.

*/j "TEXT\_SEARCH, servicename, portnumber"*

Configures the DB2 Text Search server with the provided service name and port number.

*/j "TEXT\_SEARCH, portnumber"*

Configures the DB2 Text Search server with a default service name and the provided port number. Valid port numbers must be within the 1024 - 65535 range.

`/?` Displays usage information for the **db2iupdt** command.

## Example

### For UNIX and Linux operating systems

A `db2inst2` instance is associated to a DB2 copy of DB2 database product installed at `DB2DIR1`. You have another copy of a DB2 database product on the same computer at `DB2DIR2` for the same version of the DB2 database product that is installed on `DB2DIR1`. To update the instance to run from the DB2 copy installed at `DB2DIR1` to the DB2 copy installed at `DB2DIR2`, issue the following command:

```
DB2DIR2/instance/db2iupdt db2inst2
```

If the DB2 copy installed at `DB2DIR2` is at level lower than the DB2 copy installed at `DB2DIR1`, issue:

```
DB2DIR2/instance/db2iupdt -D db2inst2
```

### Update an instance to a higher level within a release

To update a DB2 instance to a higher level or from one DB2 installation path to another, enter a command such as the following:

```
DB2DIR/instance/db2iupdt db2inst1
```

where `DB2DIR` represents the installation location of your DB2 copy. If this command is run from a DB2 pureScale Feature copy, the existing `db2inst1` must have an instance type of `dsf`. If the `db2inst1` instance is a DB2 pureScale instance, this example can update it from one level to a different level of DB2 ESE with the DB2 pureScale Feature. This example does not apply to updating an `ese` type instance to a DB2 pureScale instance. The next example outlines this procedure.

### Update for an instance other than a DB2 pureScale instance to a DB2 pureScale instance

To update an instance to a DB2 pureScale instance:

```
DB2DIR/instance/db2iupdt
-cf host2
-cfnet host2-ib0
-m host1
-mnet host1-ib0
-instance_shared_dev /dev/hdisk1
-tbdev /dev/hdisk2
-u db2fenc1
db2inst1
```

where `DB2DIR` represents the installation location of your DB2 copy.

This command also uses `/dev/hdisk1` to create a shared file system to store instance shared files and sets up `/dev/hdisk2` as the shared device path that will act as a tiebreaker. The value of the `-tbdev` parameter must be different from the value of the `-instance_shared_dev` parameter.

### Scale a DB2 pureScale instance (by using `db2iupdt -add` or `db2iupdt -drop`)

The following examples apply to a DB2 pureScale environment:



- Update a DB2 pureScale instance to add a member. To add a member called `host1` with a netname of `host1-ib0` to the DB2 pureScale instance `db2sdin1` enter a command such as the following:

```
DB2DIR/instance/db2iupdt -d -add -m host1 -mnet host1-ib0 db2sdin1
```

where *DB2DIR* represents the installation location of your DB2 copy.

- Update a DB2 pureScale instance to add a second cluster caching facility. To add a cluster caching facility called `host2` with a netname of `host2-ib0` to the DB2 pureScale instance `db2sdin1` enter a command such as the following:

```
DB2DIR/instance/db2iupdt -d -add -cf host2 -cfnet host2-ib0 db2sdin1
```

where *DB2DIR* represents the installation location of your DB2 copy.

- Drop a member from a DB2 pureScale instance. To drop a member called `host1` from the DB2 pureScale instance `db2sdin1` enter a command such as the following:

```
DB2DIR/instance/db2iupdt -d -drop -m host1 db2sdin1
```

where *DB2DIR* represents the installation location of your DB2 copy. If `host1` does not have a CF role in the same instance, the command must be run from a host other than `host1`.

## Usage notes

### For all supported operating systems

- If you use the **db2iupdt** command to update a DB2 instance from one DB2 copy to another DB2 copy of the same version of DB2 database product, the DB2 Global Profile Variables defined in an old DB2 copy installation path will not be updated over to the new installation location. The DB2 Instance Profile Variables specific to the instance will be carried over after the instance is updated.

### For UNIX and Linux operating systems

- Only DB2DB2 Enterprise Server Edition can be updated by using the **db2iupt** command.
- If you change the member topology, for example by adding or dropping a member, you must take an offline backup before you can access the database. If you attempt to access the database before taking an offline backup, the database is placed in a backup pending state.  
You can add multiple members or drop multiple members without having to take a backup after each change. For example, if you add three members, you have to take a backup only after you completed all of the add operations. However, if you add two members and then drop a member, or if you drop two members and then add a member, you must take a backup before you can perform any additional member topology changes.
- The **db2iupdt** command is located in the *DB2DIR*/instance directory, where *DB2DIR* is the location where the current version of the DB2 database product is installed.
- If you want to update a non-root instance, refer to the **db2nrupdt** non-root-installed instance update command. The **db2iupdt** does not support updating of non-root instances.
- If you are using the **su** command instead of the **login** command to become the root user, you must issue the **su** command with the **-** option

to indicate that the process environment is to be set as if you had logged in to the system with the **login** command.

- You must not source the DB2 instance environment for the root user. Running **db2iupdt** when you sourced the DB2 instance environment is not supported.
- On AIX 6.1 (or higher), when running this command from a shared DB2 copy in a system workload partition (WPAR) global environment, this command must be run as the root user. WPAR is not supported in a DB2 pureScale environment.
- When **db2iupdt** is run to update an instance to a higher level within a release, routines and libraries are copied from each member to a shared location. If there is a library with the same name but different content on each host, the library content in the shared location is that of the last host that ran the **db2iupdt** command.

#### For Windows operating systems

- The **db2iupdt** command is located in the **DB2PATH\bin** directory, where **DB2PATH** is the location where the current version of the DB2 database product is installed.
- The instance is updated to the DB2 copy from which you issue the **db2iupdt** command. However, to move your instance profile from its current location to another location, use the **/p** option and specify the instance profile path. Otherwise, the instance profile will stay in its original location after the instance update. Use the **db2iupgrade** command instead to upgrade to the current release from a previous release.

---

## Appendix D. Overview of the DB2 technical information

DB2 technical information is available in multiple formats that can be accessed in multiple ways.

DB2 technical information is available through the following tools and methods:

- DB2 Information Center
  - Topics (Task, concept and reference topics)
  - Sample programs
  - Tutorials
- DB2 books
  - PDF files (downloadable)
  - PDF files (from the DB2 PDF DVD)
  - printed books
- Command-line help
  - Command help
  - Message help

**Note:** The DB2 Information Center topics are updated more frequently than either the PDF or the hardcopy books. To get the most current information, install the documentation updates as they become available, or refer to the DB2 Information Center at [ibm.com](http://ibm.com).

You can access additional DB2 technical information such as technotes, white papers, and IBM Redbooks® publications online at [ibm.com](http://www.ibm.com). Access the DB2 Information Management software library site at <http://www.ibm.com/software/data/sw-library/>.

### Documentation feedback

We value your feedback on the DB2 documentation. If you have suggestions for how to improve the DB2 documentation, send an email to [db2docs@ca.ibm.com](mailto:db2docs@ca.ibm.com). The DB2 documentation team reads all of your feedback, but cannot respond to you directly. Provide specific examples wherever possible so that we can better understand your concerns. If you are providing feedback on a specific topic or help file, include the topic title and URL.

Do not use this email address to contact DB2 Customer Support. If you have a DB2 technical issue that the documentation does not resolve, contact your local IBM service center for assistance.

---

## DB2 technical library in hardcopy or PDF format

The following tables describe the DB2 library available from the IBM Publications Center at [www.ibm.com/e-business/linkweb/publications/servlet/pbi.wss](http://www.ibm.com/e-business/linkweb/publications/servlet/pbi.wss). English and translated DB2 Version 10.1 manuals in PDF format can be downloaded from [www.ibm.com/support/docview.wss?rs=71&uid=swg2700947](http://www.ibm.com/support/docview.wss?rs=71&uid=swg2700947).

Although the tables identify books available in print, the books might not be available in your country or region.

The form number increases each time a manual is updated. Ensure that you are reading the most recent version of the manuals, as listed below.

**Note:** The *DB2 Information Center* is updated more frequently than either the PDF or the hard-copy books.

*Table 26. DB2 technical information*

<b>Name</b>	<b>Form Number</b>	<b>Available in print</b>	<b>Last updated</b>
<i>Administrative API Reference</i>	SC27-3864-00	Yes	April, 2012
<i>Administrative Routines and Views</i>	SC27-3865-00	No	April, 2012
<i>Call Level Interface Guide and Reference Volume 1</i>	SC27-3866-00	Yes	April, 2012
<i>Call Level Interface Guide and Reference Volume 2</i>	SC27-3867-00	Yes	April, 2012
<i>Command Reference</i>	SC27-3868-00	Yes	April, 2012
<i>Database Administration Concepts and Configuration Reference</i>	SC27-3871-00	Yes	April, 2012
<i>Data Movement Utilities Guide and Reference</i>	SC27-3869-00	Yes	April, 2012
<i>Database Monitoring Guide and Reference</i>	SC27-3887-00	Yes	April, 2012
<i>Data Recovery and High Availability Guide and Reference</i>	SC27-3870-00	Yes	April, 2012
<i>Database Security Guide</i>	SC27-3872-00	Yes	April, 2012
<i>DB2 Workload Management Guide and Reference</i>	SC27-3891-00	Yes	April, 2012
<i>Developing ADO.NET and OLE DB Applications</i>	SC27-3873-00	Yes	April, 2012
<i>Developing Embedded SQL Applications</i>	SC27-3874-00	Yes	April, 2012
<i>Developing Java Applications</i>	SC27-3875-00	Yes	April, 2012
<i>Developing Perl, PHP, Python, and Ruby on Rails Applications</i>	SC27-3876-00	No	April, 2012
<i>Developing User-defined Routines (SQL and External)</i>	SC27-3877-00	Yes	April, 2012
<i>Getting Started with Database Application Development</i>	GI13-2046-00	Yes	April, 2012

*Table 26. DB2 technical information (continued)*

<b>Name</b>	<b>Form Number</b>	<b>Available in print</b>	<b>Last updated</b>
<i>Getting Started with DB2 Installation and Administration on Linux and Windows</i>	GI13-2047-00	Yes	April, 2012
<i>Globalization Guide</i>	SC27-3878-00	Yes	April, 2012
<i>Installing DB2 Servers</i>	GC27-3884-00	Yes	April, 2012
<i>Installing IBM Data Server Clients</i>	GC27-3883-00	No	April, 2012
<i>Message Reference Volume 1</i>	SC27-3879-00	No	April, 2012
<i>Message Reference Volume 2</i>	SC27-3880-00	No	April, 2012
<i>Net Search Extender Administration and User's Guide</i>	SC27-3895-00	No	April, 2012
<i>Partitioning and Clustering Guide</i>	SC27-3882-00	Yes	April, 2012
<i>pureXML Guide</i>	SC27-3892-00	Yes	April, 2012
<i>Spatial Extender User's Guide and Reference</i>	SC27-3894-00	No	April, 2012
<i>SQL Procedural Languages: Application Enablement and Support</i>	SC27-3896-00	Yes	April, 2012
<i>SQL Reference Volume 1</i>	SC27-3885-00	Yes	April, 2012
<i>SQL Reference Volume 2</i>	SC27-3886-00	Yes	April, 2012
<i>Text Search Guide</i>	SC27-3888-00	Yes	April, 2012
<i>Troubleshooting and Tuning Database Performance</i>	SC27-3889-00	Yes	April, 2012
<i>Upgrading to DB2 Version 10.1</i>	SC27-3881-00	Yes	April, 2012
<i>What's New for DB2 Version 10.1</i>	SC27-3890-00	Yes	April, 2012
<i>XQuery Reference</i>	SC27-3893-00	No	April, 2012

*Table 27. DB2 Connect-specific technical information*

<b>Name</b>	<b>Form Number</b>	<b>Available in print</b>	<b>Last updated</b>
<i>DB2 Connect Installing and Configuring DB2 Connect Personal Edition</i>	SC27-3861-00	Yes	April, 2012
<i>DB2 Connect Installing and Configuring DB2 Connect Servers</i>	SC27-3862-00	Yes	April, 2012
<i>DB2 Connect User's Guide</i>	SC27-3863-00	Yes	April, 2012

---

## Displaying SQL state help from the command line processor

DB2 products return an SQLSTATE value for conditions that can be the result of an SQL statement. SQLSTATE help explains the meanings of SQL states and SQL state class codes.

### Procedure

To start SQL state help, open the command line processor and enter:

```
? sqlstate or ? class code
```

where *sqlstate* represents a valid five-digit SQL state and *class code* represents the first two digits of the SQL state.

For example, ? 08003 displays help for the 08003 SQL state, and ? 08 displays help for the 08 class code.

---

## Accessing different versions of the DB2 Information Center

Documentation for other versions of DB2 products is found in separate information centers on [ibm.com](http://ibm.com)<sup>®</sup>.

### About this task

For DB2 Version 10.1 topics, the *DB2 Information Center* URL is <http://publib.boulder.ibm.com/infocenter/db2luw/v10r1>.

For DB2 Version 9.8 topics, the *DB2 Information Center* URL is <http://publib.boulder.ibm.com/infocenter/db2luw/v9r8/>.

For DB2 Version 9.7 topics, the *DB2 Information Center* URL is <http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/>.

For DB2 Version 9.5 topics, the *DB2 Information Center* URL is <http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/>.

For DB2 Version 9.1 topics, the *DB2 Information Center* URL is <http://publib.boulder.ibm.com/infocenter/db2luw/v9/>.

For DB2 Version 8 topics, go to the *DB2 Information Center* URL at: <http://publib.boulder.ibm.com/infocenter/db2luw/v8/>.

---

## Updating the DB2 Information Center installed on your computer or intranet server

A locally installed DB2 Information Center must be updated periodically.

### Before you begin

A DB2 Version 10.1 Information Center must already be installed. For details, see the “Installing the DB2 Information Center using the DB2 Setup wizard” topic in *Installing DB2 Servers*. All prerequisites and restrictions that applied to installing the Information Center also apply to updating the Information Center.

## About this task

An existing DB2 Information Center can be updated automatically or manually:

- Automatic updates update existing Information Center features and languages. One benefit of automatic updates is that the Information Center is unavailable for a shorter time compared to during a manual update. In addition, automatic updates can be set to run as part of other batch jobs that run periodically.
- Manual updates can be used to update existing Information Center features and languages. Automatic updates reduce the downtime during the update process, however you must use the manual process when you want to add features or languages. For example, a local Information Center was originally installed with both English and French languages, and now you want to also install the German language; a manual update will install German, as well as, update the existing Information Center features and languages. However, a manual update requires you to manually stop, update, and restart the Information Center. The Information Center is unavailable during the entire update process. In the automatic update process the Information Center incurs an outage to restart the Information Center after the update only.

This topic details the process for automatic updates. For manual update instructions, see the “Manually updating the DB2 Information Center installed on your computer or intranet server” topic.

## Procedure

To automatically update the DB2 Information Center installed on your computer or intranet server:

1. On Linux operating systems,
  - a. Navigate to the path where the Information Center is installed. By default, the DB2 Information Center is installed in the `/opt/ibm/db2ic/V10.1` directory.
  - b. Navigate from the installation directory to the `doc/bin` directory.
  - c. Run the `update-ic` script:

```
update-ic
```
2. On Windows operating systems,
  - a. Open a command window.
  - b. Navigate to the path where the Information Center is installed. By default, the DB2 Information Center is installed in the `<Program Files>\IBM\DB2 Information Center\Version 10.1` directory, where `<Program Files>` represents the location of the Program Files directory.
  - c. Navigate from the installation directory to the `doc\bin` directory.
  - d. Run the `update-ic.bat` file:

```
update-ic.bat
```

## Results

The DB2 Information Center restarts automatically. If updates were available, the Information Center displays the new and updated topics. If Information Center updates were not available, a message is added to the log. The log file is located in `doc\eclipse\configuration` directory. The log file name is a randomly generated number. For example, `1239053440785.log`.

---

## Manually updating the DB2 Information Center installed on your computer or intranet server

If you have installed the DB2 Information Center locally, you can obtain and install documentation updates from IBM.

### About this task

Updating your locally installed *DB2 Information Center* manually requires that you:

1. Stop the *DB2 Information Center* on your computer, and restart the Information Center in stand-alone mode. Running the Information Center in stand-alone mode prevents other users on your network from accessing the Information Center, and allows you to apply updates. The Workstation version of the DB2 Information Center always runs in stand-alone mode. .
2. Use the Update feature to see what updates are available. If there are updates that you must install, you can use the Update feature to obtain and install them

**Note:** If your environment requires installing the *DB2 Information Center* updates on a machine that is not connected to the internet, mirror the update site to a local file system by using a machine that is connected to the internet and has the *DB2 Information Center* installed. If many users on your network will be installing the documentation updates, you can reduce the time required for individuals to perform the updates by also mirroring the update site locally and creating a proxy for the update site.

If update packages are available, use the Update feature to get the packages. However, the Update feature is only available in stand-alone mode.

3. Stop the stand-alone Information Center, and restart the *DB2 Information Center* on your computer.

**Note:** On Windows 2008, Windows Vista (and higher), the commands listed later in this section must be run as an administrator. To open a command prompt or graphical tool with full administrator privileges, right-click the shortcut and then select **Run as administrator**.

### Procedure

To update the *DB2 Information Center* installed on your computer or intranet server:

1. Stop the *DB2 Information Center*.
  - On Windows, click **Start > Control Panel > Administrative Tools > Services**. Then right-click **DB2 Information Center** service and select **Stop**.
  - On Linux, enter the following command:  

```
/etc/init.d/db2icdv10 stop
```
2. Start the Information Center in stand-alone mode.
  - On Windows:
    - a. Open a command window.
    - b. Navigate to the path where the Information Center is installed. By default, the *DB2 Information Center* is installed in the *Program\_Files\IBM\DB2 Information Center\Version 10.1* directory, where *Program\_Files* represents the location of the Program Files directory.
    - c. Navigate from the installation directory to the `doc\bin` directory.
    - d. Run the `help_start.bat` file:



```
help_start.bat
```

- On Linux:
  - a. Navigate to the path where the Information Center is installed. By default, the *DB2 Information Center* is installed in the `/opt/ibm/db2ic/V10.1` directory.
  - b. Navigate from the installation directory to the `doc/bin` directory.
  - c. Run the `help_start` script:

```
help_start
```

The systems default Web browser opens to display the stand-alone Information Center.

3. Click the **Update** button (🔧). (JavaScript must be enabled in your browser.) On the right panel of the Information Center, click **Find Updates**. A list of updates for existing documentation displays.
4. To initiate the installation process, check that the selections you want to install, then click **Install Updates**.
5. After the installation process has completed, click **Finish**.
6. Stop the stand-alone Information Center:
  - On Windows, navigate to the `doc\bin` directory within the installation directory, and run the `help_end.bat` file:

```
help_end.bat
```

**Note:** The `help_end` batch file contains the commands required to safely stop the processes that were started with the `help_start` batch file. Do not use `Ctrl-C` or any other method to stop `help_start.bat`.
  - On Linux, navigate to the `doc/bin` directory within the installation directory, and run the `help_end` script:

```
help_end
```

**Note:** The `help_end` script contains the commands required to safely stop the processes that were started with the `help_start` script. Do not use any other method to stop the `help_start` script.
7. Restart the *DB2 Information Center*.
  - On Windows, click **Start > Control Panel > Administrative Tools > Services**. Then right-click **DB2 Information Center** service and select **Start**.
  - On Linux, enter the following command:

```
/etc/init.d/db2icdv10 start
```

## Results

The updated *DB2 Information Center* displays the new and updated topics.

---

## DB2 tutorials

The DB2 tutorials help you learn about various aspects of DB2 database products. Lessons provide step-by-step instructions.

### Before you begin

You can view the XHTML version of the tutorial from the Information Center at <http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/>.

Some lessons use sample data or code. See the tutorial for a description of any prerequisites for its specific tasks.

## DB2 tutorials

To view the tutorial, click the title.

### “pureXML” in *pureXML Guide*

Set up a DB2 database to store XML data and to perform basic operations with the native XML data store.

---

## DB2 troubleshooting information

A wide variety of troubleshooting and problem determination information is available to assist you in using DB2 database products.

### DB2 documentation

Troubleshooting information can be found in the *Troubleshooting and Tuning Database Performance* or the Database fundamentals section of the *DB2 Information Center*, which contains:

- Information about how to isolate and identify problems with DB2 diagnostic tools and utilities.
- Solutions to some of the most common problem.
- Advice to help solve other problems you might encounter with your DB2 database products.

### IBM Support Portal

See the IBM Support Portal if you are experiencing problems and want help finding possible causes and solutions. The Technical Support site has links to the latest DB2 publications, TechNotes, Authorized Program Analysis Reports (APARs or bug fixes), fix packs, and other resources. You can search through this knowledge base to find possible solutions to your problems.

Access the IBM Support Portal at [http://www.ibm.com/support/entry/portal/Overview/Software/Information\\_Management/DB2\\_for\\_Linux,\\_UNIX\\_and\\_Windows](http://www.ibm.com/support/entry/portal/Overview/Software/Information_Management/DB2_for_Linux,_UNIX_and_Windows)

---

## Terms and conditions

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

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

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

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

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

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

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

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

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



---

## Appendix E. Notices

This information was developed for products and services offered in the U.S.A. Information about non-IBM products is based on information available at the time of first publication of this document and is subject to change.

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

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

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

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

Intellectual Property Licensing  
Legal and Intellectual Property Law  
IBM Japan, Ltd.  
1623-14, Shimotsuruma, Yamato-shi  
Kanagawa 242-8502 Japan

**The following paragraph does not apply to the United Kingdom or any other country/region 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, changes, or both in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to websites not owned by IBM are provided for convenience only and do not in any manner serve as an endorsement of those

websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

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

Licenses 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 that has been exchanged, should contact:

IBM Canada Limited  
U59/3600  
3600 Steeles Avenue East  
Markham, Ontario L3R 9Z7  
CANADA

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

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

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

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

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

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

#### COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating

platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work must include a copyright notice as follows:

© (*your company name*) (*year*). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. *\_enter the year or years\_*. All rights reserved.

## Trademarks

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

The following terms are trademarks or registered trademarks of other companies

- Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.
- Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle, its affiliates, or both.
- UNIX is a registered trademark of The Open Group in the United States and other countries.
- Intel, Intel logo, Intel Inside, Intel Inside logo, Celeron, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.
- Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

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





---

# Index

## A

ALTER INDEX Text Search command 130

## C

cataloging

TCP/IP nodes 63

CLEANUP FOR TEXT Text Search command 134

CLEAR COMMAND LOCKS Text Search command 135

CLEAR EVENTS FOR INDEX Text Search command 137

commands

calling from procedure 159

db2icrt 216

db2idrop 224

db2iupdt 226

db2iupgrade

details 213

db2ts ALTER INDEX 130

db2ts CLEANUP FOR TEXT 134

db2ts CLEAR COMMAND LOCKS 135

db2ts CLEAR EVENTS FOR INDEX 137

db2ts CREATE INDEX 138

db2ts DISABLE DATABASE FOR TEXT 145

db2ts DROP INDEX 147

db2ts ENABLE DATABASE FOR TEXT 148

db2ts HELP 150

db2ts RESET PENDING 152

db2ts START FOR TEXT 153

db2ts STOP FOR TEXT 154

db2ts UPDATE INDEX 155

CREATE INDEX Text Search command 138

create instance command 216

## D

DB2 Information Center

updating 238, 240

versions 238

DB2 servers

installing

Windows 46

DB2 Setup wizard

installing

DB2 servers (Linux)DB2 servers (UNIX) 49

DB2 Text Search 55

adding a synonym dictionary 80

administration 89, 129

administration commands 89, 129

summary 89, 129

administrative routines 90, 159

administrative views

database level 197, 198

event table 202

index level 197, 199, 201, 202, 203

log table 203

staging table 203

SYSIBMTS.TSCOLLECTIONNAMES 201

SYSIBMTS.TSCONFIGURATION 201

SYSIBMTS.TSDEFAULTS 197

SYSIBMTS.TSEVENT 202

DB2 Text Search (*continued*)

administrative views (*continued*)

SYSIBMTS.TSINDEXES 199

SYSIBMTS.TSLOCKS 198

SYSIBMTS.TSSERVERS 199

SYSIBMTS.TSSTAGING 203

ALTER INDEX command 130

altering a text search index 94

altering indexes 94

asynchronous indexing 30

authorizations

database administrator 23

instance owner 23

roles 22

user performing text search queries 23

backup 96

basic search 101

capacity planning

TCP/IP port requirements 33

capacity planning and optimization 25

changing the location of a collection 95

changing the location of collections 95

changing update characteristics 94

CLEAR COMMAND LOCKS command 135

CLEAR EVENTS FOR INDEX command 137

clearing text search index events 93

code pages supported 20

command-line tools 73

commands

ALTER INDEX 130

CLEANUP FOR TEXT 134

CLEAR COMMAND LOCKS 135

CLEAR EVENTS FOR INDEX 137

CREATE INDEX 138

DISABLE DATABASE FOR TEXT 145

DROP INDEX 147

ENABLE DATABASE FOR TEXT 148

HELP 150

RESET PENDING 152

START FOR TEXT 153

STOP FOR TEXT 154

UPDATE INDEX 155

Configuration Tool 59

configuration tuning 25

configuring

Configuration Tool 59

methods 57

overview 41

response file 45

Setup Wizard 44

configuring a stand-alone server 61

CONTAINS function 119

CREATE INDEX command 138

data types

converting unsupported 19

supported 19

DISABLE DATABASE FOR TEXT command 145

disabling a database for text 77

disabling databases for 77

disabling rich text support 74

disk consumption 30

- DB2 Text Search *(continued)*
  - document formats
    - converting unsupported 19
    - supported 19
  - document truncation 20
  - DROP INDEX command 147
  - dropping a text search index 97
  - dropping indexes 97
  - ENABLE DATABASE FOR TEXT command 148
  - enabling a database for text 76
  - enabling databases for 76
  - enabling rich text support 74
  - escaping special characters 103
  - event tables
    - overview 81
    - removing messages 93
  - file descriptors 34
  - filter libraries 63
  - functions 99
  - fuzzy search 101
  - hardware and software 43
  - hardware requirements 43
  - heap memory consumption 26
  - HELP command 150
  - improving search performance 116
  - incremental index updates 91
  - index location 32
  - Index Manager 23
  - index optimization 30
  - index planning and optimization 29
  - index source 29
  - index source characteristics 29
  - index specific parameters for updates 32
  - indexes
    - altering 6
    - binary data types 83
    - creating 6, 81, 82
    - creating (binary data types) 83
    - creating (unsupported data types) 84
    - incremental updates 11
    - maintaining 88
    - searching 100
    - updating 6
  - indexing special characters 105
  - indexing threads 27
  - installation disk space requirements 54
  - installing
    - DB2 Accessories Suite filter libraries 63
    - db2\_install command 46
    - overview 41
    - response file 45
    - Setup Wizard 44
  - installing Accessories Suite 63
  - installing and configuring a stand-alone server 54
  - installing updates 72
  - integrated server 4
  - issuing commands 73
  - key features and concepts 3
  - languages supported 20
  - linguistic processing 12
    - using n-gram segmentation 12
  - locale and language 36
  - log tables 81
  - maximum heap size 26
  - morphological indexing on plain text 85, 86
  - multiple predicates 36
  - Net Search Extender comparison 205
- DB2 Text Search *(continued)*
  - non-root upgrade 70
  - optimizing a text index 30
  - overview 1, 19, 205
  - parser configuration 38
  - partitioned database environments 9
  - performance 35
  - planning and optimization 29
  - planning queries 35
  - proximity search 102
  - queue memory size 28
  - reconfiguring 57, 59
  - removing synonym dictionary 81
  - RESET PENDING command 152
  - restore 96
  - RESULTLIMIT function 37
  - rich text 74
    - overview 17
  - rich text document support 17, 63
  - roles
    - database administrator 23
    - instance owner 23
    - user performing searches 23
  - scenario 14
  - scheduling a task 98
  - SCORE function 37, 121
  - search arguments
    - performance implications 35
    - syntax 108
  - searching indexes 99
  - searching special characters 103
  - searching with the SCORE function 108
  - security overview 21, 24
  - server configuration 25
  - special characters adjacent to query terms 104
  - special characters in CJK languages 105
  - SQL 100
  - SQL search functions 119
  - stand-alone installation 46
  - stand-alone server 4
    - configuration 61
  - START FOR TEXT command 153
  - starting 75
  - STOP FOR TEXT command 154
  - stopping instance services 75
  - synonym dictionaries
    - overview 79
  - system tuning 33
  - text search collections
    - deleting orphaned 78
    - identifying orphaned 78
  - text search index specific parameters 32
  - triggers 30, 81
  - uninstallation 56
  - uninstalling DB2 Accessories Suite 65
  - unsupported data types 84
  - UPDATE INDEX command 155
  - updating a text index 90
  - updating server information 60
  - upgrade 67
  - upgrading a multi-partition instance 71
  - user roles 22
  - viewing index status 95
  - XML columns 124
  - XML documents 106, 112
  - XML namespaces 38
  - XML search functions 119

- DB2 Text Search (*continued*)
  - XQuery 100
    - xmlcolumn-contains 124
- db2icrt command
  - details 216
- db2idrop command
  - details 224
- db2iupdt command
  - details 226
- db2iupgrade command
  - details 213
- db2ts commands
  - ALTER INDEX 130
  - CLEANUP FOR TEXT 134
  - CLEAR COMMAND LOCKS 135
  - CLEAR EVENTS FOR INDEX 137
  - CREATE INDEX 138
  - DISABLE DATABASE FOR TEXT 145
  - DROP INDEX 147
  - ENABLE DATABASE FOR TEXT 148
  - HELP 150
  - RESET PENDING 152
  - START FOR TEXT 153
  - STOP FOR TEXT 154
  - UPDATE INDEX 155
- DISABLE DATABASE FOR TEXT Text Search command 145
- documentation
  - overview 235
  - PDF files 235
  - printed 235
  - terms and conditions of use 242
- DROP INDEX Text Search command 147

## E

- ENABLE DATABASE FOR TEXT Text Search command 148

## F

- functions
  - stored procedures
    - SYSTS\_ALTER 161
    - SYSTS\_CLEANUP 166
    - SYSTS\_CLEAR\_COMMANDLOCKS 167
    - SYSTS\_CLEAR\_EVENTS 169
    - SYSTS\_DROP 185
    - SYSTS\_ENABLE 187
    - SYSTS\_UPDATE 189
    - SYSTS\_UPGRADE\_CATALOG 192
    - SYSTS\_UPGRADE\_INDEX 194

## H

- help
  - SQL statements 238
- HELP command
  - Text Search 150

## I

- installation
  - silent
    - Linux 53
    - UNIX 53
    - Windows 52

## L

- Linux
  - installing
    - DB2 servers 49
    - response file 53

## N

- Net Search Extender
  - comparison with Text Search 205
  - notices 245

## P

- problem determination
  - information available 242
  - tutorials 242
- procedures
  - SYSTS\_ADMIN\_CMD 159
  - SYSTS\_CREATE 174
  - SYSTS\_START 172

## R

- remove instance command 224
- RESET PENDING DB2 Text Search command 152
- response files
  - installation
    - Linux 53
    - UNIX 53
    - Windows 52

## S

- SCORE function
  - searching text search indexes 121
- Searching for special characters 103
- services file
  - updating for TCP/IP communications 63
- silent installation
  - Linux 53
  - UNIX 53
  - Windows 52
- software requirements
  - DB2 Text Search 43
- Special characters adjacent to query terms 104
- SQL statements
  - help
    - displaying 238
  - stand-alone installation 54
  - START FOR TEXT Text Search command 153
  - STOP FOR TEXT Text Search command 154
  - synonym dictionaries
    - adding 80
    - overview 79
    - removing 81
  - SYSBMTS.TSINDEXES view 199
  - SYSBMTS.TSSERVERS 199
  - SYSTS\_ADMIN\_CMD procedure 159
  - SYSTS\_ALTER stored procedure 161
  - SYSTS\_CLEANUP stored procedure 166
  - SYSTS\_CLEAR\_COMMANDLOCKS stored procedure 167
  - SYSTS\_CLEAR\_EVENTS stored procedure 169
  - SYSTS\_CREATE procedure 174
  - SYSTS\_DISABLE procedure 183

- SYSTS\_DROP stored procedure 185
- SYSTS\_ENABLE stored procedure 187
- SYSTS\_START procedure 172
- SYSTS\_UPDATE stored procedure 189
- SYSTS\_UPGRADE\_CATALOG stored procedure 192
- SYSTS\_UPGRADE\_INDEX stored procedure 194

## T

- TCP/IP
  - updating services file 63
- terms and conditions
  - publications 242
- text indexes
  - proximity search 102
- text search
  - CONTAINS function 99
  - SCORE function 99
  - search functions 99
  - xmlcolumn-contains function 99
- Text Search 25
- text searches 74
- troubleshooting
  - online information 242
  - tutorials 242
- tuning 25
- tutorials
  - list 241
  - problem determination 242
  - pureXML 241
  - troubleshooting 242

## U

- UNIX
  - installing
    - DB2 Setup wizard 49
    - response file installation 53
- UPDATE INDEX Text Search command 155
- update instances command 226
- updates
  - DB2 Information Center 238, 240
- upgrade instance command 213

## V

- views
  - DB2 Text Search
    - index-level information 199
    - SYSIBMTS.TSINDEXES 199
- views for DB2 Text Search
  - database-level information
    - overview 197
    - SYSIBMTS.TSDEFAULTS 197
    - SYSIBMTS.TSLOCKS 198
  - index-level information
    - overview 197
    - SYSIBMTS.TSCOLLECTIONNAMES 201
    - SYSIBMTS.TSCONFIGURATION 201
    - SYSIBMTS.TSEVENT 202
    - SYSIBMTS.TSSTAGING 203

## W

- Windows
  - installing
    - DB2 servers (with DB2 Setup wizard) 46
    - response files
      - installing using 52
  - Windows service 55

## X

- XML
  - DB2 Text Search
    - EBNF grammar 106
    - search syntax 112
  - XML columns
    - text search 124
  - XML namespaces 38
  - xmlcolumn-contains function 124
- XQuery functions
  - xmlcolumn-contains 124





Printed in USA

SC27-3888-00



Spine information:

IBM DB2 10.1 for Linux, UNIX, and Windows

Text Search Guide

