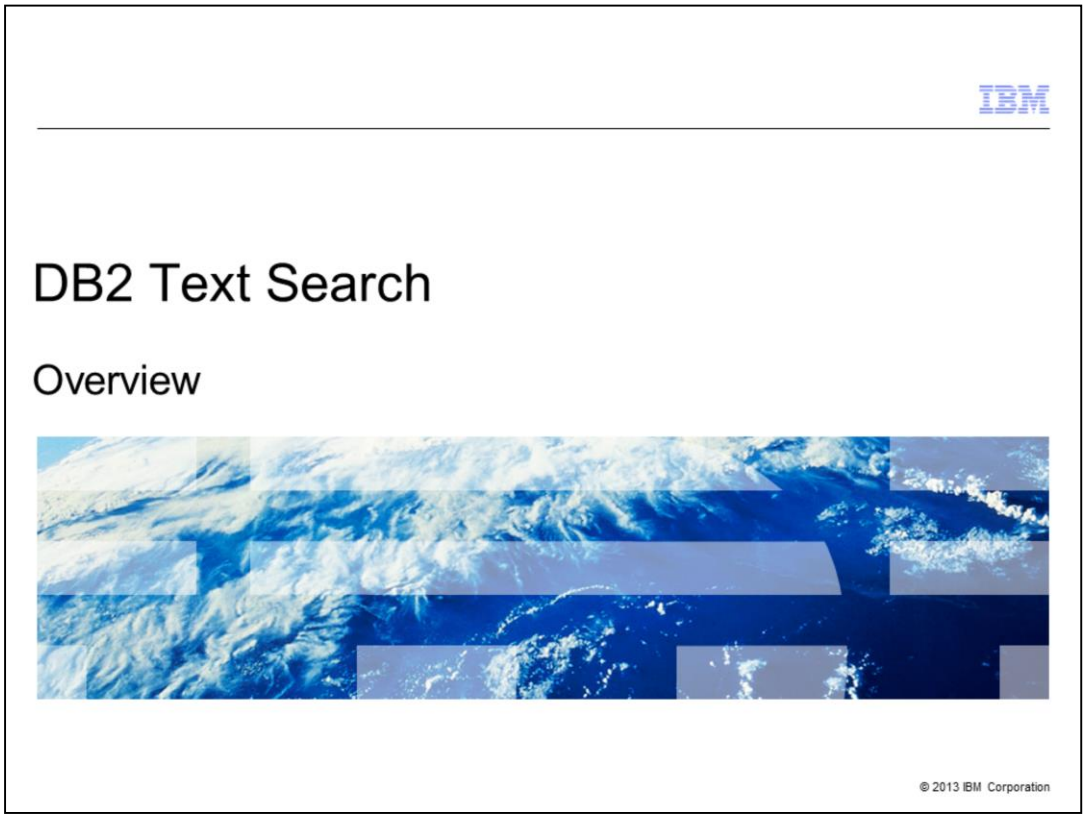
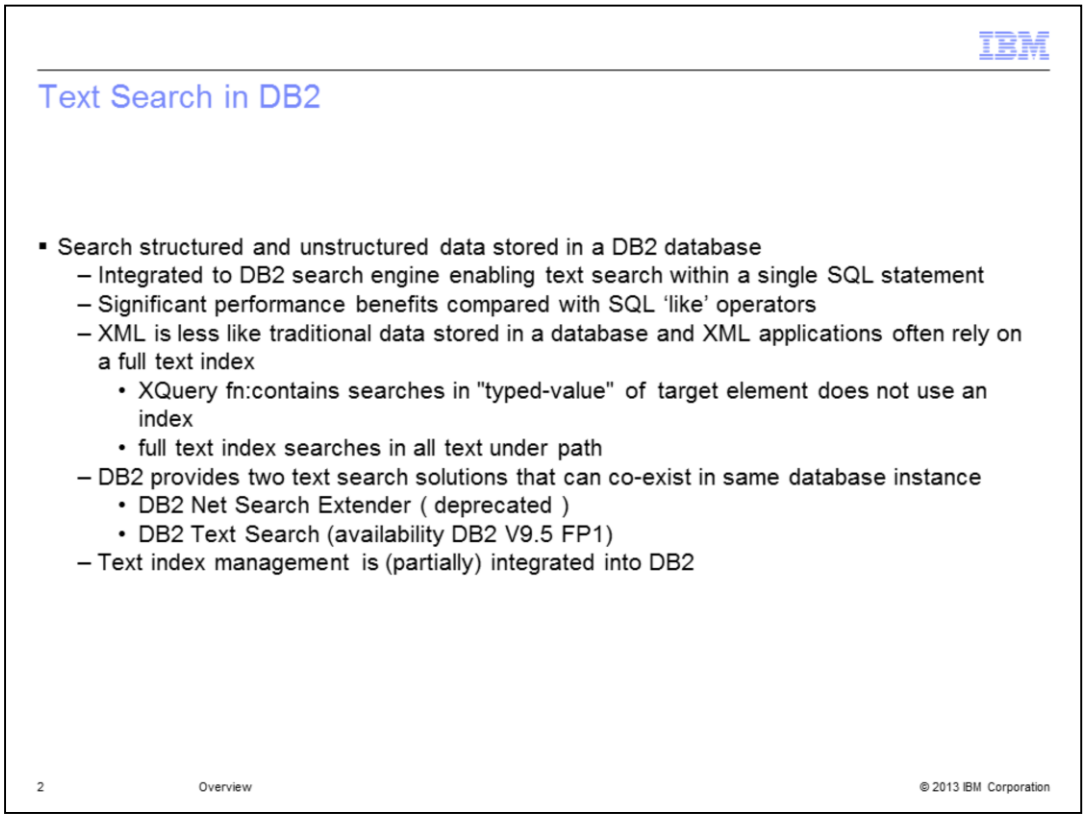


DB2® Text Search version 10.5 provides a rich set of full-text search features. This presentation provides a short overview of some of the main features.

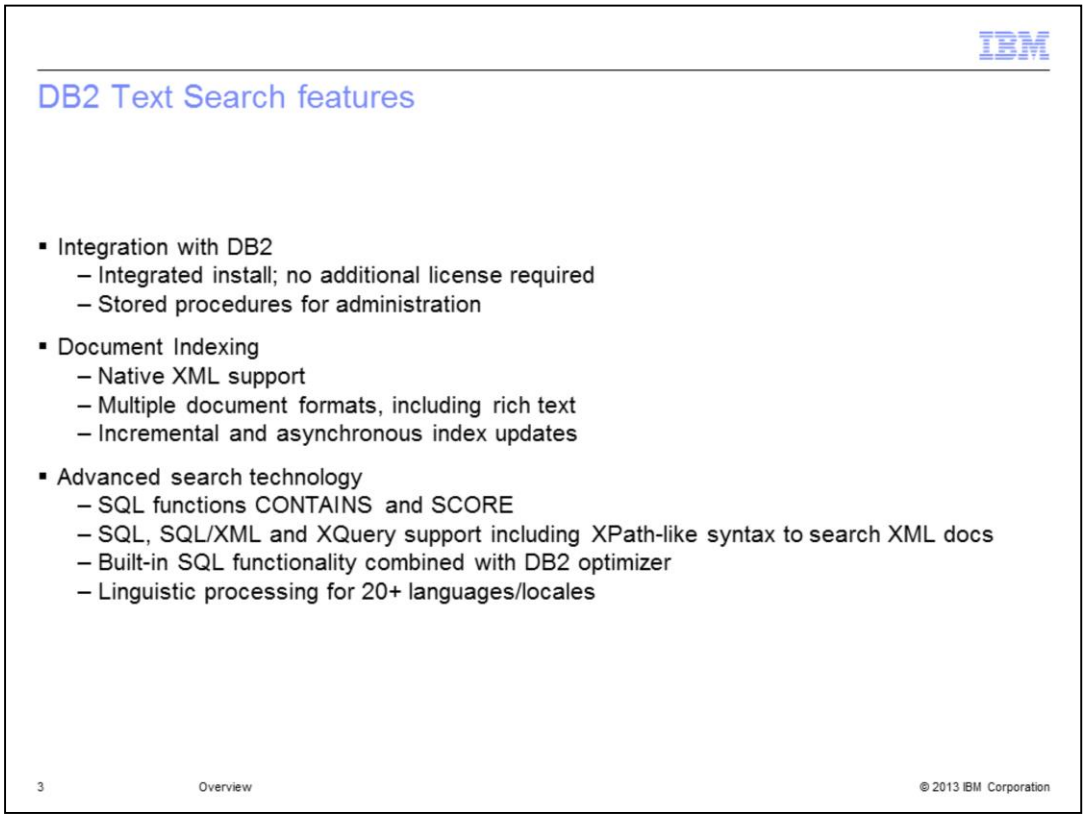


DB2 Text Search provides extensive capabilities to search both structured and unstructured data stored in a DB2 database. The Text Search functions CONTAINS(), SCORE() and xmlcolumn-contains(), are tightly integrated with DB2 such that they can be used within SQL and XQuery statements.

Text indexes offer significant performance benefits compared to SQL 'LIKE' operator. DB2 Text Search supports pureXML® and searches the data using SQL, SQL/XML and XQuery statements. The XML search functionality provided by DB2 Text Search makes it highly attractive due to its performance benefits.

DB2 provides two text search solutions that can co-exist in the same database instance. One is the DB2 Net Search Extender, or DB2 NSE, and it is already deprecated from version 10.1. The second one is the DB2 Text Search which is available since DB2 V9.5 FP1 and is the new powerful Text Search engine.

Text index management is partially integrated with DB2, that is, certain table operations like drop table, are restricted when a text index is created and in an active state.



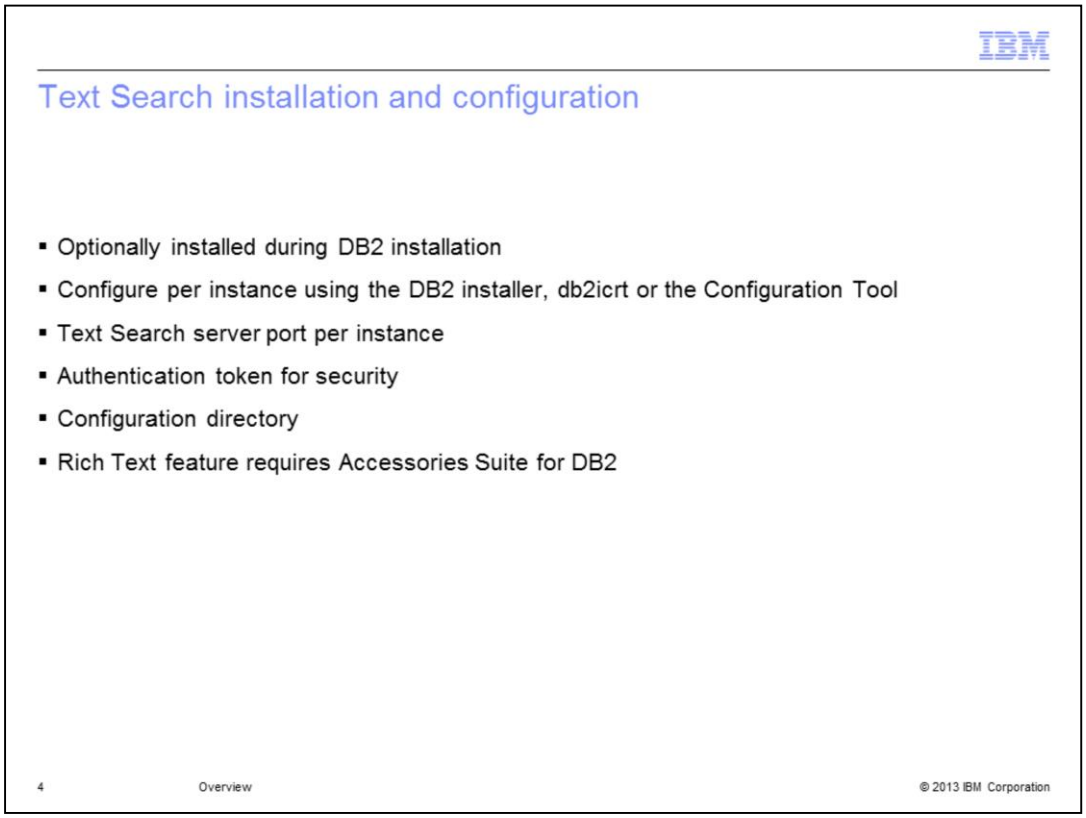
DB2 Text Search is an optionally installable component whose installation and configuration is fully integrated with DB2 server products. No additional license is required. It also provides several administrative stored procedures that can be called from any DB2 client machine.

The document formats supported are Text, HTML and XML. In addition, DB2 Text Search supports indexing and searching of rich text documents with the DB2 Accessories Suite.

The text index update is asynchronous and not part of the same transaction with which the data column in the base table has been updated.

DB2 Text Search provides advanced search technology by combining the SQL functionality with DB2 optimizer where least expensive plans are chosen.

SQL, SQL/XML and XQuery support is provided including XPath like syntax to search XML documents. It also supports advanced linguistic processing for 20 plus languages and locales.



One DB2 instance is associated with only one DB2 Text Search server instance. The instance should be initially configured for Text Search using DB2 installer or using the –j option with db2icrt or db2iupdt or db2iupgrade.

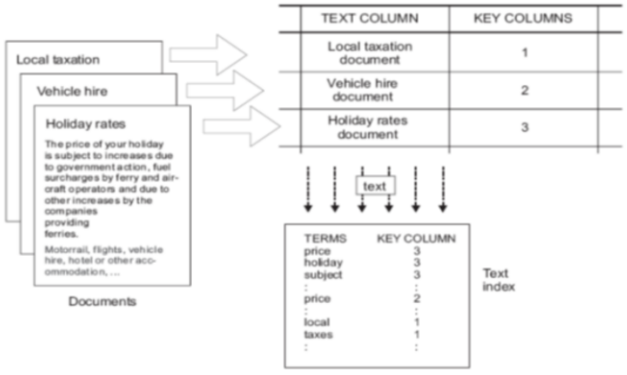
The configuration tool, referred to as configTool, can be used to do initial and subsequent configuration of DB2 Text Search. It authenticates with the DB2 Text Server for all administration commands and searches using authentication token, and the communication with the Text Server happens through HTTP Port. The DB2 Text Search instance should be configured by generating the authentication token and by setting a valid port number using the configTool.

All the configuration files are located under $INSTHOME/*sqllib*/*db2tss*/*config* where, INSTHOME is the *instance home directory*.

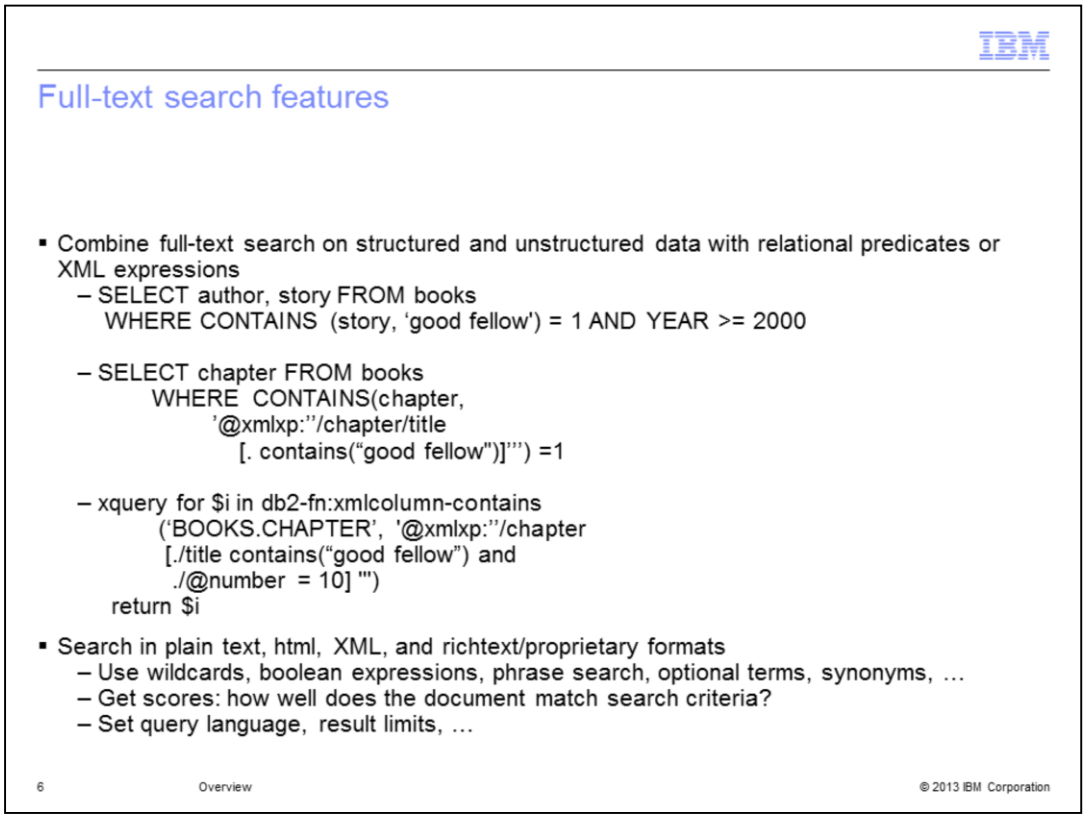
The DB2 Accessories Suite should be downloaded separately to support rich text documents.

Text indexes

- Search data in plain text, HTML, XML and rich text documents
- Text index keeps track of significant terms from text documents



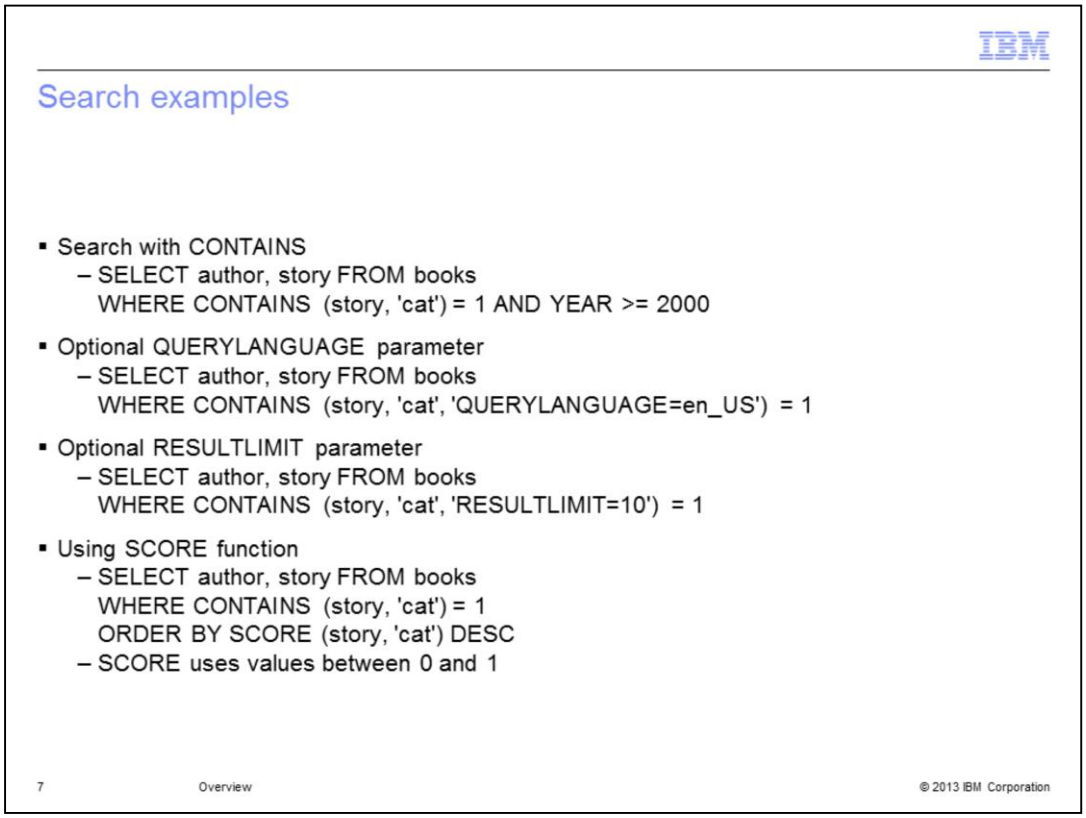
Text index must be created in order to efficiently search the text documents. The parser extracts significant terms from the document, which are used to build a text index. Any changes to the table such as inserts, updates and deletes on the documents, are tracked and reflected in the text index with the next incremental update.



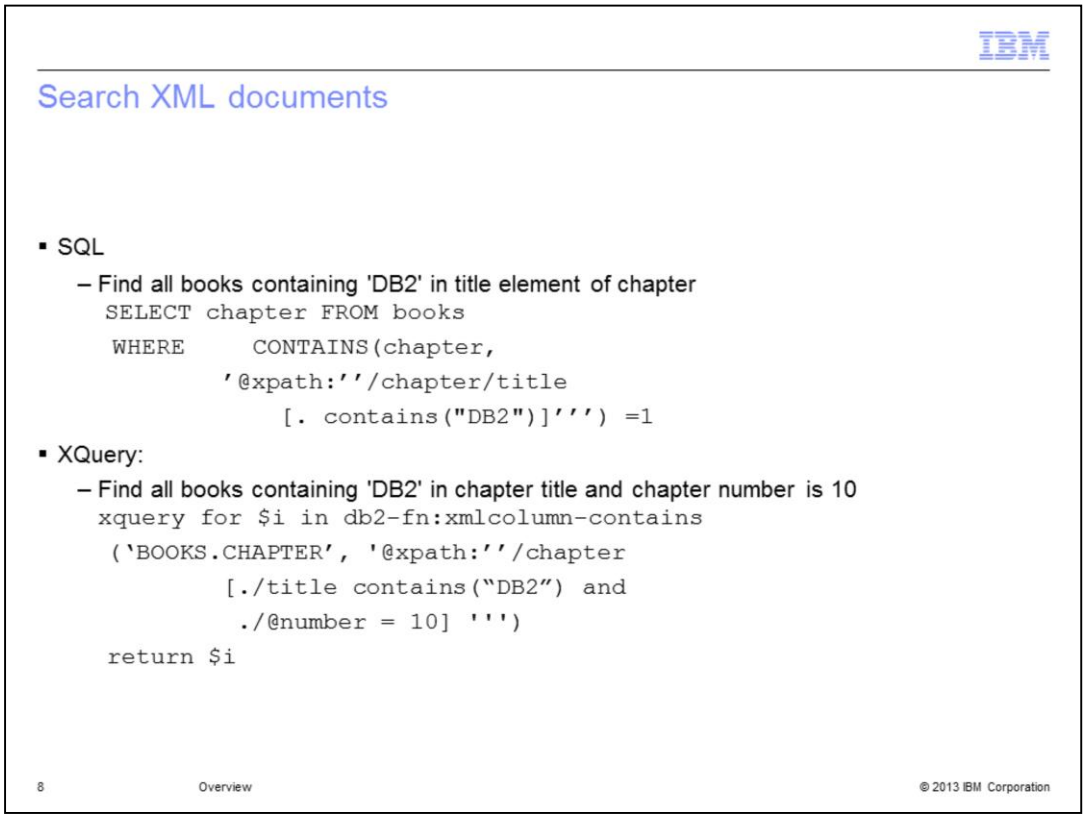
DB2 Text Search provides several features and functionality to search the structured and unstructured data. The examples displayed on this slide show how to perform searches using SQL/XML queries. Xmlxp is used to search specific portions of an XML document.

xmlcolumn-contains() is used in an XQuery expression to search for a specific term and attribute.

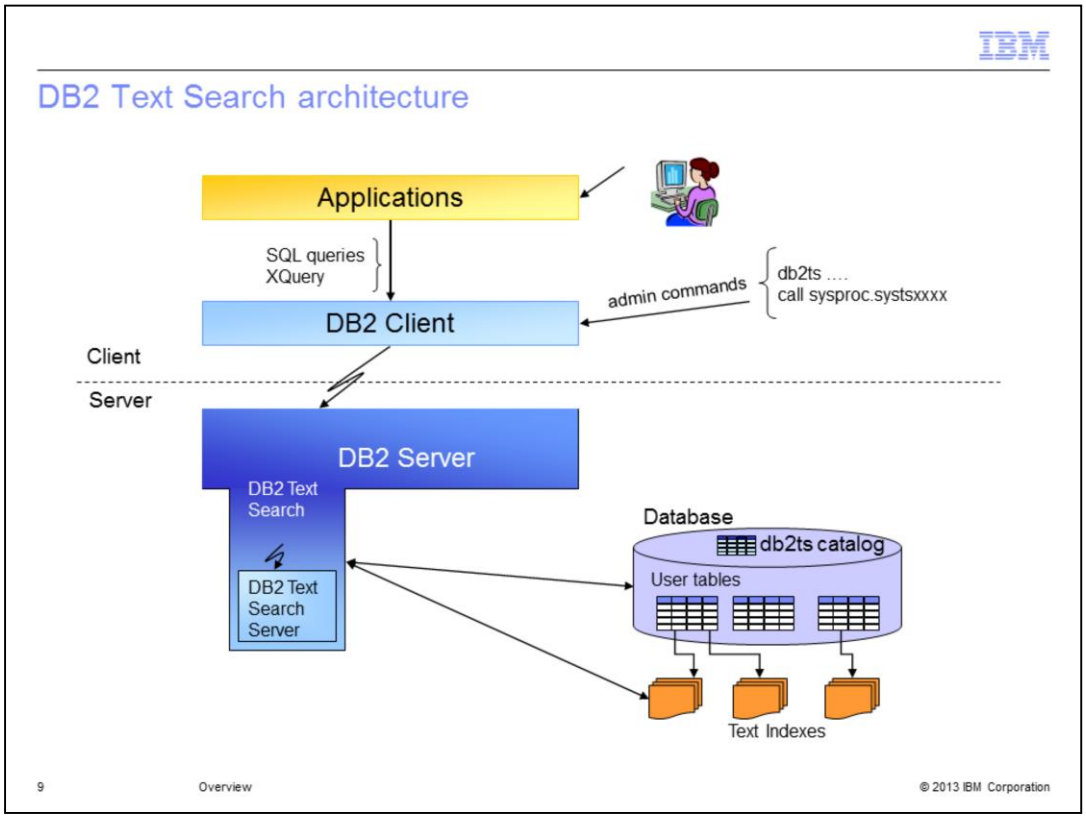
The search scope can either be extended or reduced using wild cards, Boolean expressions, phrase search, optional terms, synonyms, and so on. Other features include getting the score of a document, ability to set query language, result limit and so on.



A few more search examples are displayed on this slide. The CONTAINS() function searches the text index using the search criteria specified in the query and returns one when there is a match found. The QUERYLANGUAGE parameter is used to specify the locale of the search term. The RESULTLIMIT parameter is used to limit the number of results received. SCORE() returns a score of each document found indicating how well the document matches the given search criteria compared to other documents in the same column. It uses a value between zero and one.



The first example displayed on this slide uses the XPath syntax within the SQL CONTAINS() function to search all books containing 'DB2' in the title element of a chapter. The second example uses XQuery expression with the xmlcolumn-contains() function to search all books containing 'DB2' in the chapter title and the chapter number is 10.



The picture displayed on this slide represents the DB2 Text Search architecture. In this case, DB2 Server and DB2 Text Search Server are located on the same machine.

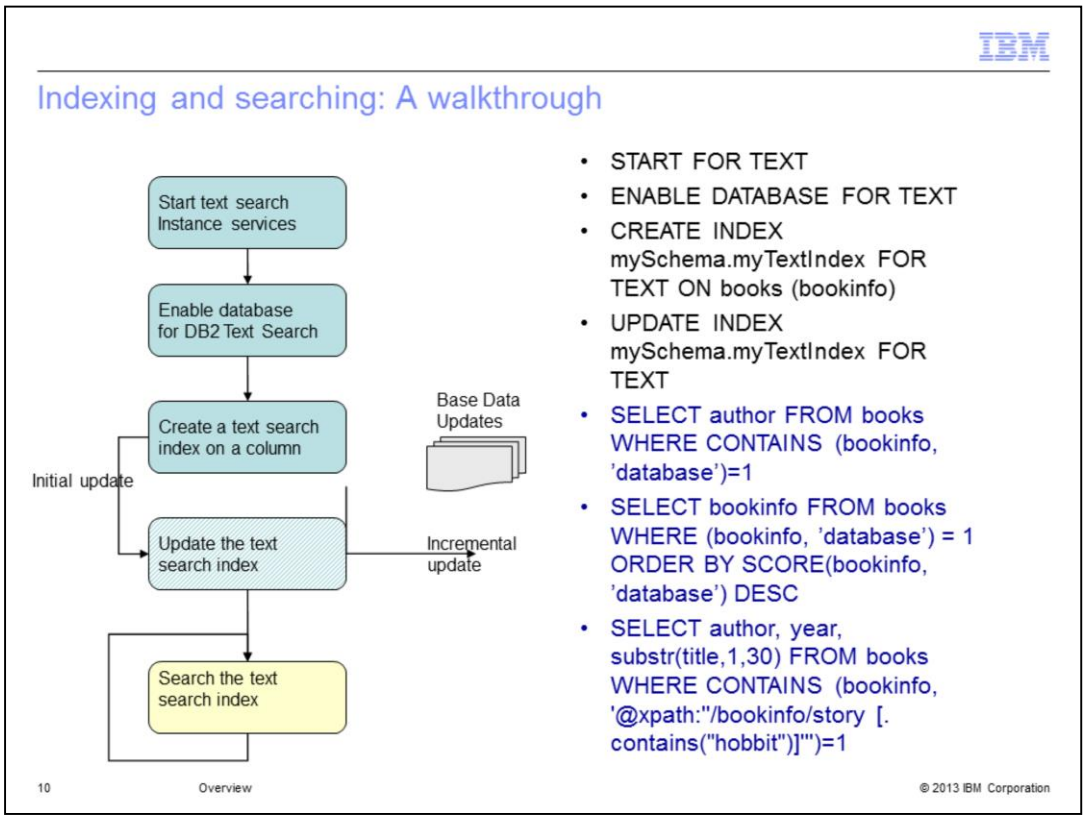
When a database is enabled for text search, catalog tables and views are created in 'SYSIBMTS' schema.

Text indexes are managed as file system objects and the low level representation of an index is known as a collection.

Once a text index is created on the user table, the staging table and log table are created.

The client applications can issue searches by way of SQL and XQuery statements.

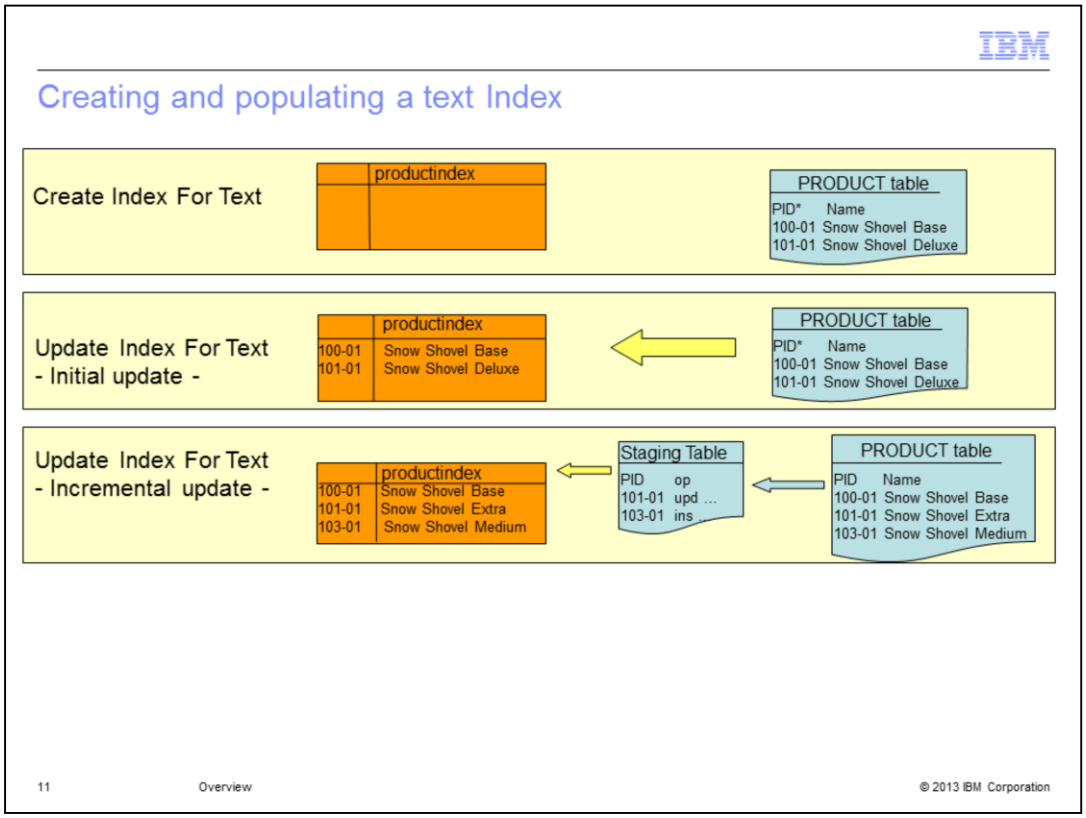
The administration commands can be ran on the DB2 server by running the db2ts command or by calling stored procedure from the DB2 client.



This slide displays a view of a walkthrough example on indexing and searching. First, the DB2 Text Search instance service should be started and then, the database must be enabled so that the text search functionality is used. The text index should be created and updated so that the searches can be issued.

The first update, known as the initial index update, reads the data directly from the text column and adds it to the index.

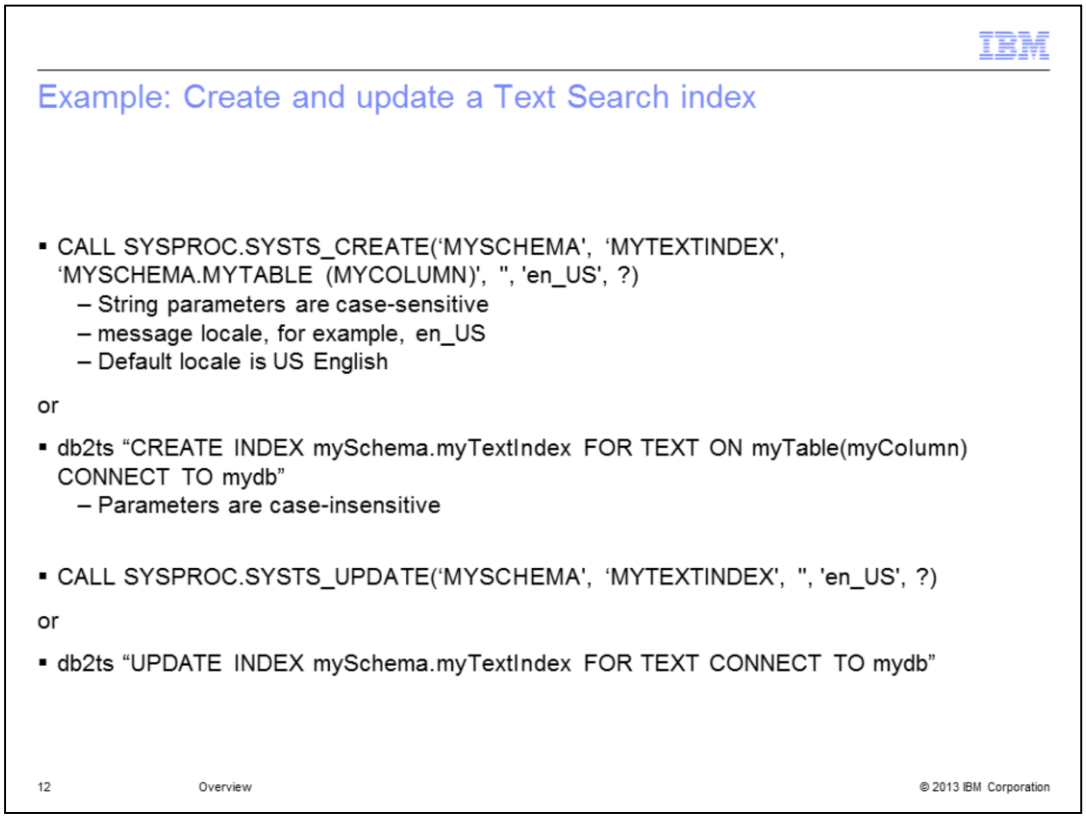
The incremental updates run asynchronously and can be started manually or automatically through a schedule; that is, the text index update is not in the same transaction as the data update. Instead, with the default setup, for each data update an entry is inserted by way of a trigger into a staging table. This staging table is then used to control the incremental update and to synchronize the text index with the data column in the base table.



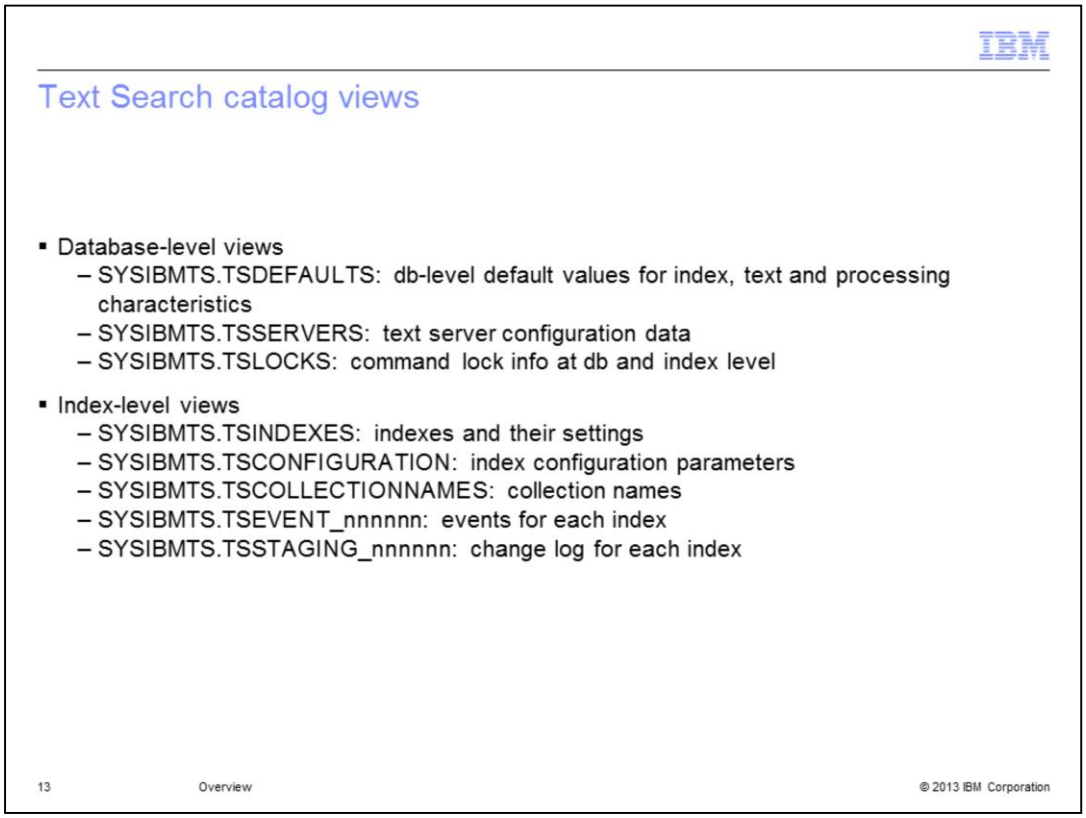
This slide clearly explains the concepts of initial and incremental update index. As displayed on this slide, once the index is created, it is empty, and no data is yet available. An empty collection is created by the Text Search server.

It is the initial index update which adds data about all the documents in the table column to the text index.

The subsequent changes happening on the table are tracked by triggers and captured in a staging table. Later, these updates are applied to an index by way of an incremental update.



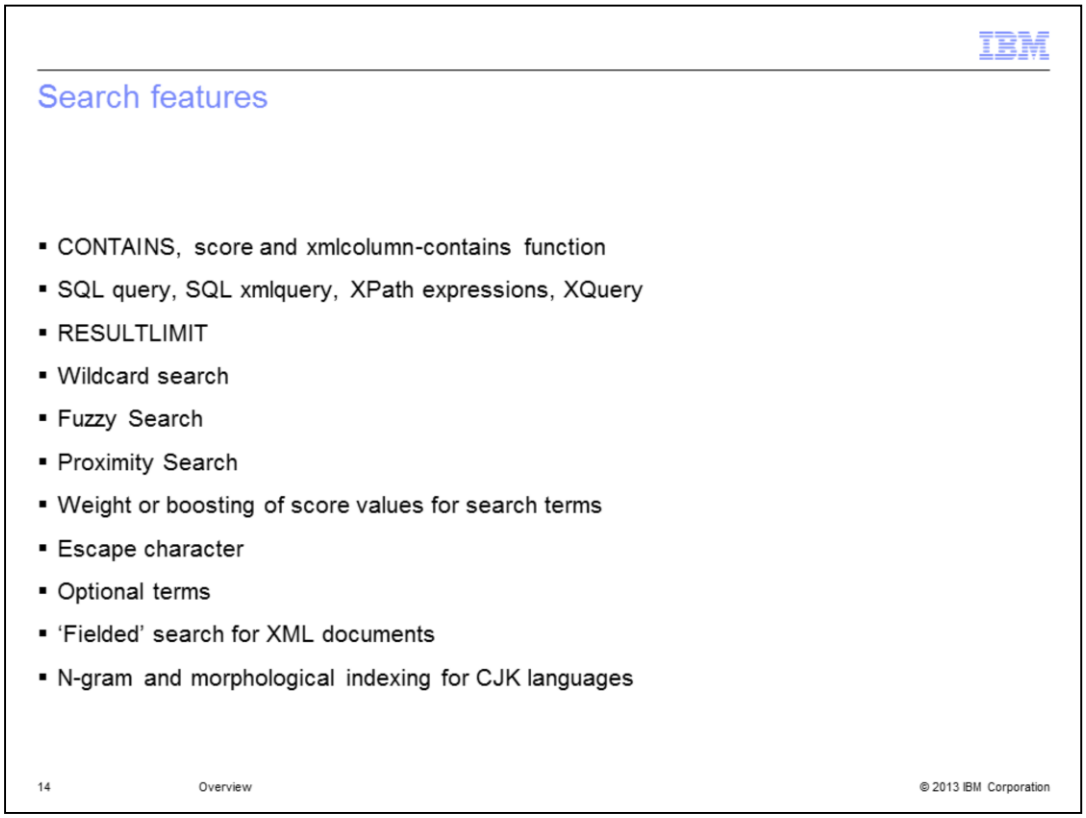
DB2 Text Search provides both command-line interface and stored procedure interface to run administrative operations. This slide provides examples for creating and updating the text index. For a detailed description of options, see the DB2 Text Search User Guide.



This slide displays the catalog views created in the database when it is enabled for Text Search. The Database-level views are SYSIBMTS.TSDEFAULTS, SYSIBMTS.TSSERVERS and SYSIBMTS.TSLOCKS.

SYSIBMTS.TSDEFAULTS shows all the default values for all text indexes in the database. For example, the default values of CODEPAGE, LANGUAGE, FORMAT and so on.

SYSIBMTS.TSSERVERS displays the Text Search server information like host name, port number, token and so on. SYSIBMTS.TSLOCKS shows the command lock information at the database and index level.

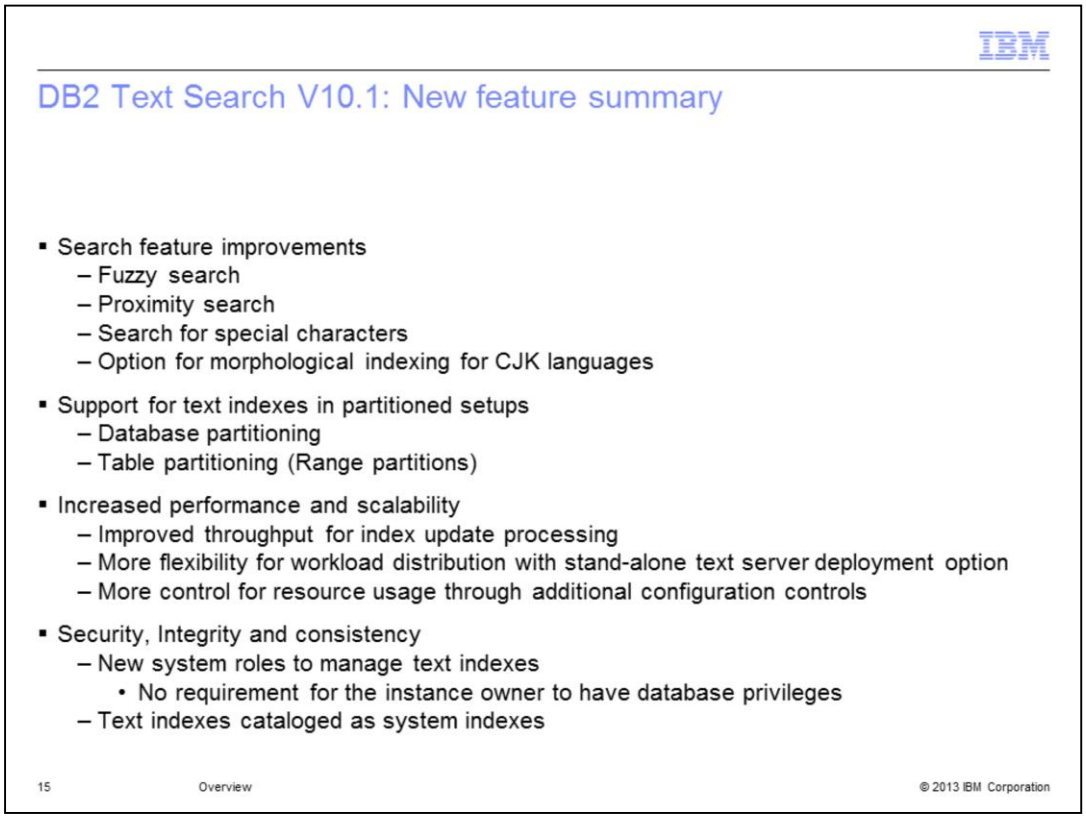


This slide displays the search features available in DB2 Text Search. The DB2 Text Search functions available are CONTAINS, SCORE and xmlcolumn-contains. These are integrated with the DB2 optimizer to select the best possible plan.

Searches are performed by incorporating these functions in SQL query, SQL/XML query, XPath expressions and XQuery statements.

DB2 Text Search also supports wild card search, thus, increasing the size of the result set and decreasing the performance. Fuzzy search can be used where ever misspellings are possible. Proximity search is used to search for the terms located at a specified distance of 'x' words. Weight or boosting of a score value for search terms is possible. Special characters can be searched using the escape character.

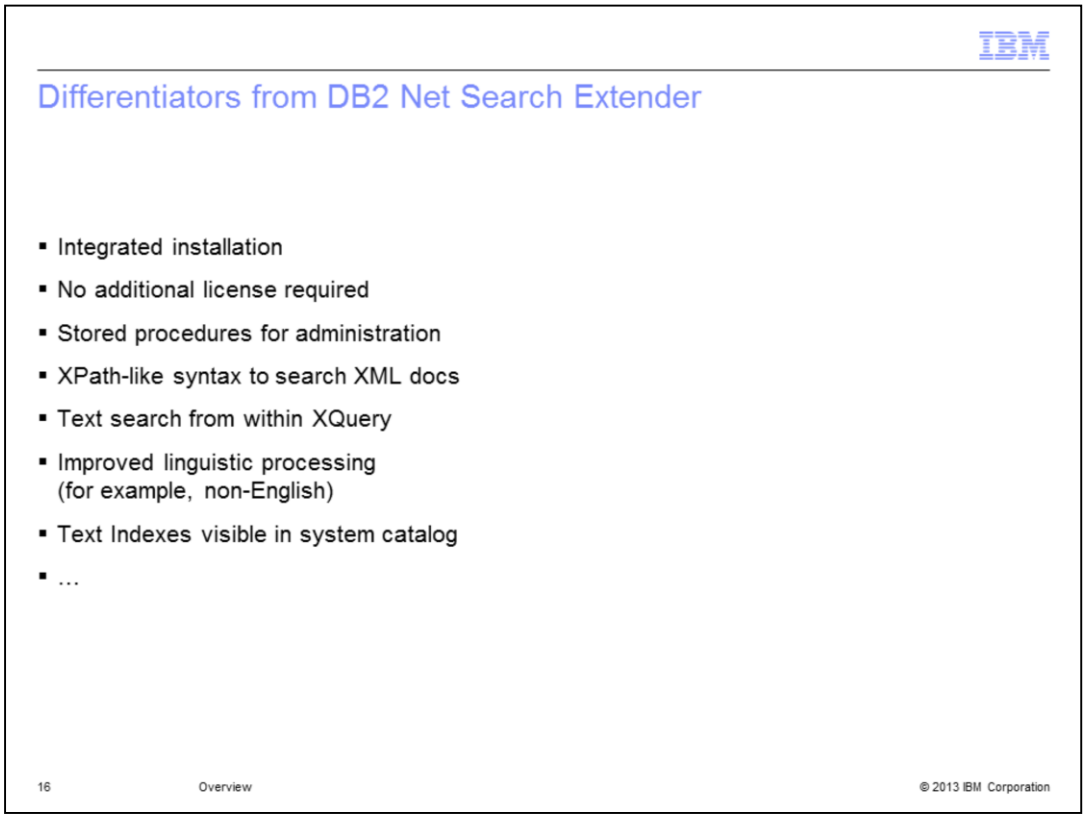
Some of the query terms can be mentioned as 'optional' during the search. All XML fields and attributes can be searched. For CJK languages, it is possible to index them using the n-gram and the morphological option.



This slide lists a summary of new features provided with DB2 Text Search V10.1. There are new search features like Fuzzy search and Proximity search available and improvements are made to the special character search and CJK language indexing.

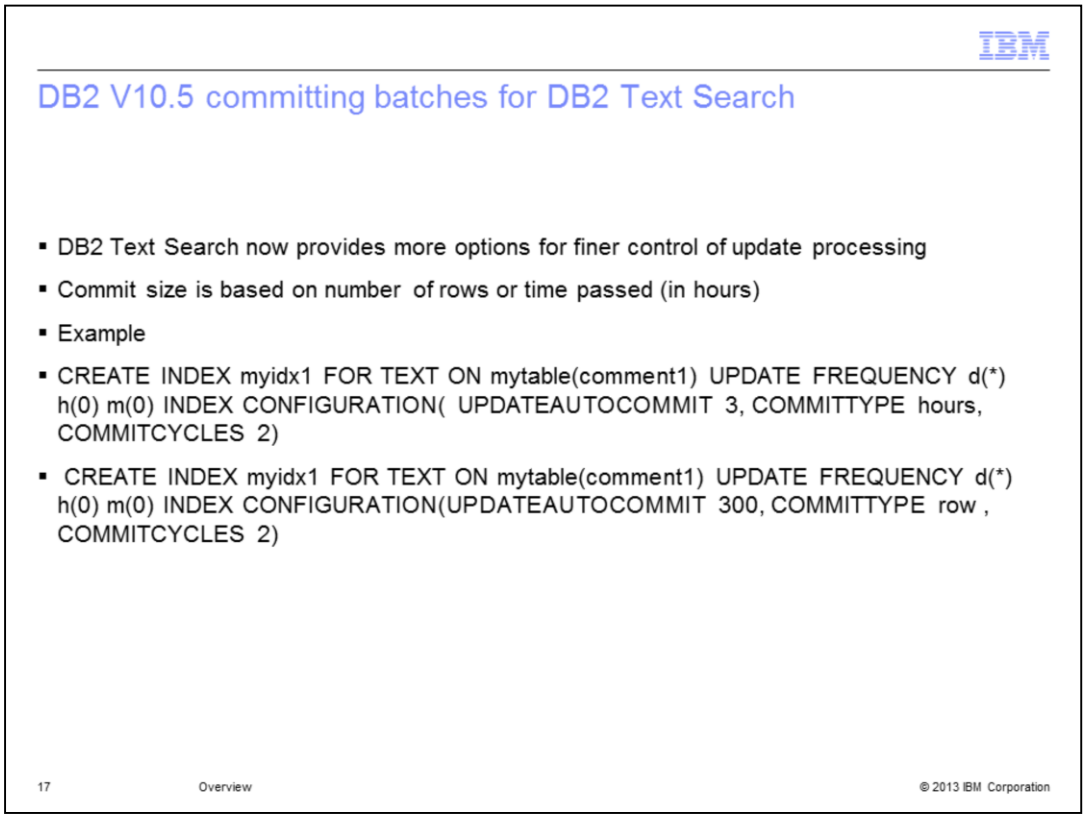
This release also supports indexing on range partitioned tables and database partitioning environment. Due to the integration of the latest Text Server, it offers increased performance and scalability.

A stand-alone server setup is supported now to have flexibility of work load distribution. New system roles have been introduced to manage the text indexes. Whenever a text index is created, an entry is created in DB2 system catalog tables as well, to prevent accidental deletion of a table on which an index is created.

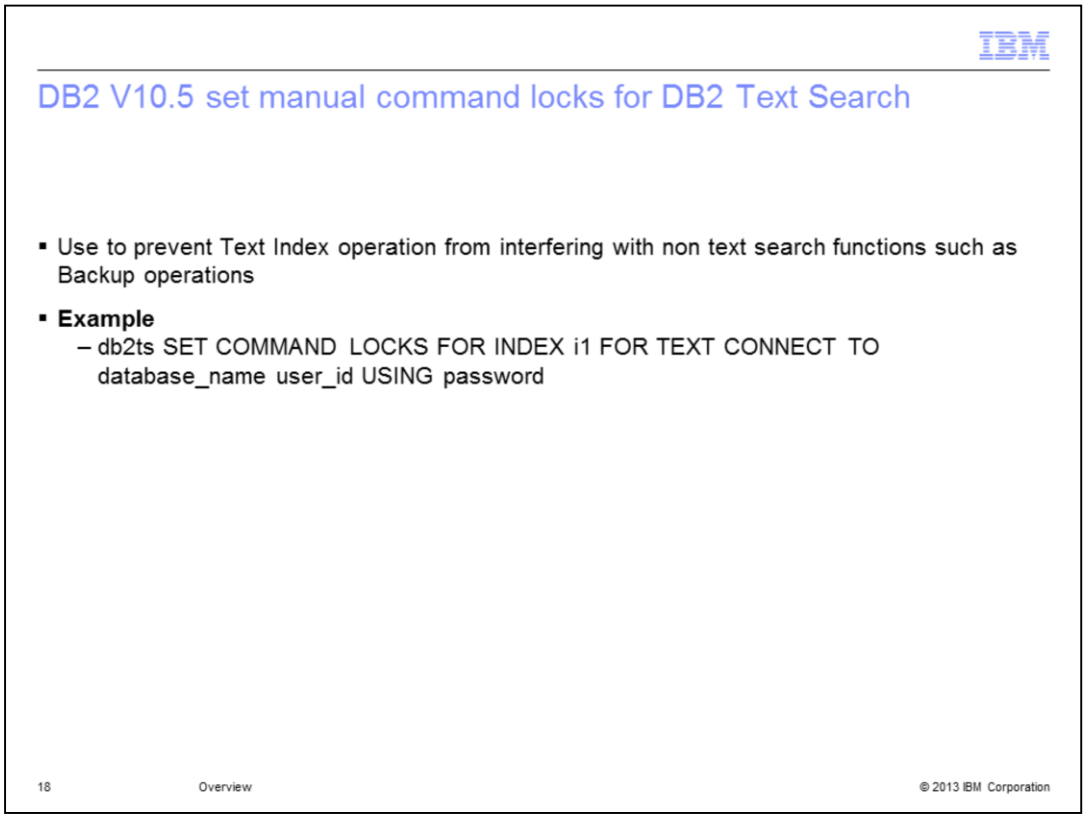


DB2 Text Search is the powerful new generation search engine offering various enhancements compared to its predecessor Net Search Extender.

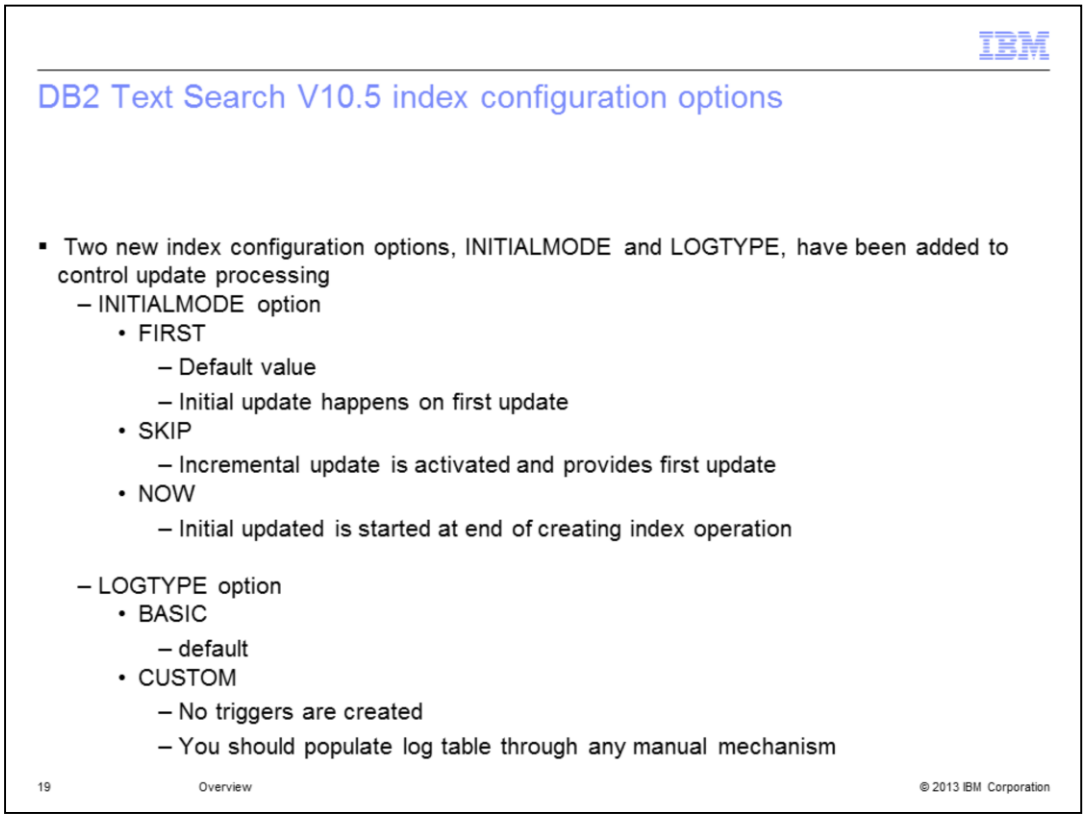
Some of the key differentiators are integrated installation with DB2, no additional license required, stored procedures for administration, and XPath-like syntax to search XML documents. Also, text search can be used from within XQuery, improved linguistic processing for non-English languages, and text indexes are visible in the system catalog to prevent accidental deletion of the base table.



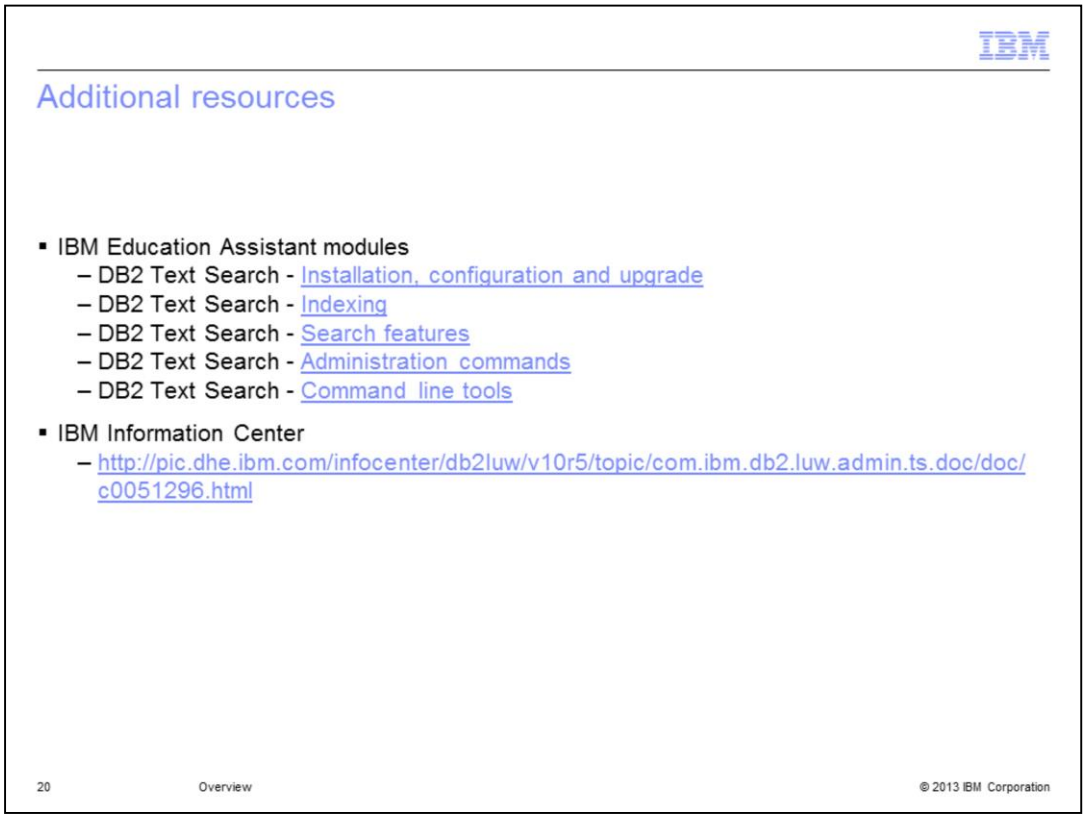
The next few slides discuss new features available with DB2 Text Search V10.5. You can now specify the commit cycle in hours to simplify the specification of commit-batches. There are additional options available now to gracefully end the index update and continue the processing next time the update starts by enabling the use of a maximum time window for updates. Using these options, you can ensure that index processing does not run during peak workload times. The sample examples provided on this slide show how to create an index using the new options available with V10.5.



The command displayed on this slide is used to manually set a lock on an index or on all unlocked indexes in a database. It is used to prevent the text index operation once the lock is applied. Once the lock is applied, it is visible in the SYSIBMTS.TSLOCKS administrative view. An example is displayed on this slide for setting a manual lock on a text index.



A new option, 'INITIALMODE‘, is introduced in this release. With this option, you can run the initial update immediately when the index is created. Also, you can defer it to the first update operation or skip it altogether.
Another option, LOGTYPE‘, is introduced to set a custom log type so that the text index administrator can decide whether triggers should be added to populate the primary log table.



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

Trademarks, disclaimer, and copyright information

Trademarks, disclaimer, and copyright information