CS 491B
Topic Presentation

Full-Text Search
& MS Search Service 2000

Wendy Tan

Emphasis

- Concept of Full-Text Search
- Using MS Search Service

Standard SQL 'like' operator

- Statement: like 'word%'
- Limitations:
  - Fast only if that field has an index
  - Only if the word is at the beginning of the field
  - fields be searched can not be indexed in production environment due to performance, such as "description" field

'like' operator

- like '%word%' is terribly slow
- Why?
  - An index is not used

Full-Text Search

- What is full-text search?
  - Builds an index of every significant word and phrase to perform searchs
- MS Search Service does full-text search

MS Search Service

- Some Facts:
  - Not part of SQL Server
  - Indexes are external to SQL Server
  - Need to set up a full-text search catalog
  - Catalog must be initally populated
  - Catalog must be constantly updated

Full-Text Search Catalogs

- A collection of full-text indexes for a single SQL Server database
- Each catalog may store multiple full-text indexes for multiple tables
- A catalog must belong to a single database
- Each table can only belong to one catalog

Full-Text Search Catalogs

- Typically a single catalog will handle all the full-text searches for a database
- Dedicating a single catalog to a very large table (one with over a million rows) will improve performance

MS Search Service Availability

- MS Search Service included with workstation-class and server-class operating systems
- Possible to install it from the SQL Server professional edition installation
- Can not be installed on any version of Window 9X or Windows XP Home

Creating a Catalog with the Wizard

- Select "Full-Text Indexing" from "Tools" menu of Enterprise Manager
- Specify a unique index to identify the rows indexed with MS Search
- Choose the columns to be full-text indexed

Valid Column Types

- Character data types:
  - char, nchar, varchar, nvarchar, text and ntext
- Image (not covered in this discussion)

Catalog Population

- When a catalog is created it is empty
- To initially populate the catalog, click 'Start Full Population'
- Will take a few seconds, a few minutes or a few hours depending on the amount of data in the indexed columns

Create a Catalog With T-SQL Code

- Not covered in this discussion

Pushing Ongoing Change to a Full-Text Index

- **Incremental populations** – uses a timestamp to pass any rows that have changed since the last population
- **Change Tracking and background population** – update occurs in the background slightly behind the SQL DML transaction

Word Search Features

- Search one word near another word
- Search with wildcards
- Search variations of a word(run,ran etc)
- Weigh importance of word and phrase againt one another
- Fuzzy word/phrase searches

Noise Files

- Exclusion of common words, such as a, the and of.
- MS Search saves them in a noise file
- Query error when search for noise word
- Plain-text file format
- Name: 'noise.enu'

Word Searches

- Two methods
  - **Contains** Function
  - **ContainsTable** Function

Contains Function

Select title
from tablename
*where **contains** (tablename.*, 'Lion')*

- First parameter: column name or * (all columns that indexed)
- Second parameter: word to be searched

The ContainsTable Function

- ContainsTable returns a result set with two columns
  - First column: Key – identify the row
  - Second Column: Rank – compares the row with other rows (range 1-1000)
    - Frequency/uniqueness in the table
    - Frequency/uniqueness in the column

ContainsTable Syntax

```
Select tablename.title, FTS.rank
  from tablename
  join containstable (tablename,*,
  'Lion', 2) FTS
  on tablename.id = FTS.[KEY]
Order by FTS.Rank Desc
```

Advanced Search Options

- Multiple Word Searches
- Searches with wildcards
- Phrase Searches
- Word-Proximity Searches
- Word-Inflection Searches
- Variable-Word-Weight Searches
- Fuzzy Searches

Multiple Word Searches

```
Select title
  from tablename
  where contains(*, 'Tortoise and Hare')
```

- Search through multiple columns
- Words has to be in the same column

Searches with Wildcards

```
Select title
  from tablename
  where contains(*, 'Hunt*')
```

- Wildcards only work at the end of word
- Contains(*, 'He pulled out the thorn*')
  = contains(*, 'He* pulled* out* the* thorn*')

Word-Proximity Searches

```
Select tablename.title, FTS.Rank
  from tablename
     join containstable(table,*, 'life NEAR
  death') FTS
  on tablename.id = FTS.[KEY]
  order by FTS.Rank Desc
```

- Within about 30 words apart

Word-Inflection Searches

Select title
from tablename
where contains(, '**FORMSOF (INFLECTIONAL,** fly')*

- Will look for "flying", "flew" – verb form

Variable-Word-Weight Search

- On a scale of 0.0 to 1.0

Select tablename.title, FTS.Rank
from tablename
join containstable(tablename, column,
*'**isabout**(Lion **weight (.2),** Eagle **weight (.5)',** 20) FTS*
on …… order by …..

Fuzzy Searches

- Exact word searches:
  - **contains, ContainsTable**
- Fuzzy Searches Options:
  - **Freetext, FreetextTable**

Fuzzy Search Example

Select tablename.title, FTS.Rank
from tablename
*join **FREETEXTTABLE***
(tablename., 'The brave hunter kills the lion', 20) FTS*
on tablename.id = FTS.[KEY]
Order by Rank,DESC