Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation for protocols, file formats, languages, standards as well as overviews of the interaction among each of these technologies.

- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you may make copies of it in order to develop implementations of the technologies described in the Open Specifications and may distribute portions of it in your implementations using these technologies or your documentation as necessary to properly document the implementation. You may also distribute in your implementation, with or without modification, any schema, IDL’s, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications.

- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.

- **Patents.** Microsoft has patents that may cover your implementations of the technologies described in the Open Specifications. Neither this notice nor Microsoft’s delivery of the documentation grants any licenses under those or any other Microsoft patents. However, a given Open Specification may be covered by Microsoft Open Specification Promise or the Community Promise. If you would prefer a written license, or if the technologies described in the Open Specifications are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.

- **Trademarks.** The names of companies and products contained in this documentation may be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights.

- **Fictitious Names.** The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

**Reservation of Rights.** All other rights are reserved, and this notice does not grant any rights other than specifically described above, whether by implication, estoppel, or otherwise.

**Tools.** The Open Specifications do not require the use of Microsoft programming tools or programming environments in order for you to develop an implementation. If you have access to Microsoft programming tools and environments you are free to take advantage of them. Certain Open Specifications are intended for use in conjunction with publicly available standard specifications and network programming art, and assumes that the reader either is familiar with the aforementioned material or has immediate access to it.
## Revision Summary

<table>
<thead>
<tr>
<th>Date</th>
<th>Revision History</th>
<th>Revision Class</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>11/06/2009</td>
<td>0.1</td>
<td>Major</td>
<td>Initial Availability</td>
</tr>
<tr>
<td>02/19/2010</td>
<td>1.0</td>
<td>Major</td>
<td>Updated and revised the technical content</td>
</tr>
<tr>
<td>03/31/2010</td>
<td>1.01</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>04/30/2010</td>
<td>1.02</td>
<td>Minor</td>
<td>Updated the technical content</td>
</tr>
<tr>
<td>06/07/2010</td>
<td>1.03</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>06/29/2010</td>
<td>1.04</td>
<td>Editorial</td>
<td>Changed language and formatting in the technical content.</td>
</tr>
<tr>
<td>07/23/2010</td>
<td>1.04</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>09/27/2010</td>
<td>1.04</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>11/15/2010</td>
<td>1.04</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>12/17/2010</td>
<td>1.05</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content.</td>
</tr>
<tr>
<td>03/18/2011</td>
<td>1.05</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>06/10/2011</td>
<td>1.05</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>01/20/2012</td>
<td>1.05</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>04/11/2012</td>
<td>1.05</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>07/16/2012</td>
<td>1.05</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
</tbody>
</table>
# Table of Contents

1 Introduction ............................................................................................................. 10  
  1.1 Glossary ................................................................................................................ 10  
  1.2 References ............................................................................................................. 11  
    1.2.1 Normative References ..................................................................................... 11  
    1.2.2 Informative References .................................................................................. 12  
  1.3 Structure Overview (Synopsis) ............................................................................ 12  
    1.3.1 Index Schema Abstract Data Model ................................................................. 13  
    1.3.2 Index Schema Abstract Data Model Classes .................................................. 14  
      1.3.2.1 ManagedProperty ....................................................................................... 14  
      1.3.2.2 FullTextIndex ............................................................................................ 15  
      1.3.2.3 RefinerConfiguration ............................................................................... 15  
      1.3.2.4 RankProfile ............................................................................................... 16  
      1.3.2.5 FullTextIndexRank .................................................................................... 18  
      1.3.2.6 ImportanceLevel ....................................................................................... 18  
      1.3.2.7 ManagedPropertyBoostComponent ......................................................... 18  
  1.4 Relationship to Protocols and Other Structures ................................................. 18  
  1.5 Applicability Statement ....................................................................................... 19  
  1.6 Versioning and Localization ............................................................................... 19  
  1.7 Vendor-Extensible Fields ..................................................................................... 19  

2 Structures .............................................................................................................. 20  
  2.1 Common Concepts and Type Definitions ......................................................... 22  
    2.1.1 Data Type Definition and Maps ..................................................................... 22  
    2.1.2 Index Field Prefix Naming Conventions ....................................................... 23  
    2.1.3 Document Summary Types ........................................................................... 24  
    2.1.4 Managed Properties ....................................................................................... 24  
    2.1.5 Internal Properties ......................................................................................... 24  
  2.2 maptransform.xml .............................................................................................. 25  
    2.2.1 Global Elements ............................................................................................. 25  
      2.2.1.1 transform-specification ............................................................................ 25  
    2.2.2 Global Attributes ............................................................................................ 25  
    2.2.3 Complex Types .............................................................................................. 25  
      2.2.3.1 CT_transform-specification ..................................................................... 25  
      2.2.3.2 CT_datatype-definitions .......................................................................... 26  
      2.2.3.3 CT_datatype ............................................................................................. 26  
      2.2.3.3.1 Required Data Type Definitions ......................................................... 27  
      2.2.3.3.2 Index Schema-Dependent Data Type Definitions ................................. 28  
      2.2.3.4 CT_number-transformations ................................................................... 28  
      2.2.3.5 CT_field .................................................................................................. 29  
  2.2.4 Simple Types .................................................................................................. 29  
    2.2.4.1 ST_offsetbits ............................................................................................. 29  
    2.2.4.2 ST_signbits ................................................................................................. 30  
    2.2.4.3 ST_exponentbits ........................................................................................ 30  
    2.2.4.4 ST_mantissabits ......................................................................................... 30  
    2.2.4.5 ST_exponent ............................................................................................... 30  
    2.2.4.6 ST_decimalplaces ...................................................................................... 31  
    2.2.4.7 ST_toint ..................................................................................................... 31  
  2.3 fieldspec.xml ...................................................................................................... 31  
    2.3.1 Global Elements ............................................................................................. 31  
      2.3.1.1 fieldlist .................................................................................................... 31  

[MS-FSSCFG] — v20120630  
Search Configuration File Format  
Copyright © 2012 Microsoft Corporation.  
Release: July 16, 2012
2.12.4.4 ST_fieldKind ................................................................. 104
2.12.4.5 ST_fieldType ................................................................. 104
2.12.4.6 ST_wildcardAtt ............................................................... 105
2.12.4.7 ST_tokenization_mode .................................................... 105
2.13 Boost Table Files ................................................................. 106
2.13.1 Occurrence Boost Table Files ............................................. 106
2.13.2 Proximity Boost Table Files .............................................. 107
2.13.3 Global Term Frequency Boost Table File ............................... 107
2.14 rankspace.xml ................................................................. 108
2.14.1 Global Elements ............................................................... 108
2.14.1.1 rankspace ................................................................. 108
2.14.2 Global Attributes .............................................................. 108
2.14.3 Complex Types ................................................................. 108
2.14.3.1 CT_rankspace ............................................................... 108
2.14.3.2 CT_ranking ................................................................. 109
2.14.4 Simple Types ................................................................. 109
2.14.4.1 ST_description ............................................................ 109
2.14.4.2 ST_alwaysZero ............................................................. 109
2.15 resultspace.xml ................................................................. 110
2.15.1 Global Elements ............................................................... 110
2.15.1.1 resultspace ................................................................. 110
2.15.2 Global Attributes .............................................................. 110
2.15.3 Complex Types ................................................................. 110
2.15.3.1 CT_resultspace ............................................................ 110
2.15.3.2 CT_result-view ........................................................... 111
2.15.3.3 CT_field ................................................................. 111
2.15.4 Simple Types ................................................................. 112
2.15.4.1 ST_index ................................................................. 112
2.15.4.2 ST_name ................................................................. 112
2.15.4.3 ST_type ................................................................. 112
2.16 search_preload ................................................................. 113
2.17 sources.xml ................................................................. 113
2.17.1 XML Content ................................................................. 113
2.18 summary.cf ................................................................. 114
2.18.1 ABNF Grammar .............................................................. 114
2.18.2 Configuration Parameter Reference .................................... 115
2.18.3 Summary Classes ............................................................ 115
2.19 summary.map ................................................................. 115
2.20 summaryclasses.xml .......................................................... 116
2.20.1 Global Elements ............................................................. 116
2.20.1.1 summary-input-classes ................................................ 116
2.20.2 Global Attributes ............................................................ 116
2.20.3 Complex Types ............................................................... 116
2.20.3.1 CT_summary-input-classes ........................................... 116
2.20.3.2 CT_summaryClass ....................................................... 117
2.20.3.3 CT_summaryField ....................................................... 117
2.20.4 Simple Types ............................................................... 118
2.20.4.1 ST_classType ............................................................ 118
2.20.4.2 ST_className ............................................................ 118
2.20.4.3 ST_summaryType ......................................................... 118
2.20.4.4 ST_compression ......................................................... 119
2.21 ManagedPropertyBoosts.xml .................................................. 119
2.21.1 Global Elements ............................................................. 119

[MS-FSSCFG] — v20120630
Search Configuration File Format

Copyright © 2012 Microsoft Corporation.

Release: July 16, 2012
<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.21.1.1</td>
<td>field-boosts</td>
<td>119</td>
</tr>
<tr>
<td>2.21.2</td>
<td>Global Attributes</td>
<td>120</td>
</tr>
<tr>
<td>2.21.3</td>
<td>Complex Types</td>
<td>120</td>
</tr>
<tr>
<td>2.21.3.1</td>
<td>CT_FieldBoosts</td>
<td>120</td>
</tr>
<tr>
<td>2.21.3.2</td>
<td>CT_RankProfile</td>
<td>120</td>
</tr>
<tr>
<td>2.21.3.3</td>
<td>CT_BoostGroup</td>
<td>121</td>
</tr>
<tr>
<td>2.21.3.4</td>
<td>CT_FieldBoost</td>
<td>121</td>
</tr>
<tr>
<td>2.21.4</td>
<td>Simple Types</td>
<td>121</td>
</tr>
<tr>
<td>2.21.4.1</td>
<td>ST_RankProfileIndex</td>
<td>121</td>
</tr>
<tr>
<td>2.22</td>
<td>finderc</td>
<td>122</td>
</tr>
<tr>
<td>2.23</td>
<td>template.rc</td>
<td>123</td>
</tr>
<tr>
<td>2.24</td>
<td>errortempl</td>
<td>123</td>
</tr>
<tr>
<td>2.25</td>
<td>footertempl</td>
<td>123</td>
</tr>
<tr>
<td>2.26</td>
<td>headertempl</td>
<td>124</td>
</tr>
<tr>
<td>2.27</td>
<td>nexttempl</td>
<td>124</td>
</tr>
<tr>
<td>2.28</td>
<td>nohitstempl</td>
<td>124</td>
</tr>
<tr>
<td>2.29</td>
<td>prevtempl</td>
<td>124</td>
</tr>
<tr>
<td>2.30</td>
<td>resulttempl</td>
<td>125</td>
</tr>
<tr>
<td>2.31</td>
<td>templates.rc</td>
<td>125</td>
</tr>
<tr>
<td>3</td>
<td>Structure Examples</td>
<td>126</td>
</tr>
<tr>
<td>3.1</td>
<td>maptransform.xml</td>
<td>126</td>
</tr>
<tr>
<td>3.2</td>
<td>fieldspec.xml</td>
<td>127</td>
</tr>
<tr>
<td>3.3</td>
<td>configuration.attributes.xml</td>
<td>127</td>
</tr>
<tr>
<td>3.4</td>
<td>fsearch.addon</td>
<td>129</td>
</tr>
<tr>
<td>3.5</td>
<td>indexConfig.xml</td>
<td>130</td>
</tr>
<tr>
<td>3.6</td>
<td>index.cf</td>
<td>134</td>
</tr>
<tr>
<td>3.7</td>
<td>fixml_mappings.xml</td>
<td>137</td>
</tr>
<tr>
<td>3.8</td>
<td>rank.cf</td>
<td>139</td>
</tr>
<tr>
<td>3.9</td>
<td>fieldProperties.xml</td>
<td>140</td>
</tr>
<tr>
<td>3.10</td>
<td>Boost Table Files</td>
<td>141</td>
</tr>
<tr>
<td>3.11</td>
<td>rankspace.xml</td>
<td>142</td>
</tr>
<tr>
<td>3.12</td>
<td>resultspace.xml</td>
<td>143</td>
</tr>
<tr>
<td>3.13</td>
<td>summary.cf</td>
<td>143</td>
</tr>
<tr>
<td>3.14</td>
<td>summaryclasses.xml</td>
<td>144</td>
</tr>
<tr>
<td>4</td>
<td>Security Considerations</td>
<td>145</td>
</tr>
<tr>
<td>5</td>
<td>Appendix A: Full XML Schemas</td>
<td>146</td>
</tr>
<tr>
<td>5.1</td>
<td>maptransform.xsd</td>
<td>146</td>
</tr>
<tr>
<td>5.2</td>
<td>fieldspec.xsd</td>
<td>147</td>
</tr>
<tr>
<td>5.3</td>
<td>configuration.attributes.xsd</td>
<td>148</td>
</tr>
<tr>
<td>5.4</td>
<td>indexConfig.xsd</td>
<td>152</td>
</tr>
<tr>
<td>5.5</td>
<td>fixml_mappings.xsd</td>
<td>159</td>
</tr>
<tr>
<td>5.6</td>
<td>fieldProperties.xsd</td>
<td>160</td>
</tr>
<tr>
<td>5.7</td>
<td>rankspace.xsd</td>
<td>162</td>
</tr>
<tr>
<td>5.8</td>
<td>resultspace.xsd</td>
<td>163</td>
</tr>
<tr>
<td>5.9</td>
<td>summaryclasses.xsd</td>
<td>164</td>
</tr>
<tr>
<td>5.10</td>
<td>ManagedPropertyBoosts.xsd</td>
<td>165</td>
</tr>
<tr>
<td>6</td>
<td>Appendix B: Product Behavior</td>
<td>167</td>
</tr>
<tr>
<td>7</td>
<td>Change Tracking</td>
<td>168</td>
</tr>
</tbody>
</table>
1 Introduction

This document specifies the Search Configuration File Format, including file naming conventions and file formats for the configuration files that derive from the index configuration for a search service application.

Sections 1.7 and 2 of this specification are normative and can contain the terms MAY, SHOULD, MUST, MUST NOT, and SHOULD NOT as defined in RFC 2119. All other sections and examples in this specification are informative.

1.1 Glossary

The following terms are defined in [MS-GLOS]:

- Augmented Backus-Naur Form (ABNF)
- UTF-8
- XML

The following terms are defined in [MS-OFCGLOS]:

- associated query
- attribute vector
- authority rank
- boundary match
- content collection
- context boost
- context catalog
- context dictionary
- datetime
- deep refinement
- default index
- document summary
- document vector
- drilling
- dynamic rank
- dynamic teaser
- equivalence class
- external occurrence boost
- fallback managed property
- FAST Index Markup Language (FIXML)
- field importance level
- field prefix
- freshness boost
- full-text index context
- full-text index field
- hit highlighted summary
- index alias
- index field
- index schema
- input summary class
- internal property
- item
- item processing
- keyword rank
- latent attribute vector
managed property
normalized occurrence boost
occurrence boost
output summary class
phrase break
position index
property context
property index
proximity boost
proximity search
quality rank
query refinement
rank
rank profile
recall
refinement bin
refinement modifier
refiner
search application
search index
search query
shallow refinement
static rank
stemming
substring search
summary class
synthetic context catalog
token
tokenization
XML attribute
XML schema definition (XSD)

The following terms are specific to this document:

**MAY, SHOULD, MUST, SHOULD NOT, MUST NOT:** These terms (in all caps) are used as described in [RFC2119]. All statements of optional behavior use either MAY, SHOULD, or SHOULD NOT.

### 1.2 References

References to Microsoft Open Specifications documentation do not include a publishing year because links are to the latest version of the technical documents, which are updated frequently. References to other documents include a publishing year when one is available.

#### 1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information. Please check the archive site, http://msdn2.microsoft.com/en-us/library/E4BD6494-06AD-4aed-9823-445E921C9624, as an additional source.


1.2.2 Informative References


[MS-OFCGLOS] Microsoft Corporation, "Microsoft Office Master Glossary".

1.3 Structure Overview (Synopsis)

This document describes an index schema configuration file set that is fully or partly derived from the index schema and is used by the indexing service, query matching service, item processing service, and query processing service in a search service application.

The configuration file set describes configuration parameters for item preprocessing prior to indexing, item indexing, query evaluation, and query/result processing. It contains both static configuration parameters and configuration parameters derived from the index schema. Static configuration parameters are required for the components to operate correctly. The components download these files from the configuration service. The configuration parameters derived from the index schema represent configuration that depends on the configuration of the search application for configuring search-related features and managed property settings.

The index schema consists of the following main configuration entities:

- **Managed property configuration**: This describes which properties of an item, as derived from the content source, are indexed with associated indexing configuration entities.

- **Full-text index field configuration**: This describes how to apply full-text queries against a particular set of managed properties.

- **Rank profile configuration**: This describes how to achieve a result set that is sorted by rank.

- **Refiner configuration**: This describes how query results can return statistical information about managed properties.
1.3.1 Index Schema Abstract Data Model

The following figure provides an abstract data model of the index schema.

Figure 1: Abstract data model of the index schema

This model describes a generic index schema for a search application that contains all the entities required to generate the configuration files described in this specification.
1.3.2 Index Schema Abstract Data Model Classes

1.3.2.1 ManagedProperty

A ManagedProperty class represents one managed property within the index schema. A managed property can be associated with a refiner configuration, as described in section 1.3.2.3.

A managed property is associated with one or more full-text index fields, as described in section 1.3.2.2. The following table describes the members of a ManagedProperty class.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name of the managed property.</td>
</tr>
<tr>
<td>Type</td>
<td>The data type for the managed property. The following data types are supported:</td>
</tr>
<tr>
<td></td>
<td><strong>Text:</strong> UTF-8 text data type for text search.</td>
</tr>
<tr>
<td></td>
<td><strong>Integer:</strong> 64-bit signed integer.</td>
</tr>
<tr>
<td></td>
<td><strong>Decimal:</strong> Fixed-point signed decimal data type.</td>
</tr>
<tr>
<td></td>
<td>The number of digits for the decimal precision is configurable.</td>
</tr>
<tr>
<td></td>
<td><strong>Float:</strong> 64-bit floating-point data type that uses base 2 for the exponent. The exponent uses 11 bits, and the mantissa uses 52 bits.</td>
</tr>
<tr>
<td></td>
<td><strong>Datetime:</strong> datetime data type. This data type is represented as a 64-bit unsigned integer in the search index and supports sorting and query refinement, as does the Integer data type.</td>
</tr>
<tr>
<td></td>
<td><strong>Boolean:</strong> Boolean data type with valid values true and false.</td>
</tr>
<tr>
<td>DecimalPrecision</td>
<td>The number of decimal positions for decimal data type.</td>
</tr>
<tr>
<td>Queryable</td>
<td>Describes whether the managed property can be queried as an individual property. Even if the Queryable member contains a value of &quot;no&quot;, the managed property can be included in a full-text index field.</td>
</tr>
<tr>
<td>SortableType</td>
<td>The full-text sort configuration for the managed property. The values are as follows:</td>
</tr>
<tr>
<td></td>
<td><strong>Disabled:</strong> Full-text sorting is not supported.</td>
</tr>
<tr>
<td></td>
<td><strong>Enabled:</strong> Full-text sorting is enabled and activated in the search index.</td>
</tr>
<tr>
<td></td>
<td><strong>Latent:</strong> Full-text sorting is enabled in the index file structures as a latent attribute vector. It is not activated in the search index. The SortableType member for the managed property can be set to &quot;Enabled&quot; without re-indexing the items.</td>
</tr>
<tr>
<td>StemmingEnabled</td>
<td>Describes whether stemming is supported for this managed property.</td>
</tr>
<tr>
<td>MaxIndexSize</td>
<td>Describes the maximum number of kilobytes of data from the managed property within an item that will be included in the search index.</td>
</tr>
<tr>
<td>MaxResultSize</td>
<td>Describes the maximum number of kilobytes that a document summary can contain.</td>
</tr>
<tr>
<td>SummaryType</td>
<td>Describes the document summary type for this managed property. The values are as follows:</td>
</tr>
<tr>
<td></td>
<td><strong>Disabled:</strong> Document summaries are not supported for this managed property.</td>
</tr>
<tr>
<td></td>
<td><strong>Static:</strong> The document summary is a textual representation of the managed property.</td>
</tr>
</tbody>
</table>
### Name  Description

**Dynamic:** The document summary is a *hit highlighted summary* of the managed property.

**ResultFallBack**
Describes a *fallback managed property* for a managed property when the **SummaryType** member contains the value "Dynamic". If a hit highlighted summary cannot be created for a query, the document summary associated with the fallback managed property is used instead.

**SubstringEnabled**
Describes whether *substring search* is supported for this managed property.

**MergeCrawledProperties**
Support for multiple string values in a managed property.

**DeleteDisallowed**
A Boolean value that indicates whether a managed property can be deleted. If set, this is a mandatory managed property.

#### 1.3.2.2 FullTextIndex

A **FullTextIndex** class represents one full-text index field within the index schema.

A full-text index field is associated with one or more managed properties through a field *importance level*. The following table describes the members of a **FullTextIndex** class.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
<td>The name of the full-text index field.</td>
</tr>
<tr>
<td><strong>IsDefault</strong></td>
<td>Describes whether this full-text index field is the <em>default index</em>.</td>
</tr>
<tr>
<td><strong>StemmingEnabled</strong></td>
<td>Describes whether stemming is supported.</td>
</tr>
</tbody>
</table>

#### 1.3.2.3 RefinerConfiguration

A **RefinerConfiguration** class represents the configuration of a refiner associated with a managed property. The following table describes the members of a **RefinerConfiguration** class.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| **RefinementType** | **DeepRefinementEnabled:** This refiner supports deep refinement.  
**DeepRefinementDisabled:** This refiner supports shallow refinement.  
**LatentRefinement:** Refinement is enabled in the index file structures as a latent attribute vector. It is not activated in the running index. **RefinementType** can later be changed to "DeepRefinementEnabled" without re-indexing the items. |
| **Divisor**     | The divisor that is used to scale down refinement values before displaying them to the user. For example, if the actual values are in bytes and the conversion unit is kilobytes, use **Divisor** set to 1024. |
| **Intervals**   | The maximum number of refinement bins to generate.                          |
| **Resolution**  | The resolution of the returned refinement bin. A resolution of 100 implies that the boundaries of the refinement bin are a multiple of 100 nanoseconds. |
| **Algorithm**   | The numeric refiner discretization algorithm:                              |
### Refinement Bin Types

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>equalfrequency</td>
<td>The value range of different refinement bins is allowed to have different widths. The widths are calculated such that approximately the same number of observations falls into each refinement bin.</td>
</tr>
<tr>
<td>equalwidth</td>
<td>The value range of each refinement bin is equal. The width is static and is not computed dynamically.</td>
</tr>
<tr>
<td>rangedivision</td>
<td>The value range of each refinement bin is considered to be equal. The width is computed dynamically and is not required to be equal.</td>
</tr>
</tbody>
</table>

### DefaultValue

The default value used for items that have no value for the managed property associated with this refiner.

### CutoffMaxBuckets

The limit for the number of refinement bins to be returned.

### Anchoring

The matching mode for string refinement modifiers. This describes how a drill-down query relates to the actual content of the referenced managed property and the completeness criteria for a match.

If the referenced property is a multi-value property, the criteria apply for individual strings within the property.

The Anchoring member describes the following anchoring modes:

- **auto**: The same as "complete" if boundary match is enabled for the managed property; otherwise, it is the same as "none".
- **none**: The refinement modifiers will not be anchored. This means that the drill-down query will match items that contain the refinement modifier terms, but the matching index field is allowed to contain additional terms before or after the terms.
- **complete**: The refinement modifiers anchor to both the beginning and the end of the index field. This means a complete match between refinement modifier and index field of the matching item.
- **prefix**: The refinement modifiers are anchored to the beginning of the index field. This means that the matching index field begins with the refinement modifier terms.
- **suffix**: The refinement modifiers are anchored to the end of the index field. This means that the matching index field ends with the refinement modifier terms.

### 1.3.2.4 RankProfile

A RankProfile class represents the configuration of a particular rank profile. A RankProfile class defines how relevance ranking of a query result is performed. This definition includes the following:

- How dynamic rank evaluation is accomplished
- If and how freshness boost is enabled
- Which set of static rank values is being used
- The relative importance of the rank components within the overall rank computation

A RankProfile class is associated with one or more full-text indexes for rank evaluation of full-text queries. In the Unified Modeling Language (UML) diagram, this is modeled through the FullTextIndexRank class. For more information about FullTextIndexRank class, see section 1.3.2.5.

A RankProfile class can be associated with one or more managed properties for quality rank evaluation. Each managed property is associated with the full-text index field through a weight that describes the relative weight of this managed property in the overall quality rank computation.
A **RankProfile** class can be associated with one or more keyword rank components which increases/decreases the rank of items in the result set which contain keywords. In the Unified Modeling Language (UML) diagram, this is modeled through the **ManagedPropertyBoostComponent** class. For more on information about the **ManagedPropertyBoostComponent** class, see section 1.3.2.7.

The following table describes the members of a **RankProfile** class.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name of the rank profile.</td>
</tr>
<tr>
<td>IsDefault</td>
<td>A Boolean value that specifies whether this is the default rank profile.</td>
</tr>
<tr>
<td>StopWordThreshold</td>
<td>If <strong>StopWordThreshold</strong> is $X$, any search term that occurs in fewer than $X$ documents on a search node will always be fully ranked on that search node. If the search term occurs in more than $X$ documents on a search node, implementation-specific rank optimizations imply that not all matching documents will achieve dynamic rank for a query.</td>
</tr>
<tr>
<td>PositionStopWordThreshold</td>
<td>This member controls whether a search term will contribute to the proximity component of the rank score for a particular query. If $D$ is the number of documents matching the search term on a particular search node, and $O$ is the total number of occurrences of the search term across the $D$ documents, and $X$ is the <strong>PositionStopWordThreshold</strong> value, any search term that has $D + O$ less than $X$ will always be taken into account when proximity boost is calculated on this search node. If the search term has a value $D + O$ greater than $X$, position information will not be retrieved for that term on this search node. Therefore, the search term will not be taken into account when proximity boost is calculated on this search node.</td>
</tr>
<tr>
<td>QualityWeight</td>
<td>The relevance coefficient for the quality rank component.</td>
</tr>
<tr>
<td>AuthorityWeight</td>
<td>The relevance coefficient for the <strong>authority rank</strong> component.</td>
</tr>
<tr>
<td>QueryAuthorityWeight</td>
<td>The relevance coefficient for the query authority rank component.</td>
</tr>
<tr>
<td>FreshnessWeight</td>
<td>The relevance coefficient for the freshness rank component.</td>
</tr>
</tbody>
</table>
| FreshnessResolution  | The resolution for calculating freshness boost. Resolution set to "Hour" indicates that documents with a time stamp within the same hour will get the same freshness rank boost. Valid values are as follows:  
   - Second
   - Minute
   - Hour
   - Day
   - Year  |
| RankModelName        | A rank model is an implementation-specific profile that controls the detailed rank parameters. The default rank model is named "default".             |
1.3.2.5 FullTextIndexRank

A FullTextIndexRank class represents the configuration of proximity boost and **context boost** weight parameters associated with a particular full-text index field for a rank profile.

This class is associated with one or more ImportanceLevel classes, which represent the relevance weight of each field importance level.

The following table describes the members of the FullTextIndexRank class.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ProximityWeight</td>
<td>The relevance coefficient for the proximity boost component related to this full-text index field for the associated rank profile.</td>
</tr>
<tr>
<td>ContextWeight</td>
<td>The relevance coefficient for the context boost component related to this full-text index field for the associated rank profile.</td>
</tr>
</tbody>
</table>

1.3.2.6 ImportanceLevel

An ImportanceLevel class represents the relevance weight for the field importance level within a full-text index field for a rank profile. The following table describes the members of the ImportanceLevel class.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Level</td>
<td>The field importance level number.</td>
</tr>
<tr>
<td>Weight</td>
<td>The relevance coefficient for the context boost component associated with this field importance level.</td>
</tr>
</tbody>
</table>

1.3.2.7 ManagedPropertyBoostComponent

A ManagedPropertyBoostComponent class represents a specification of keywords that increases/decreases the rank of an item if they occur in the specified managed property in that item. This does not change the recall of a query.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ManagedPropertyName</td>
<td>Name of managed property this keyword rank boost applies to. Only items with keywords in the specific managed property get increased/decreased rank.</td>
</tr>
<tr>
<td>BoostValue</td>
<td>One or more concatenated boost value(s) of the keyword rank component. A boost value is formatted as &quot;&lt;boost term&gt;, &lt;boost amount&gt;&quot;. For example, &quot;Microsoft, 200&quot;. Multiple boost values can be given at once. For example, &quot;Microsoft, 200, Word, 3000&quot;.</td>
</tr>
</tbody>
</table>

1.4 Relationship to Protocols and Other Structures

The following figure provides a high-level overview of the services and protocols associated with the configuration files described in this document.
Figure 2: Service and protocol relationship

The following protocols and services use the configuration files described in this document:

- The Distributed Query Execution Protocol, as described in [MS-FSDQE].
- The item processing service performs item pre-processing prior to indexing.
- The indexing service performs content indexing.
- The query matching service performs distributed query execution against the index.
- The query processing service performs query pre-processing and result post-processing.

All configuration files are stored on the configuration service and downloaded through the protocol described in [MS-FSCX].

For more information about the services and protocols listed earlier, see [MS-FSO].

1.5 Applicability Statement

The file format structures described in this document apply to full-text search applications.

1.6 Versioning and Localization

None.

1.7 Vendor-Extensible Fields

None.
2 Structures

In the following sections, the schema definition might differ from the processing rules imposed by the protocol. The XSD in this specification provides a base description of the file format. The text that introduces the XSD specifies additional restrictions that reflect protocol behavior. For example, the schema definition might allow for an element to be empty, null, or not present but the behavior of the protocol as specified restricts the same elements to being non-empty, present, and not null.

A configuration file set for an index schema MUST consist of all the configuration files specified in this section, and MUST be based on the same set of managed properties as specified in [MS-FSAS], section 3.9.

The specified file path MUST be used when the Configuration XML-RPC Protocol is requesting the file. For more details, see [MS-FSCX] section 2.2.26.

The structures in this document are stored and transferred through [MS-FSCX]. Protocol clients of [MS-FSCX] that require notifications when these structures have changed MUST register with the protocol server of [MS-FSCX] by using the RegisterModule method as specified in [MS-FSCX] section 3.1.4.29, and by using the following elements as specified in [MS-FSCX] section 2.2.1.3:

- **port**: An integer that contains the port number where the protocol client listens to XML-RPC requests.
- **type**: A string that contains the name of the component requesting the notification.
- **version**: A string whose value is implementation specific.
- **name**: A string that contains the name of the component requesting the notification.
- **alerts**: A struct of AlertType data type, as specified in [MS-FSCX] section 2.2.1.4, that contains the "configfile" value.

The name of the component that is requesting the notification MUST be one of the following values as specified in [MS-FSCX] section 2.2.1.7:

- **ProcessorServer**: The item processing component.
- **Search Dispatcher**: The query processing component.
- **Search Engine**: The indexing and query matching component.

The specification of each configuration file contains a section that begins with the following table. The table specifies the high-level requirements for the configuration file.

<table>
<thead>
<tr>
<th>Table row name</th>
<th>Table row meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Middleware Protocol storage path</td>
<td>The specified file path MUST be used when:</td>
</tr>
<tr>
<td></td>
<td>The protocol server as specified in [MS-FSAS] is storing the configuration file in the configuration service upon a change of index schema that affects any configuration parameter in the file.</td>
</tr>
<tr>
<td></td>
<td>A component that is using the file is requesting a download of the configuration file from the configuration service as specified in [MS-FSCX].</td>
</tr>
<tr>
<td></td>
<td>The specified file path MUST be used when the Configuration XML-RPC Protocol is requesting the file. For more details, see [MS-FSCX] section 3.1.4.24.</td>
</tr>
<tr>
<td>Table row name</td>
<td>Table row meaning</td>
</tr>
<tr>
<td>----------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>Type of data</td>
<td>Configuration information derived from the index schema. Non-configurable protocol-related information. Implementation-specific configuration information.</td>
</tr>
<tr>
<td>File format</td>
<td>This row specifies the configuration file format used and is one of the formats specified in the following table.</td>
</tr>
</tbody>
</table>

The following table specifies the configuration file formats used in this document. The description of the individual files refers to the file formats specified in this table. The files MUST be formatted according to the corresponding file format identifier specified in the table.

<table>
<thead>
<tr>
<th>File format</th>
<th>File syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>XML schema file</td>
<td>An XML-formatted configuration file, which contains name/value pairs as XML attributes. This document provides documentation on the XML file in XML schema definition (XSD) syntax. All XML configuration files MUST be formatted as specified in [XMLSCHEMA] for the particular configuration file. The XML document MUST NOT contain formal references to the XML schema.</td>
</tr>
<tr>
<td>Fixed XML file</td>
<td>An XML-formatted configuration file that contains name/value pairs as XML attributes. The configuration file content is fixed and MUST NOT be changed.</td>
</tr>
<tr>
<td>ABNF text file</td>
<td>A text-formatted configuration file where the syntax is documented through Augmented Backus-Naur Form (ABNF) grammar, as specified in [RFC5234].</td>
</tr>
</tbody>
</table>
| Name value text file | Each line contains a name-value pair. The file MUST use the following ABNF syntax:  
  
  file     = 1*(comment / newlines / parameter)  
  parameter = parname SP parvalue newline  
  parname   = 1*(ALPHA / DIGIT)  
  parvalue  = 1*VCHAR  
  comment   = "#" *VCHAR newline  
  newlines  = 1*newline  
  newline   = *SP crlf  
  crlf      = LF / (CR LF)  

  **parname:** Specifies the name of the configuration parameter.  
  **parvalue:** Specifies the value of the configuration parameter.  
  Lines that begin with a number sign (#) contain comments. Empty lines are allowed and MUST be discarded. |
| Name: value text file | Each line contains a name-value pair. The file MUST use the following ABNF syntax:  
  
  file     = 1*(comment / newlines / parameter)  
  parameter = parname ":" SP parvalue newline  
  parname   = 1*(ALPHA / DIGIT)  
  parvalue  = 1*VCHAR  
  comment   = "#" *VCHAR newline  
  newlines  = 1*newline  
  newline   = *SP crlf  
  crlf      = LF / (CR LF)  


2.1 Common Concepts and Type Definitions

This section specifies index schema concepts, data types, and naming definitions used in the subsequent sections that specify the individual configuration file structures.

2.1.1 Data Type Definition and Maps

The following table specifies the supported managed property data types used in the configuration files specified in this document. The first column specifies the corresponding data types in the abstract data model for the index schema, as described in section 1.3.2.1. The second column specifies the data types used in section 2 of this document.

<table>
<thead>
<tr>
<th>Data type for schema abstract data model</th>
<th>Data type used in configuration files specified in this document</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Text</td>
<td>String</td>
<td>A data type for UTF-8 text, as specified in [RFC3629].</td>
</tr>
<tr>
<td>Integer</td>
<td>INT</td>
<td>A 64-bit signed integer.</td>
</tr>
<tr>
<td>Decimal</td>
<td>DECIMAL_[N]&lt;dp&gt;</td>
<td>A fixed-point signed decimal data type where N is a digit that specifies the number of digits for the decimal precision. A DECIMAL data type is specified for each unique decimal precision specified for managed properties within the index schema. For example, DECIMAL_2 represents a decimal</td>
</tr>
<tr>
<td>Data type for schema abstract data model</td>
<td>Data type used in configuration files specified in this document</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------</td>
<td>---------------------------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>DECIMAL_NAV</td>
<td>An internal data type used for the DECIMAL data type in attribute vectors.</td>
<td></td>
</tr>
<tr>
<td>Float</td>
<td>FLOAT2B</td>
<td>A 64-bit floating-point data type that uses base 2 for the exponent. The exponent uses 11 bits, and the mantissa uses 52 bits.</td>
</tr>
<tr>
<td>Datetime</td>
<td>DATETIME</td>
<td>The datetime data type. This data type is represented as a 64-bit unsigned integer in the index and supports sorting and query refinement, as does the Integer data type.</td>
</tr>
<tr>
<td>Boolean</td>
<td>String</td>
<td>A string data type, the only valid values of which are true and false.</td>
</tr>
</tbody>
</table>

### 2.1.2 Index Field Prefix Naming Conventions

For index structures, index schema–related configuration files and specific query operations use different name representations of index fields in the index. Index field naming is based on the following generic syntax.

<prefix><index field name>

In the preceding syntax, prefix is a field prefix that specifies the representation of an index field in the index. The following table specifies the supported values for the field prefix.

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Empty (no prefix)</td>
<td>There is no field prefix. This applies to an internal property within the search index. For more details, see section 2.1.4.</td>
</tr>
<tr>
<td>bsum</td>
<td>A UTF-8–encoded document summary. This is a text representation of a managed property within the index schema, without any formatting.</td>
</tr>
<tr>
<td>bsrc</td>
<td>A UTF-8–encoded text representation of a managed property within the index schema. This representation contains additional markup used when the dynamic teaser object is created.</td>
</tr>
<tr>
<td>batv</td>
<td>An attribute vector representation of the managed property that specifies the sortable representation of the property within the index that is used in sorting-related query operations.</td>
</tr>
<tr>
<td>bavn</td>
<td>An attribute vector representation of the managed property that specifies the query refinement representation of the property. This attribute is used in deep refinement–related query operations.</td>
</tr>
<tr>
<td>bcat</td>
<td>A context catalog that contains one full-text index field; that is, one or more managed properties available for full-text index search.</td>
</tr>
<tr>
<td>bcon</td>
<td>A searchable representation of a managed property within an index.</td>
</tr>
<tr>
<td>Prefix</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>bdlg</td>
<td>The diagnostic debug log for the dynamic teaser object representation of a managed property.</td>
</tr>
</tbody>
</table>

### 2.1.3 Document Summary Types

The following table specifies the valid document summary types returned for query results.

<table>
<thead>
<tr>
<th>Document summary type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>A UTF-8–encoded character string, as specified in [RFC3629]. The length of the string does not exceed 64 kilobytes. The string length is encoded as a 16-bit unsigned integer in the result details returned for a query.</td>
</tr>
<tr>
<td>longstring</td>
<td>A UTF-8–encoded character string, as specified in [RFC3629]. The length of the string can exceed 64 kilobytes. The string length is encoded as a 32-bit unsigned integer in the result details returned for a query. If the most significant bit is set, then compression is enabled, see the first table in section 2.8.3.33.</td>
</tr>
<tr>
<td>data</td>
<td>Used only for internal document summary representation inside the index. This is exposed in configuration files specified in this document, but is not used on any interface.</td>
</tr>
</tbody>
</table>

### 2.1.4 Managed Properties

The following table specifies the managed properties that MUST appear in a search index. This table corresponds to managed properties with `DeleteDisallowed` set in the index schema, as described in section 1.3.2.1.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>anchortext</td>
<td>Anchor text that points to the item.</td>
</tr>
<tr>
<td>assocqueries</td>
<td>The managed property that represents the associated query information for the item.</td>
</tr>
<tr>
<td>docacl</td>
<td>Access right information for the item, as specified in [MS-FSCF] section 2.2.38.</td>
</tr>
<tr>
<td>docvector</td>
<td>The standard managed property for document vector used for the find-similar feature.</td>
</tr>
<tr>
<td>language</td>
<td>The primary language of the item.</td>
</tr>
<tr>
<td>languages</td>
<td>All detected languages in the item.</td>
</tr>
<tr>
<td>url</td>
<td>The item’s primary URL.</td>
</tr>
<tr>
<td>urls</td>
<td>The set of URLs associated with the item and the item duplicates within the item’s equivalence class.</td>
</tr>
</tbody>
</table>

### 2.1.5 Internal Properties

Internal properties are present in configuration files, in addition to managed properties, as specified in section 2.1.4. Requirements associated with internal properties are explicitly stated for each configuration file. The following table specifies the internal property names and their definitions.
### Property | Description
--- | ---
**internalid** | An internal identifier for an item. The internal property value **MUST** be unique across all index columns of a search application.

**contentid** | An identifier that provides a valid external identification of an item that matches a query such as a URI.

**contentids** | If an item can be referenced by more than one valid URIs, this property **MUST** contain a space-separated list of valid **contentid** values.

**collection** | The name of the **content collection** where the item resides.

**ranklog** | A built-in property that is reserved for rank information used for diagnostic purposes.

---

### 2.2 maptransform.xml

Configuration parameters in the maptransform.xml file are derived from the index schema and contain information for managed properties.

The following table provides information about the file.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Middleware Protocol storage path</td>
<td>RTSearch/webcluster/</td>
</tr>
<tr>
<td>Type of data</td>
<td>Configuration information derived from index schema.</td>
</tr>
<tr>
<td>File format</td>
<td>XML schema file.</td>
</tr>
</tbody>
</table>

The configuration specifies how numeric and **datetime** values represented as strings in items and queries are mapped to the data types, as specified in section 2.1.1. The mapping converts all numeric information, including **datetime** values, to 64-bit integer representations that reside in the index data structures.

### 2.2.1 Global Elements

#### 2.2.1.1 transform-specification

The **transform-specification** element contains definitions for data type transformations.

```xml
<xs:element name="transform-specification" type="CT_transform-specification"/>
```

### 2.2.2 Global Attributes

None.

### 2.2.3 Complex Types

#### 2.2.3.1 CT_transform-specification

Referenced by: navigators

A complex type that is a container for data type transformation specifications.
This complex type is defined as follows:

```xml
<xs:complexType name="CT_transform_specification">
  <xs:sequence>
    <xs:element name="datatype-definitions" type="CT_datatype-definitions"/>
    <xs:element name="number-transformations" type="CT_number-transformations"/>
  </xs:sequence>
</xs:complexType>
```

datatype-definitions: A CT_datatype-definitions element that specifies numeric data types.

number-transformations: A CT_number-transformations element that specifies data types for numeric managed properties.

### 2.2.3.2 CT_datatype-definitions

Referenced by: CT_transform_specification

A complex type that specifies numeric data types regarding floating-point configuration. The element contains a set of datatype sub-elements.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_datatype-definitions">
  <xs:sequence>
    <xs:element name="datatype" type="CT_datatype" minOccurs="1" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

datatype: A CT_datatype element that specifies a data type.

### 2.2.3.3 CT_datatype

Referenced by: CT_datatype-definitions

A complex type that specifies a data type for an index schema.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_datatype">
  <xs:attribute name="name" type="xs:string" use="required"/>
  <xs:attribute name="offsetbits" type="ST_offsetbits" use="required"/>
  <xs:attribute name="signbits" type="ST_signbits" use="required"/>
  <xs:attribute name="exponentbits" type="ST_exponentbits" use="required"/>
  <xs:attribute name="mantissabits" type="ST_mantissabits" use="required"/>
  <xs:attribute name="expbase" type="ST_expbase" use="required"/>
  <xs:attribute name="decimalplaces" type="ST_decimalplaces" default="0"/>
  <xs:attribute name="toint" type="ST_toint"/>
</xs:complexType>
```

Attributes:
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>The name of a valid data type. MUST be one of the supported data types used in configuration files, as specified in section 2.1.1.</td>
</tr>
<tr>
<td>offsetbits</td>
<td>An ST_offsetbits attribute.</td>
</tr>
<tr>
<td>signbits</td>
<td>An ST_signbits attribute that specifies whether the data type is signed. MUST be set as specified in sections 2.2.3.3.1 and 2.2.3.3.2.</td>
</tr>
<tr>
<td>exponentbits</td>
<td>An ST_exponentbits attribute that specifies the number of bits used for the exponent. MUST be set as specified in sections 2.2.3.3.1 and 2.2.3.3.2.</td>
</tr>
<tr>
<td>mantissabits</td>
<td>An ST_mantissabits attribute that specifies the number of bits used for the mantissa. MUST be set as specified in sections 2.2.3.3.1 and 2.2.3.3.2.</td>
</tr>
<tr>
<td>expbase</td>
<td>An ST_expbase attribute that specifies the base for the exponent. MUST be set as specified in sections 2.2.3.3.1 and 2.2.3.3.2.</td>
</tr>
<tr>
<td>decimalplaces</td>
<td>An ST_decimalplaces attribute that specifies decimal precision for DECIMAL data types. MUST be set as specified in section 2.2.3.3.2.</td>
</tr>
<tr>
<td>toint</td>
<td>An ST_toint attribute. MUST be set as specified in section 2.2.3.3.2.</td>
</tr>
</tbody>
</table>

The following subsections specify the datatype elements that MUST be present and associate them with the corresponding attribute restrictions.

### 2.2.3.3.1 Required Data Type Definitions

The following datatype elements MUST be present within the datatype-definitions element with the attribute values specified in the following XML.

```
<datatype
   name="INT"
   offsetbits="0"
   signbits="1"
   exponentbits="0"
   mantissabits="63"
   expbase="0"/>
<datatype
   name="FLOAT2B"
   offsetbits="0"
   signbits="1"
   exponentbits="11"
   mantissabits="52"
   expbase="2"/>
<datatype
   name="FLOAT10B"
   offsetbits="0"
   signbits="1"
   exponentbits="11"
   mantissabits="52"
   expbase="10"/>
<datatype
   name="DATETIME"
```
Section 2.1.1 specifies the relation to index schema types.

The FLOAT10B data type MUST be specified as indicated, but is not used.

2.2.3.3.2 Index Schema-Dependent Data Type Definitions

The datatype elements with name beginning with "DECIMAL" MUST be mapped from the corresponding data types in the index schema, as specified in section 2.1.1. Two datatype elements MUST be specified for each unique combination of index schema data type Decimal and a corresponding DecimalPrecision value, as specified in section 1.3.2.1.

A datatype element must be specified with the following attributes set:

- **name**: "DECIMAL_<N>", where <N> is the DecimalPrecision value
- **offsetbits**: 0
- **signbits**: 1
- **exponentbits**: 0
- **mantissabits**: 63
- **expbase**: 0
- **decimalplaces**: "<N>", where <N> is the DecimalPrecision value

A datatype element must be specified with the following attributes set:

- **name**: "DECIMAL_NAV_<N>", where <N> is the DecimalPrecision value
- **offsetbits**: 0
- **signbits**: 1
- **exponentbits**: 0
- **mantissabits**: 63
- **expbase**: 0
- **decimalplaces**: "<N>", where <N> is the DecimalPrecision value
- **toint**: yes

2.2.3.4 CT_number-transformations

Referenced by: CT_transform-specification

A complex type that specifies data types for numeric managed properties used for converting text-formatted data types to the 64-bit numeric representation that is internal to the index.
Managed properties of type **String** MUST NOT be specified in this element.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_number-transformations">
  <xs:sequence>
    <xs:element name="field" type="CT_field"
      minOccurs="1" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

**field**: A **CT_field** element that specifies the data type for a managed property.

Attributes: None.

### 2.2.3.5 CT_field

Referenced by: **CT_number-transformations**

A complex type that specifies the data type for a managed property, as specified in the index schema.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_field">
  <xs:attribute name="name" type="xs:string" use="required"/>
  <xs:attribute name="datatype" type="xs:string"/>
</xs:complexType>
```

Attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>The name of a numeric managed property.</td>
</tr>
<tr>
<td>datatype</td>
<td>MUST be one of the data types specified in the <strong>datatype-definitions</strong> element, as specified in section 2.2.3.3.</td>
</tr>
</tbody>
</table>

### 2.2.4 Simple Types

#### 2.2.4.1 ST_offsetbits

Referenced by: **CT_datatype**

A simple type that specifies the number of offset bits. This parameter is not used and MUST be 0.

```xml
<xs:simpleType name="ST_offsetbits">
  <xs:restriction base="xs:string">
    <xs:enumeration value="0"/>
  </xs:restriction>
</xs:simpleType>
```
2.2.4.2  **ST_signbits**

Referenced by: **CT_datatype**

A simple type that specifies the number of bits used for the signed value indication.

```xml
<xs:simpleType name="ST_signbits">
  <xs:restriction base="xs:string">
    <xs:enumeration value="0"/>
    <xs:enumeration value="1"/>
  </xs:restriction>
</xs:simpleType>
```

2.2.4.3  **ST_exponentbits**

Referenced by: **CT_datatype**

A simple type that specifies the number of bits that the exponent uses.

```xml
<xs:simpleType name="ST_exponentbits">
  <xs:restriction base="xs:string">
    <xs:enumeration value="0"/>
    <xs:enumeration value="11"/>
  </xs:restriction>
</xs:simpleType>
```

2.2.4.4  **ST_mantissabits**

Referenced by: **CT_datatype**

A simple type that specifies number of bits that the mantissa uses.

```xml
<xs:simpleType name="ST_mantissabits">
  <xs:restriction base="xs:string">
    <xs:enumeration value="52"/>
    <xs:enumeration value="63"/>
    <xs:enumeration value="64"/>
  </xs:restriction>
</xs:simpleType>
```

2.2.4.5  **ST_expbase**

Referenced by: **CT_datatype**

A simple type that specifies the exponent base.

```xml
<xs:simpleType name="ST_expbase">
  <xs:restriction base="xs:string">
    <xs:enumeration value="0"/>
    <xs:enumeration value="2"/>
    <xs:enumeration value="10"/>
  </xs:restriction>
</xs:simpleType>
```
2.2.4.6 **ST_decimalplaces**

Referenced by: **CT_datatype**

A simple type that specifies decimal precision in a range from 0 through 32.

```xml
<xs:simpleType name="ST_decimalplaces">
  <xs:restriction base="xs:integer">
    <xs:minInclusive value="0"/>
    <xs:maxInclusive value="32"/>
  </xs:restriction>
</xs:simpleType>
```

2.2.4.7 **ST_toint**

Referenced by: **CT_datatype**

A simple type that specifies an implementation-specific parameter used for decimal data types.

```xml
<xs:simpleType name="ST_toint">
  <xs:restriction base="xs:string">
    <xs:enumeration value="yes"/>
  </xs:restriction>
</xs:simpleType>
```

2.3 **fieldspec.xml**

The following table provides information about the file.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Middleware Protocol storage path</td>
<td>QRServer/webcluster/etc/qrserver/</td>
</tr>
<tr>
<td>Type of data</td>
<td>Configuration information derived from index schema.</td>
</tr>
<tr>
<td>File format</td>
<td>XML schema file.</td>
</tr>
</tbody>
</table>

This file contains data type sort configuration information for a protocol client that implements the protocol specified in [MS-FSDQE]. It also contains formatting for queries submitted through that protocol.

### 2.3.1 Global Elements

#### 2.3.1.1 fieldlist

The **fieldlist** element is a container for **field** elements.

```xml
<xs:element name="fieldlist" type="CT_fieldlist"/>
```

### 2.3.2 Global Attributes

None.
2.3.3 Complex Types

2.3.3.1 CT_fieldlist

Referenced by: fieldlist

A complex type that specifies field elements for all sortable managed properties.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_fieldlist">
  <xs:sequence>
    <xs:element name="field" type="CT_field" minOccurs="1" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

field: A CT_field element.

Attributes: None.

2.3.3.2 CT_field

Referenced by: CT_fieldlist

A complex type that specifies sorting configuration for a managed property.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_field">
  <xs:attribute name="name" type="xs:string" use="required"/>
  <xs:attribute name="sorttype" type="ST_sorttype"/>
</xs:complexType>
```

Attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>The name of a sortable managed property.</td>
</tr>
<tr>
<td>sorttype</td>
<td>An ST_sorttype attribute that specifies sort configuration for the managed property.</td>
</tr>
</tbody>
</table>

2.3.4 Simple Types

2.3.4.1 ST_sorttype

Referenced by: CT_field

A simple type that specifies sort configuration for the managed property.

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_sorttype">
  <xs:restriction base="xs:string">
  </xs:restriction>
</xs:simpleType>
```
2.4 resultfield.map

The following table provides information about the file.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Middleware Protocol storage path</td>
<td>QRServer/webcluster/etc/qrserver/</td>
</tr>
<tr>
<td>Type of data</td>
<td>Implementation-specific configuration information.</td>
</tr>
<tr>
<td>File format</td>
<td>Name: value text file.</td>
</tr>
</tbody>
</table>

2.4.1 File Content

The configuration file content is static and MUST contain the following.

```plaintext
url: contentid
docvector: docvector
```

2.4.2 Configuration Parameter Details

The file contains a set of index field name mappings of the following type:

- `<result field name>`: `<field name returned by [MS-FSDQE]>`

The following table specifies the two mapping configurations that MUST be in the file.

<table>
<thead>
<tr>
<th>Mapping</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>url: contentid</td>
<td>Not used, but MUST be in the file.</td>
</tr>
<tr>
<td>docvector: docvector</td>
<td>The standard managed property for the document vector used for the find-similar feature.</td>
</tr>
</tbody>
</table>

2.5 configuration.attributes.xml

The following table provides information about the file.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Middleware Protocol storage path</td>
<td>QRServer/webcluster/etc/qrserver/tango</td>
</tr>
<tr>
<td>Item</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Type of data</td>
<td>Configuration information derived from index schema.</td>
</tr>
<tr>
<td>File format</td>
<td>XML schema file.</td>
</tr>
</tbody>
</table>

Configuration parameters in this file are derived from the index schema and contain refiner configuration information.

The configuration information enables processing and presentation of refiner data in query results according to the configuration of the refiner index schema.

### 2.5.1 Global Elements

#### 2.5.1.1 navigators

The `navigators` element contains all refiner definitions, as specified by the `navigator` element.

```xml
<xs:element name="navigators" type="CT_navigators"/>
```

### 2.5.2 Global Attributes

None.

### 2.5.3 Complex Types

#### 2.5.3.1 CT_navigators

Referenced by: `<navigators>`

A complex type that specifies a list of refiner definitions. This corresponds to the index schema `Refiner` element specified in section 1.3.2.3.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_navigators">
  <xs:sequence>
    <xs:element maxOccurs="unbounded" name="navigator" type="CT_navigator"/>
  </xs:sequence>
</xs:complexType>
```

**navigator**: A `CT_navigator` element that specifies a refiner.

Attributes: None.

#### 2.5.3.2 CT_navigator

Referenced by: `CT_navigators`

A complex type that specifies a refiner used for processing of a query result.

This complex type is defined as follows:
<xs:complexType name="CT_navigator">
  <xs:all>
    <xs:element name="datetime" type="CT_datetimeNav" minOccurs="0" maxOccurs="1"/>
    <xs:element name="integer" type="CT_numericNav" minOccurs="0" maxOccurs="1"/>
    <xs:element name="double" type="CT_numericNav" minOccurs="0" maxOccurs="1"/>
    <xs:element name="fixedpoint" type="CT_fixedpoint" minOccurs="0" maxOccurs="1"/>
    <xs:element name="string" type="CT_stringNav" minOccurs="0" maxOccurs="1"/>
    <xs:element name="score" type="CT_score" minOccurs="1" maxOccurs="1"/>
  </xs:all>
  <xs:attribute name="deephits" type="xs:int" use="required"/>
  <xs:attribute name="name" type="xs:string" use="required"/>
  <xs:attribute name="cutminbuckets" type="xs:int" use="optional"/>
  <xs:attribute name="deep" type="ST_deep" use="required"/>
  <xs:attribute name="passive" type="ST_passive" use="required"/>
  <xs:attribute name="field" type="xs:string" use="required"/>
  <xs:attribute name="separator" type="xs:string" use="optional"/>
  <xs:attribute name="cutmaxbuckets" type="xs:int" use="optional"/>
  <xs:attribute name="cutfreq" type="xs:int" use="optional"/>
  <xs:attribute name="type" type="ST_type" use="required"/>
  <xs:attribute name="display" type="xs:string" use="required"/>
  <xs:attribute name="unit" type="xs:string" use="required"/>
  <xs:attribute name="multimode" type="ST_multimode" use="optional"/>
  <xs:attribute name="signed" type="ST_signed" use="optional"/>
</xs:complexType>

datetime: A CT_datetime element. MUST be present if type is set to "datetime".
integer: A CT_numericNav element. MUST be present if type is set to "integer".
double: A CT_numericNav element. MUST be present if type is set to "float".
fixedpoint: A CT_fixedpoint element. MUST be present if type is set to "fixedpoint".
string: A CT_stringNav element. MUST be present if type is set to "string".
score: A CT_score element.

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>deephits</td>
<td>An implementation-specific parameter. MUST be set to 0.</td>
</tr>
<tr>
<td>name</td>
<td>The name of the refiner. MUST be the name of the managed property on which the refiner is based.</td>
</tr>
<tr>
<td>cutminbuckets</td>
<td>An implementation-specific parameter. MUST be set to 0 for refiners of type string. MUST NOT be present for refiners of other type.</td>
</tr>
<tr>
<td>deep</td>
<td>An ST_yesno attribute that specifies deep refinement configuration.</td>
</tr>
<tr>
<td>yes</td>
<td>Deep refinement is configured. Corresponds to RefinementType set to &quot;DeepRefinementEnabled&quot; in the index schema, as specified in section 1.3.2.3.</td>
</tr>
<tr>
<td>no</td>
<td>Shallow refinement is configured. Corresponds to RefinementType set to &quot;DeepRefinementDisabled&quot; in the index schema, as specified</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>passive</td>
<td>An <code>ST_alwaysno</code> attribute that specifies an implementation-specific parameter.</td>
</tr>
<tr>
<td>field</td>
<td>The name of the managed property associated with the refiner.</td>
</tr>
<tr>
<td>separator</td>
<td>The separator character sequence used to separate multiple string values in a multi-value managed property. MUST be set to a semicolon (;) to enable multi-value refiners that correspond to <code>IsMultiValued</code> set to &quot;yes&quot; in the index schema, as specified in section 1.3.2.1. MUST be set to an empty string (&quot;&quot;), for all other refiners of type string. MUST NOT be present for all other refiners.</td>
</tr>
<tr>
<td>cutmaxbuckets</td>
<td>The limit for the number of unique refinement bins that are returned for a search query. Corresponds to <code>CutoffMaxBuckets</code> in the index schema, as specified in section 1.3.2.3.</td>
</tr>
<tr>
<td>cutfreq</td>
<td>An <code>ST_alwaysZero</code> implementation-specific attribute. MUST be present for all refiners of type string. MUST NOT be present for all other refiners.</td>
</tr>
<tr>
<td>modifier</td>
<td>The refinement modifier. Corresponds to the <code>Name</code> attribute in the index schema, as specified in section 1.3.2.1.</td>
</tr>
<tr>
<td>type</td>
<td>An <code>ST_type</code> attribute that specifies the refiner type.</td>
</tr>
<tr>
<td>display</td>
<td>The display leading string for the refiner.</td>
</tr>
<tr>
<td>unit</td>
<td>The display unit string for the refiner.</td>
</tr>
<tr>
<td>multimode</td>
<td>An <code>ST_multimode</code> implementation-specific attribute.</td>
</tr>
<tr>
<td>signed</td>
<td>Specifies whether this refiner is based on a managed property with a signed data type. MUST be set to a value of &quot;yes&quot; for refiners associated with managed properties of types <code>Integer</code>, <code>Decimal</code>, <code>Float</code>, and <code>Double</code>. MUST be set to &quot;no&quot; for all other refiners. For more information about managed property types, see section 1.3.2.1.</td>
</tr>
</tbody>
</table>

### 2.5.3.3 CT_datetimeNav

Referenced by: CT_navigator

A complex type that specifies a datetime refiner. A `datetime` refiner is a special case of an integer refiner.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_datetimeNav">
    <xs:sequence>
        <xs:element name="integer" type="CT_numericNav" minOccurs="1" maxOccurs="1"/>
    </xs:sequence>
</xs:complexType>

integer: A CT_numericNav element.
2.5.3.4 CT_fixedpoint

Referenced by: CT_navigator

A complex type that specifies a decimal refiner. A decimal refiner is a special case of an integer refiner element.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_fixedpoint">
  <xs:sequence>
    <xs:element name="integer" type="CT_numericNav" minOccurs="1" maxOccurs="1"/>
  </xs:sequence>
  <xs:attribute name="decimals" type="xs:int" use="required"/>
</xs:complexType>
```

integer: A CT_numericNav element.

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>decimals</td>
<td>Specifies the decimal precision for a refiner. MUST be set to the same decimal precision as the associated managed property. For more information about the DecimalPrecision index schema, see section 1.3.2.1.</td>
</tr>
</tbody>
</table>

2.5.3.5 CT_numericNav

Referenced by: CT_navigator, CT_datetimeNav, CT_fixedpoint

A complex type that specifies a numeric refiner.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_numericNav">
  <xs:sequence>
    <xs:element name="discretize" type="CT_discretize"/>
    <xs:element name="display" type="CT_display"/>
  </xs:sequence>
</xs:complexType>
```

discretize: A CT_discretize element.

display: A CT_display element.

Attributes: None.

2.5.3.6 CT_stringNav

Referenced by: CT_navigator

A complex type that specifies a string refiner.

This complex type is defined as follows:
sort: A `CT_sort` element.

filter: A `CT_filter` element.

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>anchoring</td>
<td>An <code>ST_anchoring</code> attribute that specifies matching mode for string refinement modifiers.</td>
</tr>
</tbody>
</table>

### 2.5.3.7 CT_sort

Referenced by: `CT_stringNav`

A complex type that specifies sort criteria for a refiner.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_sort">
  <xs:attribute name="by" type="ST_by" use="required"/>
  <xs:attribute name="order" type="ST_order" use="required"/>
</xs:complexType>
```

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>By</td>
<td>An <code>ST_by</code> attribute that specifies the sorting algorithm for string refiners.</td>
</tr>
<tr>
<td>Order</td>
<td>An <code>ST_order</code> attribute that specifies sorting order.</td>
</tr>
</tbody>
</table>

### 2.5.3.8 CT_filter

Referenced by: `CT_stringNav`

A complex type that specifies filter criteria for a refiner.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_filter">
  <xs:attribute name="buckets" type="xs:integer" use="required"/>
  <xs:attribute name="frequency" type="xs:integer" use="required"/>
</xs:complexType>
```

Attributes:
### Name | Description
---|---
Buckets | Specifies the maximum number of returned refinement bins for a refiner.
Frequency | Specifies a limit for returned refinement bins based on frequency.

#### 2.5.3.9 CT_display

Referenced by: **CT_numericNav**

A complex type that specifies display properties for a numeric refiner. The element MUST be present for all refiner types except the string type, as specified in section 2.5.3.2.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_display">
  <xs:sequence>
    <xs:element name="first" type="CT_firstLast"/>
    <xs:element name="middle" type="CT_middle"/>
    <xs:element name="last" type="CT_firstLast"/>
  </xs:sequence>
  <xs:attribute name="divisor" type="xs:float" use="required"/>
</xs:complexType>
```

**first**: A **CT_firstLast** element that specifies the display string for the first refinement bin.

**middle**: A **CT_middle** element that specifies the display string for all refinement bins except the first and last bin.

**last**: A **CT_firstLast** element that specifies the display string for the last refinement bin.

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>divisor</td>
<td>The divisor used to scale down refinement values before displaying to the user. This corresponds to <strong>Divisor</strong> in the index schema, as specified in section 1.3.2.3.</td>
</tr>
</tbody>
</table>

#### 2.5.3.10 CT_firstLast

Referenced by: **CT_display**

A complex type that specifies a display string for the first and last refinement bin.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_firstLast">
  <xs:attribute name="offset" type="xs:int" use="required"/>
  <xs:attribute name="format" type="xs:string" use="required"/>
</xs:complexType>
```

Attributes:
### Name | Description
--- | ---
**offset** | An offset value that is added to the label value. This MUST be set to 0.

**format** | A formatting text string that defines the text and the formatting of the value range in the refiner labels. The implementer can use it to specify how the label string is formatted. The text string MUST contain one C-style sprintf format code of type `%g` that is replaced with the actual value. MUST be set to the value "Before %s" (first refinement bin) and "%s or later" (last refinement bin) for **datetime** refiners. MUST be set to the value "Less than %s" (first refinement bin) and "%s and up" (last refinement bin) for all other numeric refiners.

### 2.5.3.11 CT_middle

**Referenced by:** **CT_display**

A complex type that specifies a display string for all refinement bins except the first and last bin.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_middle">
    <xs:attribute name="offset1" type="xs:int" use="required"/>
    <xs:attribute name="offset2" type="xs:int" use="required"/>
    <xs:attribute name="format" type="xs:string" use="required"/>
</xs:complexType>
```

#### Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>offset1</strong></td>
<td>An offset value that is added to the label value for the first value. MUST be set to 0.</td>
</tr>
<tr>
<td><strong>offset2</strong></td>
<td>An offset value that is added to the label value for the second value. MUST be set to 0.</td>
</tr>
<tr>
<td><strong>format</strong></td>
<td>A text string with formatting codes. The implementer uses it to specify how the label string is formatted. The text string MUST contain two C-style sprintf format codes of type <code>%g</code> that are replaced with the actual value. MUST be set to the value &quot;From %s to %s&quot; for <strong>datetime</strong> refiners. MUST be set to the value &quot;%s up to %s&quot; for all other numeric refiners.</td>
</tr>
</tbody>
</table>

### 2.5.3.12 CT_discretize

**Referenced by:** **CT_numericNav**

A complex type that specifies attributes for refiner discretization. The **discretize** element MUST contain an **equalfrequency**, an **equalwidth**, or a **rangedivision** element.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_discretize">
    <xs:choice>
        <xs:element name="equalfrequency" type="CT_equalfrequency"/>
    </xs:choice>
</xs:complexType>
```
equalfrequency: A CT_equalfrequency element. This element MUST be present if the algorithm contains the value "equalfrequency".

equalwidth: A CT_equalwidth element. This element MUST be present if the algorithm contains the value "equalwidth".

rangedivision: A CT_rangedivision element. This element MUST be present if the algorithm contains the value "rangedivision".

These child elements specify the value distribution for the refinement bins. Refer to section 1.3.2.3 for more details.

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>algorithm</td>
<td>An ST_algorithm attribute that specifies the discretization algorithm.</td>
</tr>
</tbody>
</table>

2.5.3.13 CT_equalfrequency

Referenced by: CT_discretize

A complex type that specifies parameters for the equalfrequency mode.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_equalfrequency">
  <xs:attribute name="intervals" type="xs:int" use="required"/>
  <xs:attribute name="resolution" type="xs:int" use="required"/>
</xs:complexType>
```

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>intervals</td>
<td>The maximum number of refinement bins generated. Corresponds to Intervals in the index schema, as specified in section 1.3.2.3.</td>
</tr>
<tr>
<td>resolution</td>
<td>The resolution of the returned refinement bins. Corresponds to Resolution in the index schema, as specified in section 1.3.2.3.</td>
</tr>
</tbody>
</table>

2.5.3.14 CT_rangedivision

Referenced by: CT_discretize

A complex type that specifies parameters for the rangedivision mode.

This complex type is defined as follows:
<xs:complexType name="CT_rangedivision">
    <xs:attribute name="intervals" type="xs:int" use="required"/>
    <xs:attribute name="resolution" type="xs:int" use="required"/>
</xs:complexType>

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>intervals</td>
<td>The maximum number of refinement bins generated. Corresponds to <strong>Intervals</strong> in the index schema, as specified in section 1.3.2.3.</td>
</tr>
<tr>
<td>resolution</td>
<td>The resolution of the returned refinement bins. Corresponds to <strong>Resolution</strong> in the index schema, as specified in section 1.3.2.3.</td>
</tr>
</tbody>
</table>

2.5.3.15 **CT_equalwidth**

Referenced by: **CT_discretize**

A complex type that specifies parameters for the **equalwidth** mode.

This complex type is defined as follows:

<xs:complexType name="CT_equalwidth">
    <xs:attribute name="resolution" type="xs:int" use="required"/>
</xs:complexType>

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>resolution</td>
<td>The resolution of the returned refinement bins. Corresponds to <strong>Resolution</strong> in the index schema, as specified in section 1.3.2.3.</td>
</tr>
</tbody>
</table>

2.5.3.16 **CT_score**

Referenced by: **CT_navigator**

A complex type that specifies implementation-specific scoring parameters for creating refinement bins. The values are not configurable and MUST be according to the XML schema.

This complex type is defined as follows:

<xs:complexType name="CT_score">
    <xs:attribute name="count" type="ST_alwaysZero" use="required"/>
    <xs:attribute name="constant" type="ST_alwaysOne" use="required"/>
    <xs:attribute name="buckets" type="ST_alwaysZero" use="required"/>
    <xs:attribute name="entropy" type="ST_alwaysOne" use="required"/>
    <xs:attribute name="offset" type="ST_alwaysZero" use="required"/>
    <xs:attribute name="ratio" type="ST_alwaysZero" use="required"/>
</xs:complexType>
Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>count</td>
<td>An implementation-specific parameter.</td>
</tr>
<tr>
<td>constant</td>
<td>An implementation-specific parameter.</td>
</tr>
<tr>
<td>buckets</td>
<td>An implementation-specific parameter.</td>
</tr>
<tr>
<td>entropy</td>
<td>An implementation-specific parameter.</td>
</tr>
<tr>
<td>offset</td>
<td>An implementation-specific parameter.</td>
</tr>
<tr>
<td>ratio</td>
<td>An implementation-specific parameter.</td>
</tr>
</tbody>
</table>

2.5.4 Simple Types

2.5.4.1 ST_type

Referenced by: CT_navigator

A simple type that specifies the refiner type. For more information about the types of managed properties, see section 1.3.2.1.

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_type">
  <xs:restriction base="xs:string">
    <xs:enumeration value="string"/>
    <xs:enumeration value="datetime"/>
    <xs:enumeration value="integer"/>
    <xs:enumeration value="float"/>
    <xs:enumeration value="fixedpoint"/>
  </xs:restriction>
</xs:simpleType>
```

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>A refiner for a managed property of type Text or Boolean.</td>
</tr>
<tr>
<td>datetime</td>
<td>A refiner for a managed property of type Datetime.</td>
</tr>
<tr>
<td>integer</td>
<td>A refiner for a managed property of type Integer.</td>
</tr>
<tr>
<td>float</td>
<td>A refiner for a managed property of type Float.</td>
</tr>
<tr>
<td>fixedpoint</td>
<td>A refiner for a managed property of type Decimal.</td>
</tr>
</tbody>
</table>

2.5.4.2 ST_multimode

Referenced by: CT_navigator

A simple type that specifies an implementation-specific parameter with a fixed value.

This simple type is defined as follows:
<xs:simpleType name="ST_multimode">
    <xs:restriction base="xs:string">
        <xs:enumeration value="needed"/>
    </xs:restriction>
</xs:simpleType>

2.5.4.3 ST_anchoring

Referenced by: CT_stringNav

A simple type that specifies matching mode for string refinement modifiers. Corresponds to Anchoring in the index schema, as specified in section 1.3.2.3.

This simple type is defined as follows:

<xs:simpleType name="ST_anchoring">
    <xs:restriction base="xs:string">
        <xs:enumeration value="auto"/>
        <xs:enumeration value="none"/>
        <xs:enumeration value="complete"/>
        <xs:enumeration value="prefix"/>
        <xs:enumeration value="suffix"/>
    </xs:restriction>
</xs:simpleType>

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>auto</td>
<td>The same as &quot;complete&quot; if boundary match is enabled for the managed property. Otherwise, it is the same as &quot;none&quot;.</td>
</tr>
<tr>
<td>none</td>
<td>The refinement modifiers are not anchored.</td>
</tr>
<tr>
<td>complete</td>
<td>The refinement modifiers are anchored to both the beginning and the end of the index field.</td>
</tr>
<tr>
<td>prefix</td>
<td>The refinement modifiers are anchored to the beginning of the index field.</td>
</tr>
<tr>
<td>suffix</td>
<td>The refinement modifiers are anchored to the end of the index field.</td>
</tr>
</tbody>
</table>

2.5.4.4 ST_algorithm

Referenced by: CT_discretize

A simple type that specifies the discretization algorithm. Corresponds to Algorithm in the index schema, as specified in section 1.3.2.3.

This simple type is defined as follows:

<xs:simpleType name="ST_algorithm">
    <xs:restriction base="xs:string">
        <xs:enumeration value="equalfrequency"/>
        <xs:enumeration value="equalwidth"/>
        <xs:enumeration value="rangedivision"/>
    </xs:restriction>
</xs:simpleType>
<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>equalfrequency</td>
<td>The value range of different refinement bins can have different widths.</td>
</tr>
<tr>
<td>equalwidth</td>
<td>The value range of each refinement bin is always equal.</td>
</tr>
<tr>
<td>rangedivision</td>
<td>The value range of each refinement bin is considered to be equal. The width is computed dynamically and is not required to be equal.</td>
</tr>
</tbody>
</table>

### 2.5.4.5 ST_by

Referenced by: **CT_sort**

A simple type that specifies the sorting algorithm for string refiners.

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_by">
  <xs:restriction base="xs:string">
    <xs:enumeration value="auto"/>
    <xs:enumeration value="name"/>
    <xs:enumeration value="frequency"/>
    <xs:enumeration value="number"/>
  </xs:restriction>
</xs:simpleType>
```

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>auto</td>
<td>Orders a combination of frequency and number. Numeric sorting is used if the index field content is numbers; otherwise, frequency sorting is used.</td>
</tr>
<tr>
<td>name</td>
<td>Orders by refinement name.</td>
</tr>
<tr>
<td>frequency</td>
<td>Orders by occurrence within the refinement bins.</td>
</tr>
<tr>
<td>number</td>
<td>Treats the strings as numeric and uses numeric sorting.</td>
</tr>
</tbody>
</table>

### 2.5.4.6 ST_order

Referenced by: **CT_sort**

A simple type that specifies the sorting direction.

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_order">
  <xs:restriction base="xs:string">
    <xs:enumeration value="ascending"/>
    <xs:enumeration value="descending"/>
  </xs:restriction>
</xs:simpleType>
```

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>ascending</td>
<td>Ascending sorting order.</td>
</tr>
<tr>
<td>Value</td>
<td>Meaning</td>
</tr>
<tr>
<td>-------------</td>
<td>----------------------------------------</td>
</tr>
<tr>
<td>descending</td>
<td>Descending sorting order.</td>
</tr>
</tbody>
</table>

2.5.4.7  **ST_alwaysOne**

Referenced by: **CT_score**

A simple type that specifies a fixed attribute value of 1.

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_alwaysOne">
  <xs:restriction base="xs:string">
    <xs:enumeration value="1"/>
  </xs:restriction>
</xs:simpleType>
```

2.5.4.8  **ST_alwaysZero**

Referenced by: **CT_score**

A simple type that specifies a fixed attribute value of 0.

```xml
<xs:simpleType name="ST_alwaysZero">
  <xs:restriction base="xs:string">
    <xs:enumeration value="0"/>
  </xs:restriction>
</xs:simpleType>
```

2.5.4.9  **ST_yesno**

Referenced by: **CT_navigator**

A simple type that specifies the Boolean condition values "yes" and "no".

This simple type is defined as follows.

```xml
<xs:simpleType name="ST_yesno">
  <xs:restriction base="xs:string">
    <xs:enumeration value="yes"/>
    <xs:enumeration value="no"/>
  </xs:restriction>
</xs:simpleType>
```

2.5.4.10 **ST_alwaysno**

Referenced by: **CT_navigator**

A simple type that specifies a fixed attribute value of "no".

This simple type is defined as follows:
2.6 fdispatch.addon

The following table provides information about the file.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Middleware Protocol storage path</td>
<td>RTSearch/webcluster/</td>
</tr>
<tr>
<td>Type of data</td>
<td>Non-configurable protocol-related data.</td>
</tr>
<tr>
<td>File format</td>
<td>Name value text file.</td>
</tr>
</tbody>
</table>

This file contains configuration information that is used to implement the protocol specified in [MS-FSDQE]. The data is static, or not configurable.

2.6.1 File Content

The configuration file content is static, or not configurable, and MUST be as specified in the following code.

```plaintext
maxoffset = 4000

# Don't change the Juniper settings in this file - they must be in sync with the
# config in rtsearch/fsearch.addon (fsearchrc)
#
# juniper.dynsum.highlight_on \x02
# juniper.dynsum.highlight_off \x03
# juniper.dynsum.continuation \x1E
```

One newline (LF, ASCII decimal value 10) character MUST delimit each line. A carriage return (CR, ASCII decimal value 13) MAY precede the LF character.

2.6.2 Configuration Parameter Details

The following table specifies the configuration parameter lines that MUST be present in this file. The line syntax and parameter values MUST be as specified in the Parameter column in the following table.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>maxoffset = 4000</td>
<td>The maxoffset parameter specifies the maximum offset requested in a search query on [MS-FSDQE].</td>
</tr>
<tr>
<td>juniper.dynsum.highlight_on \x02</td>
<td>The hexadecimal byte value 0x02 that specifies the beginning of highlighting in a dynamic teaser object.</td>
</tr>
<tr>
<td>juniper.dynsum.highlight_off \x03</td>
<td>The hexadecimal byte value 0x03 that specifies the end of highlighting in a dynamic teaser object.</td>
</tr>
</tbody>
</table>
2.7  fsearch.addon

The following table provides information about the file.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Middleware Protocol storage path</td>
<td>RTSearch/webcluster/</td>
</tr>
<tr>
<td>Type of data</td>
<td>Implementation-specific configuration information. Non-configurable protocol-related information. Configuration information derived from index schema.</td>
</tr>
<tr>
<td>File format</td>
<td>Name value text file.</td>
</tr>
</tbody>
</table>

This file contains configuration information for a protocol server that is implementing the protocol specified in [MS-FSDQE]. The file specifies configuration information for evaluation of queries and creation of hit highlighted summaries.

The file contains three main types of configuration information, as specified in the following subsections. The sequence of the parameters in the file SHOULD follow the sequence specified in the following subsections.

2.7.1  Static Hit Highlighted Summary Parameters

The following table specifies static configuration parameters associated with the hit highlighted summary for the query result.

All parameters specified MUST be present in the file, and the parameter values MUST NOT be changed from specified values.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>juniper.dynsum.highlight_on \x02</td>
<td>Hexadecimal byte value 0x02 that specifies the beginning of highlighting in a dynamic teaser object.</td>
</tr>
<tr>
<td>juniper.dynsum.highlight_off \x03</td>
<td>Hexadecimal byte value 0x03 that specifies the end of highlighting in a dynamic teaser object.</td>
</tr>
<tr>
<td>juniper.dynsum.continuation \x1E</td>
<td>Hexadecimal byte value 0x1E that separates sections in a dynamic teaser object. A section is a consecutive snippet of text from the matched item. Different sections within the teaser object can derive from different parts of the matched item.</td>
</tr>
<tr>
<td>juniper.dynsum.escape_markup off</td>
<td>Highlighting markup for the dynamic</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>teaser obje</td>
<td>teaser object is not escaped on the [MS-FSQR] interface.</td>
</tr>
<tr>
<td>juniper.regions phrase_break</td>
<td>Specifies that phrase break is used.</td>
</tr>
<tr>
<td>juniper.phrase_break.query \xC7\x82</td>
<td>The two configured byte values 0xc7 and 0x82 are recognized as phrase break characters in the dynamic teaser object source in the FAST Index Markup Language (FIXML) object. A section of a teaser object MUST NOT span across any of these phrase break characters. These phrase break characters MUST be removed from the resulting teaser object. For more information about syntax details, see [MS-FSFIXML].</td>
</tr>
<tr>
<td>juniper.phrase_break.type partitioning</td>
<td>The phrase break implies partitioning of the text when performing dynamic teaser object evaluation.</td>
</tr>
<tr>
<td>juniper.phrase_break.resist 10</td>
<td>The proximity resistance, which specifies the equivalent word distance between two terms divided by a phrase break.</td>
</tr>
<tr>
<td>juniper.phrase_break.report filter</td>
<td>Remove phrase break sequence.</td>
</tr>
<tr>
<td>juniper.matcher.wordfolder.type advanced</td>
<td>An implementation-specific parameter.</td>
</tr>
<tr>
<td>juniper.dynsum.separators \x09\x1F\x1D</td>
<td>The three configured byte values 0x09, 0x1f, and 0x1d are recognized as special word separator characters in the dynamic teaser object source in FIXML. For more details, see [MS-FSFIXML]. These word separators MUST be removed from the resulting dynamic teaser object. Languages such as Chinese, Japanese, and Korean, known as CJK languages, do not have a consistent way to separate searchable tokens in the text. The way a sentence is split into searchable tokens depends on context. The separator characters specify word separation that was not present in the source of the indexed item.</td>
</tr>
<tr>
<td>simplequery.config.parent normalteaser simplequery.dynsum.stat yes simplequery.dynsum.stat.type query simplequerystat.config.parent simplequery juniper.views nohigh;nomarkup;stat;query;hithighlight normalteaser.config.parent juniper normalteaser.view.query simplequery normalteaser.view.stat simplequerystat normalteaser.view.nohigh normalteaser_nohigh normalteaser_nohigh.config.parent normalteaser</td>
<td>These parameters are implementation-specific.</td>
</tr>
</tbody>
</table>
Parameter | Description
---|---
normalteaser_nohigh.dynsum.highlight_on | 
normalteaser_nohigh.dynsum.highlight_off | 
teasermarkup.dynsum.highlight_on | 
teasermarkup.dynsum.highlight_off | 
teasermarkup.dynsum.escape_markup on | 
teasermarkup.dynsum.ref_field bsumviewsourcelurl | 
juniper.rewriter.config lemconfig | 
juniper.rewriter.lemconfig.type lemmatizer | 
juniper.rewriter.lemconfig.config | 
etc/LemmatizationConfig.xml | 
juniper.rewriter.lemconfig.for_query 1 | 

### 2.7.2 Configuration Parameters Derived from Index Schema

The following table specifies configuration parameters that are derived from the index schema. Each row in the table represents ABNF grammar for one parameter line in the file.

<table>
<thead>
<tr>
<th>Parameter ABNF grammar</th>
<th>Description</th>
</tr>
</thead>
</table>
| parameter = prname 
".matcher.indexes " idxs 
prname = 1*(DIGIT / ALPHA) 
idxs = idx +(";" idx) 
idx = seprop / syfield / fufield / lev 
seprop = prname 
syfield = "bt1bidx" prname 
fufield = 1*(DIGIT / ALPHA) 
lev = cat "." level 
cat = "bcat" fufield 
level = "bidx" fufield "lvl1" | An implementation-specific configuration for each managed property that has **SummaryType** set to "Dynamic", according to section [1.3.2.1](#).  
**prname**: The name of the managed property.  
**seprop**: The name of the managed property. MUST be included if the managed property has **Queryable** set to "yes", according to section [1.3.2.1](#).  
**syfield**: The name of the managed property with prefix "bt1bidx". MUST be included if the managed property has **Queryable** set to "yes", according to section [1.3.2.1](#).  
**fufield**: The name of a full-text index field. MUST be included for each full-text index field of which the managed property is part.  
**lev**: MUST be included for each full-text index field of which the managed property is part. |
| parameter = "juniper.config.default_index" 
cat | Specifies the default index for search queries. The default index is used when no property index is supplied as a parameter to the search query or if the property index does not exist. MUST be specified as the name of the full-text index field without the field prefix.  
**cat**: The name of the default context catalog.  
Corresponds to the value of the index schema **FullTextIndex Name** attribute for the **FullTextIndex** element with **IsDefault** set to **true**, as specified in section [1.3.2.2](#). |
| parameter = prop ".config.parent normalteaser" 
prop = 1*(DIGIT / ALPHA) | One entry MUST appear in the file for each managed property that supports dynamic document summary.  
This corresponds to managed properties in the index schema that has **SummaryType** set to "Dynamic", according to section [1.3.2.1](#).  
**prop**: The name of the managed property. |
### Parameter ABNF grammar

<table>
<thead>
<tr>
<th>Parameter ABNF grammar</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parameter = prop &quot;,.fallback0.field&quot; ftype fprop prop = name ftype = &quot;bsrc&quot; / &quot;bsum&quot; fprop = name name = 1*(DIGIT / ALPHA / &quot;[&quot; / &quot;]&quot; / &quot;_&quot; / &quot;.&quot;)</td>
<td>One entry MUST appear in the file for each managed property that supports dynamic document summary. This corresponds to managed properties in the index schema that has <strong>SummaryType</strong> set to &quot;Dynamic&quot;, according to section 1.3.2.1. <em>prop</em>: The name of the managed property. <em>fprop</em>: The name of the fallback managed property. <em>ftype</em>: MUST be &quot;bsrc&quot; if the fallback managed property is the managed property itself. MUST be &quot;bsum&quot; if the fallback managed property is another managed property.</td>
</tr>
<tr>
<td>parameter = prop &quot;,.fallback0.when&quot; atype prop = 1*(DIGIT / ALPHA) atype = always / always_process</td>
<td>One entry MUST appear in the file for each managed property that supports dynamic document summary. This corresponds to managed properties in the index schema that has <strong>SummaryType</strong> set to &quot;Dynamic&quot;, according to section 1.3.2.1. <em>prop</em>: The name of the managed property. <em>atype</em>: MUST be &quot;always_process&quot; if the fallback managed property is the managed property itself. MUST be &quot;always&quot; if the fallback managed property is another managed property.</td>
</tr>
<tr>
<td>parameter = &quot;attributevectors.disable&quot; latentprops latentprops = 1*latentprop <em>(;&quot;latentprop&quot;) latentprop = ltype prop ltype = &quot;bavn&quot; / &quot;batv&quot; prop = 1</em>(DIGIT / ALPHA / &quot;[&quot; / &quot;]&quot; / &quot;_&quot; / &quot;.&quot;)</td>
<td>A list of latent attribute vectors. The parameter MUST be present in the file even if no latent attribute vectors are configured. One <strong>latentprop</strong> entry MUST be present with <strong>ltype</strong>=&quot;bavn&quot; for each refiner specified as latent. For more information, see section 1.3.2.3. One <strong>latentprop</strong> entry MUST be present with <strong>ltype</strong> set to &quot;batv&quot; for each managed property specified as latent sortable. For more information, see section 1.3.2.1. <em>prop</em>: The name of the associated managed property.</td>
</tr>
</tbody>
</table>

#### 2.8 indexConfig.xml

The following table provides information about the file.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Middleware Protocol storage path</td>
<td>RTSearch/webcluster/</td>
</tr>
<tr>
<td>Type of data</td>
<td>Configuration information derived from index schema. Implementation-specific configuration information.</td>
</tr>
<tr>
<td>File format</td>
<td>XML schema file.</td>
</tr>
</tbody>
</table>

This file contains search-related configuration parameters derived from the index schema. The file is the basis for mapping of managed properties into FIXML objects and MUST be consistent with the corresponding configuration settings in the configuration files index.cf and rank.cf.

The file MUST contain the following DOCTYPE specification.

```xml
<!DOCTYPE FastIndexingConfig SYSTEM "http://www.fast.no/DTD/fixconf5_2.dtd">
```
The document type definition (DTD) reference MUST be ignored.

2.8.1 Global Elements

2.8.1.1 FastIndexingConfig

The `FastIndexingConfig` element is the root element.

```xml
<xs:element name="FastIndexingConfig" type="CT_FastIndexingConfig"/>
```

2.8.2 Global Attributes

None.

2.8.3 Complex Types

2.8.3.1 CT_FastIndexingConfig

Referenced by: `FastIndexingConfig`

A complex type that specifies the root element of the index configuration.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_FastIndexingConfig">
  <xs:sequence>
    <xs:element name="catalogList" type="CT_catalogList"/>
    <xs:element name="defaultIndex" type="CT_defaultIndex"/>
    <xs:element name="staticRankClassList" type="CT_staticRankClassList"/>
    <xs:element name="rankProfileList" type="CT_rankProfileList"/>
    <xs:element name="attributeVectorList" type="CT_attributeVectorList"/>
    <xs:element name="summaryClassList" type="CT_summaryClassList"/>
    <xs:element name="summaryFieldOverrideList" type="CT_summaryFieldOverrideList"/>
  </xs:sequence>
</xs:complexType>
```

catalogList: A `CT_catalogList` element.
defaultIndex: A `CT_defaultIndex` element.
staticRankClassList: A `CT_staticRankClassList` element.
rankProfileList: A `CT_rankProfileList` element.
attributeVectorList: A `CT_attributeVectorList` element.
summaryClassList: A `CT_summaryClassList` element.
summaryFieldOverrideList: A `CT_summaryFieldOverrideList` element.

Attributes: None.
2.8.3.2  CT_catalogList

Referenced by: CT_FastIndexingConfig

A complex type that is a container for context catalog elements. For more information about the specification of the context catalogs that MUST be present, see section 2.8.5.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_catalogList">
  <xs:sequence>
    <xs:element maxOccurs="unbounded" name="catalog" type="CT_catalog"/>
  </xs:sequence>
</xs:complexType>
```

catalog: A CT_catalog element.

Attributes: None.

2.8.3.3  CT_catalog

Referenced by: CT_catalogList

A complex type that specifies a context catalog. This is an index structure that represents a particular view of the searchable content.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_catalog">
  <xs:sequence>
    <xs:element maxOccurs="unbounded" name="context" type="CT_context"/>
    <xs:element maxOccurs="unbounded" name="index" type="CT_index"/>
  </xs:sequence>
  <xs:attribute name="name" type="xs:string" use="required"/>
  <xs:attribute name="type" type="ST_catalogType" use="required"/>
  <xs:attribute name="synthetic" type="ST_yesno" use="required"/>
  <xs:attribute name="wildcard" type="ST_yesno" use="required"/>
</xs:complexType>
```

context: A CT_context element that specifies a property context.

index: A CT_index element that specifies a property index.

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Specifies the name of the context catalog. Refer to section 2.8.5 for details about context catalog types and associated names.</td>
</tr>
<tr>
<td>type</td>
<td>An ST_catalogType attribute that specifies the type of the context catalog.</td>
</tr>
</tbody>
</table>
| synthetic | yes: Specifies that this is a synthetic context catalog.  
          | no: Specifies that this is not a synthetic context catalog. |
| wildcard | Specifies that wildcard search is enabled for this context catalog. MUST be set as specified in |
Name | Description
--- | ---
section 2.8.5.

2.8.3.4 **CT_context**

Referenced by: **CT_catalog**

A complex type that specifies a property context. This is the representation of a managed property or an internal property inside the index data structures.

For more information about element and attribute settings for various property contexts, see section 2.8.5.

This element is defined as follows:

```xml
<xs:complexType name="CT_context">
  <xs:attribute name="name" type="xs:string" use="required"/>
  <xs:attribute name="type" type="ST_contextType" use="required"/>
</xs:complexType>
```

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>The name of the property context.</td>
</tr>
<tr>
<td>type</td>
<td>An <strong>ST_contextType</strong> attribute that specifies the property context type associated with how to apply <strong>occurrence boost</strong> for this property context.</td>
</tr>
</tbody>
</table>

2.8.3.5 **CT_index**

Referenced by: **CT_catalog**

A complex type that specifies one property index for the context catalog.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_index">
  <xs:sequence>
    <xs:element maxOccurs="8" name="contextRef" type="CT_contextRef"/>
    <xs:element minOccurs="0" name="alias" type="CT_alias"/>
  </xs:sequence>
  <xs:attribute name="name" type="xs:string" use="required"/>
  <xs:attribute name="subStringSearch" type="ST_SubstringRange" use="optional"/>
  <xs:attribute name="phraseIndex" type="ST_alwaysOff" use="required"/>
  <xs:attribute name="posIndex" type="ST_onoff" use="required"/>
  <xs:attribute name="prefixSearch" type="ST_alwaysOff" use="required"/>
  <xs:attribute name="drillSubIndex" type="xs:string" use="optional"/>
</xs:complexType>
```

**contextRef**: A **CT_contextRef** element.

**alias**: A **CT_alias** element.
Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>The name of the property index.</td>
</tr>
<tr>
<td>subStringSearch</td>
<td>An <code>ST_substringRange</code> attribute that specifies substring search support.</td>
</tr>
<tr>
<td>prefixSearch</td>
<td>Not used; MUST be set to &quot;off&quot;.</td>
</tr>
<tr>
<td>phraseIndex</td>
<td>Not used; MUST be set to &quot;off&quot;.</td>
</tr>
<tr>
<td>posIndex</td>
<td>Enables or disables position index. Disabling position index is set to &quot;off&quot; and implies that no proximity operators or proximity ranking can be applied to a query.</td>
</tr>
<tr>
<td>drillSubIndex</td>
<td>The name of the next sub-index level when performing drilling.</td>
</tr>
</tbody>
</table>

2.8.3.6 CT_contextRef

Referenced by: CT_index

A complex type that specifies a property context included in this property index.

This complex type is defined as follows:

```
<xs:complexType name="CT_contextRef">
  <xs:attribute name="name" type="xs:string" use="required"/>
</xs:complexType>
```

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>The name of the property context.</td>
</tr>
</tbody>
</table>

2.8.3.7 CT_alias

Referenced by: CT_index

A complex type that specifies the name of the index alias for this property index.

This complex type is defined as follows:

```
<xs:complexType name="CT_alias">
  <xs:attribute name="name" type="xs:string" use="optional"/>
</xs:complexType>
```

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>The name of the index alias.</td>
</tr>
</tbody>
</table>
2.8.3.8  CT_defaultIndex

Referenced by: CT_FastIndexingConfig

A complex type that specifies the default index for search queries. The default index is used when no property index is supplied as a parameter to the search query or if the property index does not exist.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_defaultIndex">
    <xs:attribute name="indexName" type="xs:string" use="required"/>
    <xs:attribute name="catalogName" type="xs:string" use="required"/>
</xs:complexType>
```

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>indexName</td>
<td>The name of the property index for the default index.</td>
</tr>
<tr>
<td>catalogName</td>
<td>The name of context catalog for the default index.</td>
</tr>
</tbody>
</table>

2.8.3.9  CT_staticRankClassList

Referenced by: CT_FastIndexingConfig

A complex type that specifies an implementation-specific parameter set. Attribute values MUST be set according to fixed values given in the XML schema.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_staticRankClassList">
    <xs:sequence>
        <xs:element name="staticRankClass" type="CT_staticRankClass"/>
    </xs:sequence>
    <xs:attribute name="bitsUsedForId" type="ST_alwaysZero" use="required"/>
</xs:complexType>
```

```xml
<xs:complexType name="CT_staticRankClass">
    <xs:sequence>
        <xs:element name="rankField" type="CT_rankField"/>
    </xs:sequence>
    <xs:attribute name="name" type="ST_dummy" use="required"/>  
</xs:complexType>
```

2.8.3.10  CT_rankProfileList

Referenced by: CT_FastIndexingConfig

A complex type that is a container for rank profiles. The first rank profile in the list is the default rank profile. The default rank profile is used if a search query does not specify a rank profile.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_rankProfileList">
    <xs:sequence>
        <xs:element name="rankProfile" type="CT_rankProfile"/>
    </xs:sequence>
    <xs:attribute name="name" type="ST_dummy" use="required"/>  
</xs:complexType>
```
rankProfile: A CT_rankProfile element.

Attributes: None.

2.8.3.11 CT_rankProfile

Referenced by: CT_FastIndexingConfig

A complex type that specifies how relevance ranking of a query result will be performed.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_rankProfile">
  <xs:sequence>
    <xs:element name="staticRankParameters" type="CT_staticRankParameters"/>
    <xs:element name="dynamicRankParameters" type="CT_dynamicRankParameters"/>
    <xs:element name="freshnessBoostParameters" type="CT_freshnessBoostParameters" minOccurs="0"/>
  </xs:sequence>
  <xs:attribute name="name" type="xs:string" use="required"/>
  <xs:attribute name="tuneFactor" type="ST_tuneFactor" use="required"/>
  <xs:attribute name="tuneBias" type="ST_alwaysZero" use="required"/>
</xs:complexType>
```

staticRankParameters: A CT_staticRankParameters element.

dynamicRankParameters: A CT_dynamicRankParameters element.

freshnessBoostParameters: A CT_freshnessBoostParameters element. MUST be included only if the index schema specifies freshness boost.

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>The name of the rank profile.</td>
</tr>
<tr>
<td>tuneFactor</td>
<td>An ST_tuneFactor attribute that is not used but MUST be set to 1.00.</td>
</tr>
<tr>
<td>tuneBias</td>
<td>An ST_alwaysZero attribute that is not used but MUST be set to 0.</td>
</tr>
</tbody>
</table>

2.8.3.12 CT_staticRankParameters

Referenced by: CT_rankProfile

A complex type that specifies how to calculate the static rank part of the total rank score.

This complex type is defined as follows:
qualityComponentList: A CT_qualityComponentList element.
Attributes: None.

2.8.3.13 CT_qualityComponentList

Referenced by: CT_staticRankParameters

A complex type that is a container for qualityComponent elements.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_qualityComponentList">
  <xs:sequence>
    <xs:element maxOccurs="unbounded" name="qualityComponent" type="CT_qualityComponent"/>
  </xs:sequence>
</xs:complexType>
```

qualityComponent: A CT_qualityComponent element.
Attributes: None.

2.8.3.14 CT_qualityComponent

Referenced by: CT_qualityComponentList

A complex type that specifies the quality part of the static rank. It refers to an attribute vector that contains a static rank point for each item and a coefficient that reflects the importance of this qualityComponent element. By using a set of qualityComponent elements, an implementer can change the static rank behavior by modifying the coefficients for the various components.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_qualityComponent">
  <xs:attribute name="attributeVector" type="xs:string" use="required"/>
  <xs:attribute name="coefficient" type="xs:decimal" use="required"/>
</xs:complexType>
```

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>attributeVector</td>
<td>The attribute vector that contains the static rank for each item.</td>
</tr>
<tr>
<td>coefficient</td>
<td>The static rank coefficient factor for this quality component.</td>
</tr>
</tbody>
</table>
2.8.3.15  CT_dynamicRankParameters

Referenced by: CT_rankProfile

A complex type that specifies dynamic rank configuration. Dynamic rank parameters give an item a dynamic rank value with respect to a query. The dynamic rank value for a query is the sum of the dynamic rank value for each token.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_dynamicRankParameters">
  <xs:sequence>
    <xs:element name="catalogRankList" type="CT_catalogRankList"/>
  </xs:sequence>
  <xs:attribute name="binLow" type="xs:unsignedInt" use="required"/>
  <xs:attribute name="binHigh" type="xs:unsignedInt" use="required"/>
  <xs:attribute name="binSize" type="xs:decimal" use="required"/>
  <xs:attribute name="posBinLow" type="xs:unsignedInt" use="required"/>
  <xs:attribute name="posBinHigh" type="xs:unsignedInt" use="required"/>
  <xs:attribute name="posBinSize" type="xs:decimal" use="required"/>
  <xs:attribute name="xNearPosBinLow" type="xs:unsignedInt" use="required"/>
  <xs:attribute name="xNearPosBinHigh" type="xs:unsignedInt" use="required"/>
  <xs:attribute name="xNearPosBinSize" type="xs:decimal" use="required"/>
  <xs:attribute name="rankCutoff" type="xs:unsignedByte" use="required"/>
  <xs:attribute name="rankCutoffAdvVal" type="xs:unsignedByte" use="required"/>
  <xs:attribute name="firstOccProximity" type="ST_yesno" use="required"/>
  <xs:attribute name="proximity" type="ST_yesno" use="required"/>
  <xs:attribute name="phraseProximity" type="ST_yesno" use="required"/>
  <xs:attribute name="proximityPairBeforeFirstOccProximityTriple" type="ST_yesno" use="required"/>
  <xs:attribute name="proximityTripleBeforeFirstOccProximityQuad" type="ST_yesno" use="required"/>
  <xs:attribute name="clampStaticRank" type="ST_yesno" use="required"/>
</xs:complexType>
```

catalogRankList: A CT_catalogRankList element.

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>binLow</td>
<td>MUST be set to 0.</td>
</tr>
<tr>
<td>binHigh</td>
<td>MUST be the StopWordThreshold parameter for this rank profile from the index schema, as specified in section 1.3.2.4.</td>
</tr>
<tr>
<td>binSize</td>
<td>MUST be set to 4294967295.00.</td>
</tr>
<tr>
<td>posBinLow</td>
<td>MUST be set to 0.</td>
</tr>
<tr>
<td>posBinHigh</td>
<td>MUST be the PositionStopWordThreshold parameter for this rank profile from the index schema, as specified in section 1.3.2.4.</td>
</tr>
<tr>
<td>posBinSize</td>
<td>MUST be set to 4294967295.00.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>xNearPosBinLow</td>
<td>MUST be set to 0.</td>
</tr>
<tr>
<td>xNearPosBinHigh</td>
<td>For words specified directly within an <strong>ONEAR</strong> or <strong>NEAR proximity search</strong> operator, calculate the number of items plus number of positions for the word. If xNearPosBinHigh is lower than this number, the proximity search operation is performed as an <strong>AND</strong> query operation instead of a proximity search to optimize performance.</td>
</tr>
<tr>
<td>xNearPosBinsize</td>
<td>MUST be set to 4294967295.00.</td>
</tr>
<tr>
<td>superiorBoost</td>
<td>MUST be set to 0.</td>
</tr>
<tr>
<td>rankCutoff</td>
<td>MUST be set to 0.</td>
</tr>
<tr>
<td>rankCutoffAdvVal</td>
<td>MUST be set to 0.</td>
</tr>
<tr>
<td>firstOccProximity</td>
<td>If set to &quot;yes&quot;, use of first occurrence proximity is enabled. If set to &quot;no&quot;, use of first occurrence proximity is disabled. Default value is &quot;yes&quot;.</td>
</tr>
<tr>
<td>Proximity</td>
<td>MUST be &quot;yes&quot;. Specifies that proximity ranking will be applied.</td>
</tr>
<tr>
<td>phraseProximity</td>
<td>MUST be &quot;yes&quot;. Specifies that phrase-specific proximity ranking will be applied.</td>
</tr>
<tr>
<td>proximityPairBeforeFirstOccProximityTriple</td>
<td>MUST be &quot;yes&quot;. Implementation-specific parameter associated with proximity evaluation.</td>
</tr>
<tr>
<td>proximityTripleBeforeFirstOccProximityQuad</td>
<td>MUST be &quot;yes&quot;. Implementation-specific parameter associated with proximity evaluation.</td>
</tr>
<tr>
<td>clampStaticRank</td>
<td>MUST be set to &quot;no&quot;.</td>
</tr>
</tbody>
</table>

### 2.8.3.16  **CT_catalogRankList**

Referenced by: **CT_dynamicRankParameters**

A complex type that specifies detailed rank parameters for the context catalogs that support ranking.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_catalogRankList">
  <xs:sequence>
    <xs:element maxOccurs="unbounded" name="extNumOccBoostOnlyCatalog" type="CT_extNumOccBoostOnlyCatalog"/>
    <xs:element name="rankedCatalog" type="CT_rankedCatalog"/>
  </xs:sequence>
</xs:complexType>
```

**extNumOccBoostOnlyCatalog**: A **CT_extNumOccBoostOnlyCatalog** element.

**rankedCatalog**: A **CT_rankedCatalog** element.
Attributes: None.

2.8.3.17 CT_extNumOccBoostOnlyCatalog

Referenced by: CT_catalogRankList

A complex type that specifies a context catalog that supports un-normalized external occurrence boost values to the query.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_extNumOccBoostOnlyCatalog">
  <xs:attribute name="catalogName" type="xs:string" use="required"/>
  <xs:attribute name="fileName" type="xs:string" use="required"/>
</xs:complexType>
```

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>catalogName</td>
<td>The name of the context catalog.</td>
</tr>
<tr>
<td>fileName</td>
<td>The name of the boost table file associated with this context catalog. The boost table file MUST be, according to section 2.13.1, boost table format &lt;model&gt;_ctx_extnumoccboost.tbl. If no boosting will be applied, the value of fileName MUST be &quot;NULL&quot;.</td>
</tr>
</tbody>
</table>

2.8.3.18 CT_rankedCatalog

Referenced by: CT_catalogRankList

A complex type that is a container for boost elements associated with context catalogs for a full-text index field.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_rankedCatalog">
  <xs:attribute name="catalogName" type="xs:string" use="required"/>
  <xs:sequence>
    <xs:element name="andBoost" type="CT_boostValue"/>
    <xs:element name="orBoost" type="CT_boostValue"/>
    <xs:element name="phraseBoost" type="CT_boostValue"/>
    <xs:element name="rankBoost" type="CT_boostValue"/>
    <xs:element name="anyBoost" type="CT_boostValue"/>
    <xs:element name="nearBoost" type="CT_boostValue"/>
    <xs:element name="orderedNearBoost" type="CT_boostValue"/>
    <xs:element name="firstOccBoost" type="CT_occBoost"/>
    <xs:element name="extNumOccBoost" type="CT_occBoost"/>
    <xs:element maxOccurs="unbounded" name="proximityBoost" type="CT_proximityBoost"/>
    <xs:element name="divTableBoost" type="CT_divTableBoost"/>
    <xs:element name="contextBoostList" type="CT_contextBoostList"/>
  </xs:sequence>
</xs:complexType>
```
andBoost: A CT_boostValue element that specifies rank boost value for the AND operator.

orBoost: A CT_boostValue element that specifies rank boost value for the OR operator.

phraseBoost: A CT_boostValue element that specifies rank boost value for the PHRASE operator.

rankBoost: A CT_boostValue element that specifies rank boost value for the RANK operator.

anyBoost: A CT_boostValue element that specifies rank boost value for the ANY operator.

nearBoost: A CT_boostValue element that specifies rank boost value for the NEAR operator.

orderedNearBoost: A CT_boostValue element that specifies rank boost value for the ONEAR operator.

numOccBoost: A CT_occBoost element that specifies the occurrence boost values associated with the number of occurrences of the query term in the item.

firstOccBoost: A CT_occBoost element that specifies the first-occurrence boost value associated with the position of the first occurrences of the query term in the item.

dextNumOccBoost: A CT_occBoost element that specifies the external occurrence boost value associated with the number of occurrences of the query term in the external property contexts of the item.

proximityBoost: A CT_proximityBoost element.

divTableBoost: A CT_divTableBoost element.

contextBoostList: A CT_contextBoostList element.

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>catalogName</td>
<td>The name of the context catalog.</td>
</tr>
</tbody>
</table>

2.8.3.19 CT_boostValue

Referenced by: CT_rankedCatalog

A complex type that specifies the boost values for a particular query operator when it is used on terms in this context catalog. If the referring element is not present, the default boost value is 0.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_boostValue">
  <xs:attribute name="value" type="xs:unsignedInt" use="required"/>
</xs:complexType>
```

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>An integer boost value.</td>
</tr>
</tbody>
</table>
### 2.8.3.20 CT_occBoost

Referenced by: CT_rankedCatalog

A complex type that specifies a file used to retrieve boost values for term-occurrence-related boosting. This is a **normalized occurrence boost**.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_occBoost">
  <xs:attribute name="fileName" type="xs:string" use="required"/>
</xs:complexType>
```

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fileName</td>
<td>The name of the boost table file associated with this context catalog. The boost table file MUST be, according to section 2.13.1, boost table format <code>&lt;model&gt;_&lt;ctxt&gt;_numocccboost.tbl</code>.</td>
</tr>
</tbody>
</table>

### 2.8.3.21 CT_proximityBoost

Referenced by: CT_rankedCatalog

A complex type that specifies the boost value associated with the correlation between positions of the occurrences in an item for word pairs. There cannot be two proximityBoost elements with same firstOcc and direction attribute values within the same rankedCatalog element.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_proximityBoost">
  <xs:attribute name="fileName" type="xs:string" use="required"/>
  <xs:attribute name="tableSet" type="xs:unsignedByte" use="required"/>
  <xs:attribute name="firstOcc" type="ST_yesno" use="required"/>
  <xs:attribute name="direction" type="ST_direction" use="required"/>
</xs:complexType>
```

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fileName</td>
<td>The name of the boost table file associated with this context catalog. The boost table file MUST be according to section 2.13.2. If no boosting will be applied, the value of fileName MUST be &quot;NULL&quot;.</td>
</tr>
<tr>
<td>firstOcc</td>
<td>The proximity boost scope: &lt;strong&gt;yes&lt;/strong&gt;: The table SHOULD be used for the boosting associated with the first occurrence in an item for two words. &lt;strong&gt;no&lt;/strong&gt;: The table SHOULD be used for the boosting associated with all occurrences in an item for two words.</td>
</tr>
<tr>
<td>direction</td>
<td><strong>forward</strong>: Forward proximity boost based on boost table, as specified in the file to which the fileName attribute refers. <strong>backward</strong>: Backward proximity boost based on boost table, as specified in the file that the</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>fileName</td>
<td>attribute references.</td>
</tr>
<tr>
<td>tableSet</td>
<td>Not used. MUST be set to 0.</td>
</tr>
</tbody>
</table>

### 2.8.3.22 CT_divTableBoost

Referenced by: **CT_rankedCatalog**

A complex type that specifies a division table associated with global frequency of a particular term.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_divTableBoost">
  <xs:attribute name="fileName" type="xs:string" use="required"/>
</xs:complexType>
```

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fileName</td>
<td>The name of the boost table file associated with this context catalog. The boost table file MUST be according to section 2.13.3. If no boosting will be applied, the value of fileName MUST be &quot;NULL&quot;.</td>
</tr>
</tbody>
</table>

### 2.8.3.23 CT_freshnessBoostParameters

Referenced by: **CT_rankProfile**

A complex type that specifies parameters associated with freshness boost that adds a rank score boost to an item that is based on its age.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_freshnessBoostParameters">
  <xs:sequence>
    <xs:element name="freshnessBoostFileRef" type="CT_freshnessBoostFileRef"/>
    <xs:element name="freshnessBoostDateTimeResolution" type="CT_freshnessBoostDateTimeResolution"/>
    <xs:element name="freshnessBoostCoefficient" type="CT_freshnessBoostCoefficient"/>
  </xs:sequence>
</xs:complexType>
```

**freshnessBoostFileRef**: A **CT_freshnessBoostFileRef** element.

**freshnessBoostDateTimeResolution**: A **CT_freshnessBoostDateTimeResolution** element.

**freshnessBoostCoefficient**: A **CT_freshnessBoostCoefficient** element.

Attributes: None.
2.8.3.24  CT_freshnessBoostFileRef

Referenced by: CT_freshnessBoostParameters

A complex type that specifies a reference to an attribute vector used for freshness boost evaluation.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_freshnessBoostFileRef">
  <xs:attribute name="name" type="xs:string" use="required"/>
</xs:complexType>
```

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of an attribute vector of type <code>datetime</code>, whose format is &quot;batv&lt;managed property name&gt;&quot;.</td>
</tr>
</tbody>
</table>

2.8.3.25  CT_freshnessBoostDateTimeResolution

Referenced by: CT_freshnessBoostParameters

A complex type that specifies valid `datetime` resolution for freshness relevance boost.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_freshnessBoostDateTimeResolution">
  <xs:attribute name="value" type="ST_freshnessBoostDateTimeResolution" use="required"/>
</xs:complexType>
```

2.8.3.26  CT_freshnessBoostCoefficient

Referenced by: CT_freshnessBoostParameters

A complex type that specifies the freshness boost coefficient. Multiply the freshness boost value with the coefficient when calculating the total freshness boost value. If coefficient value 0 is used, no freshness boost value is computed or added.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_freshnessBoostCoefficient">
  <xs:attribute name="value" type="xs:unsignedByte" use="required"/>
</xs:complexType>
```

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>The multiplication coefficient for freshness boost.</td>
</tr>
</tbody>
</table>
2.8.3.27  CT_contextBoostList

Referenced by: CT_rankedCatalog

A complex type that is a container for context boost elements.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_contextBoostList">
  <xs:sequence>
    <xs:element maxOccurs="unbounded" name="contextBoost" type="CT_contextBoost"/>
  </xs:sequence>
</xs:complexType>
```

contextBoost: A CT_contextBoost element.

Attributes: None.

2.8.3.28  CT_contextBoost

Referenced by: CT_contextBoost

A complex type that specifies the context boost parameters for the rank profile. This element increases the ranking result for all result pages whose query terms exist in this property context. The value attribute is normalized with respect to term frequency within a column; the pairValue, tripleValue, and quadValue attributes are not.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_contextBoost">
  <xs:attribute name="contextName" type="xs:string" use="required"/>
  <xs:attribute name="value" type="xs:unsignedInt" use="required"/>
  <xs:attribute name="pairValue" type="xs:unsignedInt" use="required"/>
  <xs:attribute name="tripleValue" type="xs:unsignedInt" use="required"/>
  <xs:attribute name="quadValue" type="xs:unsignedInt" use="required"/>
</xs:complexType>
```

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contextName</td>
<td>The name of the property context.</td>
</tr>
<tr>
<td>value</td>
<td>Boost occurrences in this property context with value.</td>
</tr>
<tr>
<td>pairValue</td>
<td>Boost pair of occurrences in this property context with pairValue when evaluating two words from the same context catalog in parallel.</td>
</tr>
<tr>
<td>tripleValue</td>
<td>Boost triple of occurrences in this property context with tripleValue when evaluating three words from the same context catalog in parallel.</td>
</tr>
<tr>
<td>quadValue</td>
<td>Boost quad of occurrences in this property context with quadValue when evaluating four words from the same context catalog in parallel.</td>
</tr>
</tbody>
</table>
2.8.3.29 CT_attributeVectorList

Referenced by: CT_FastIndexingConfig

A complex type that specifies zero or more attribute vectors. Each attribute vector represents a full-text sort or query refinement view of a managed property.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_attributeVectorList">
  <xs:sequence>
    <xs:element minOccurs="0" maxOccurs="unbounded" name="attributeVector" type="CT_attributeVector"/>
  </xs:sequence>
</xs:complexType>
```

**attributeVector:** A CT_attributeVector element.

Attributes: None.

2.8.3.30 CT_attributeVector

Referenced by: CT_attributeVectorList

A complex type that specifies an attribute vector.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_attributeVector">
  <xs:attribute name="name" type="xs:string" use="required"/>
  <xs:attribute name="type" type="ST_attributeTypes" use="required"/>
  <xs:attribute name="multi" type="ST_yesno" use="required"/>
  <xs:attribute name="signedValue" type="ST_yesno" use="required"/>
  <xs:attribute name="alphaSortPath" type="xs:string" use="optional"/>
  <xs:attribute name="alphaSortMasterFile" type="xs:string" use="optional"/>
</xs:complexType>
```

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>The name of the attribute vector.</td>
</tr>
<tr>
<td></td>
<td><strong>batv</strong>&lt;managed property name&gt;: Attribute vector for managed property</td>
</tr>
<tr>
<td></td>
<td>sorting.</td>
</tr>
<tr>
<td></td>
<td><strong>bavn</strong>&lt;managed property name&gt;: Attribute vector for query refinement.</td>
</tr>
<tr>
<td>type</td>
<td>An ST_attributeTypes attribute that specifies the data type for this</td>
</tr>
<tr>
<td></td>
<td>attribute vector.</td>
</tr>
<tr>
<td>multi</td>
<td><strong>yes:</strong> The managed property associated with the attribute vector is</td>
</tr>
<tr>
<td></td>
<td>multi-valued.</td>
</tr>
<tr>
<td></td>
<td><strong>no:</strong> The managed property associated with the attribute vector is</td>
</tr>
<tr>
<td></td>
<td>single-valued.</td>
</tr>
<tr>
<td>signedValue</td>
<td>A Boolean value that specifies whether the values in this attribute vector</td>
</tr>
<tr>
<td></td>
<td>are signed. This value is relevant only for <strong>int64</strong> vectors.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>alphaSortPath</td>
<td>Represents the path to optional alpha sort files for this string attribute vector. Changing the alpha sort configuration has no impact on the protocol, as specified in [MS-FSDQE], except to change the sorting sequence. However, this implementation-specific parameter enables a protocol server to handle custom configuration of result sorting that differs from standard ASCII, as specified in [MS-FSDQE].</td>
</tr>
<tr>
<td>alphaSortMasterFile</td>
<td>A configuration-specific alpha sort file used to list one sub-configuration file for each language for alpha sorting based on the selected language in the query, which can contain configuration information for alternative alphanumeric string sorting.</td>
</tr>
</tbody>
</table>

### 2.8.3.31 CT_summaryClassList

Referenced by: **CT_FastIndexingConfig**

A complex type that specifies that a **summary class** list contains one or more summary classes. The list MUST contain at least one **input summary class** and one **output summary class** named `servedcontent`.

A new input summary class MUST be added to `summaryClassList` whenever an index schema change alters the number of fields in the index schema whose **SummaryType** attribute contains a value of "Static" or "Dynamic". The initial input summary class MUST be named `content` and MUST always be present. Subsequently generated input summary classes MUST be named `content<generation>`, where `<generation>` specifies the index schema generation. Summary class named `content` specifies generation 1. Summary class named `content2` specifies generation 2.

The output summary class MUST include all managed properties that query results will provide according to the index schema.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_summaryClassList">
  <xs:sequence>
    <xs:element maxOccurs="unbounded" name="summaryClass" type="CT_summaryClass"/>
  </xs:sequence>
  <xs:attribute name="fieldTypeUsedForId" type="ST_alwaysInteger" use="required"/>
  <xs:attribute name="defaultOutputClassName" type="xs:string" use="required"/>
</xs:complexType>
```

**summaryClass:** A **CT_summaryClass** element.

**Attributes:**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldTypeUsedForId</td>
<td>Specifies the data type of the summary class identifier that is used in summary.cf. MUST be set to &quot;integer&quot;.</td>
</tr>
<tr>
<td>defaultOutputClassName</td>
<td>Specifies the default output summary class used in delivering document summaries, as specified in [MS-FSDQE]. The default output summary class</td>
</tr>
</tbody>
</table>
The summary class specifications MUST contain a `summaryField` element with `name` set to "bsum `<name of managed property>`" for all managed properties specified in the index schema. The summary class specifications MUST contain a `summaryField` element with `name` set to "bsrc `<name of managed property>`" for all managed properties specified in the index schema with `SummaryType` set to "Dynamic".

The summary class specifications with `type` set to "in" MUST contain `summaryField` elements for all managed properties specified in the index schema for the particular index generation, as specified in section 2.8.3.31.

The summary class specification with `type` set to "out" MUST contain `summaryField` elements for all managed properties specified in the index schema for the latest index generation, as specified in section 2.8.3.31.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_summaryClass">
  <xs:sequence>
    <xs:element maxOccurs="unbounded" name="summaryField" type="CT_summaryField"/>
  </xs:sequence>
  <xs:attribute name="name" type="xs:string" use="required"/>
  <xs:attribute name="type" type="ST_summaryClassTypes" use="required"/>
</xs:complexType>
```

**summaryField**: A `CT_summaryField` element.

**Attributes**:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>The name of the summary class.</td>
</tr>
<tr>
<td>type</td>
<td>A <code>ST_summaryClassTypes</code> attribute that specifies the summary class type.</td>
</tr>
</tbody>
</table>

### 2.8.3.33 CT_summaryField

Referenced by: **CT_summaryClass**

A complex type that specifies a document summary associated with the item.

This complex type is defined as follows:
<xs:complexType name="CT_summaryField">
  <xs:attribute name="name" type="xs:string" use="required"/>
  <xs:attribute name="type" type="ST_summaryFieldTypes" use="required"/>
  <xs:attribute name="defaultValue" type="xs:string" use="required"/>
  <xs:attribute name="compression" type="ST_onoff" use="optional"/>
</xs:complexType>

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>The name of the summary field. All summary field names in a summary class MUST be unique.</td>
</tr>
<tr>
<td>type</td>
<td>An ST_summaryFieldTypes attribute that specifies a document summary type.</td>
</tr>
<tr>
<td>compression</td>
<td>Document summary compression:</td>
</tr>
<tr>
<td></td>
<td>on: Zlib deflate compression is applied when the indexing service (see [MS-FSO] section 2.1.1.5) stores the document summary on disk. For details about document summary compression, see [MS-FSIXDS] section 2.1.16.2.</td>
</tr>
<tr>
<td></td>
<td>off: Compression is not applied.</td>
</tr>
<tr>
<td>defaultValue</td>
<td>The default value for the managed property. MUST be set to an empty string (&quot;&quot;).</td>
</tr>
</tbody>
</table>

2.8.3.34 CT_summaryFieldOverrideList

Referenced by: CT_FastIndexingConfig

A complex type that specifies override instructions for the document summary.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_summaryFieldOverrideList">
  <xs:sequence>
    <xs:choice maxOccurs="unbounded">
      <xs:element name="overrideWithRankLog" type="CT_overrideWithRankLog"/>
      <xs:element name="overrideWithDynamicTeaser" type="CT_overrideWithDynamicTeaser"/>
      <xs:element name="overrideWithJuniperLog" type="CT_overrideWithJuniperLog"/>
      <xs:element name="overrideWithDynamicTeaserMetric" type="CT_overrideWithDynamicTeaserMetric"/>
    </xs:choice>
  </xs:sequence>
</xs:complexType>
```

overrideWithRankLog: A CT_overrideWithRankLog element.

overrideWithDynamicTeaser: A CT_overrideWithDynamicTeaser element.

overrideWithJuniperLog: A CT_overrideWithJuniperLog element.

overrideWithDynamicTeaserMetric: A CT_overrideWithDynamicTeaserMetric element.

Attributes: None.
2.8.3.35  CT_overrideWithDynamicTeaser

Referenced by:  CT_summaryFieldOverrideList

A complex type that specifies a dynamic teaser override. If the output summary class contains a managed property with a name specified for summaryFieldName, the resulting document summary delivered on the [MS-FSDQE] interface MUST contain a dynamic teaser object based on query match with the managed property specified in sourceSummaryFieldName.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_overrideWithDynamicTeaser">
  <xs:attribute name="summaryFieldName" type="xs:string" use="required"/>
  <xs:attribute name="sourceSummaryFieldName" type="xs:string" use="required"/>
</xs:complexType>
```

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>summaryFieldName</td>
<td>The name of the document summary for which to generate a dynamic teaser object. The document summary MUST be of type string or longstring, and MUST be named &quot;bsum&lt;managed property name&gt;&quot;.</td>
</tr>
<tr>
<td>sourceSummaryFieldName</td>
<td>The Name of the document summary to be used for creating the dynamic teaser object. The document summary MUST be of type string or longstring, and MUST be named &quot;bsrc&lt;managed property name&gt;&quot;. For more information about the difference between bsum* and bsric document summaries, see section 2.1.2.</td>
</tr>
</tbody>
</table>

2.8.3.36  CT_overrideWithDynamicTeaserMetric

Referenced by:  CT_summaryFieldOverrideList

A complex type that specifies an implementation-specific override for a dynamic teaser. If the output summary class contains a managed property with name as in the summaryFieldName attribute, provide an alternative document summary output containing the quality of the dynamic teaser object based on the sourceSummaryFieldName managed property in the document summary created during indexing. The output is implementation-specific information intended for debugging.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_overrideWithDynamicTeaserMetric">
  <xs:attribute name="summaryFieldName" type="xs:string" use="required"/>
  <xs:attribute name="sourceSummaryFieldName" type="xs:string" use="required"/>
</xs:complexType>
```

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>summaryFieldName</td>
<td>The name of the document summary for which to generate log output for a dynamic teaser object. The document summary MUST be of type string</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>or &quot;longstring&quot;, and MUST be named &quot;bdpm&lt;managed property name&gt;&quot;.</td>
</tr>
<tr>
<td>sourceSummaryFieldName</td>
<td>The name of the managed property with which to override.</td>
</tr>
</tbody>
</table>

### 2.8.3.37 CT_overrideWithRankLog

Referenced by: **CT_summaryFieldOverrideList**

A complex type that specifies an implementation-specific override for a dynamic teaser. If the output summary class contains a managed property with name as given by the `summaryFieldName` attribute, provide an alternative document summary output containing detailed rank log information based on the `sourceSummaryFieldName` attribute in the document summary created during indexing.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_overrideWithRankLog">
  <xs:attribute name="summaryFieldName" type="xs:string" use="required"/>
</xs:complexType>
```

**Attributes:**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>summaryFieldName</td>
<td>The name of the managed property for which to generate log output.</td>
</tr>
</tbody>
</table>

### 2.8.3.38 CT_overrideWithJuniperLog

Referenced by: **CT_summaryFieldOverrideList**

A complex type that specifies an implementation-specific override for a dynamic teaser. If the output summary class contains a managed property with name as given by the `summaryFieldName` attribute, provide an alternative document summary output containing detailed juniper log information based on the `sourceSummaryFieldName` attribute in the document summary created during indexing.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_overrideWithJuniperLog">
  <xs:attribute name="summaryFieldName" type="xs:string" use="required"/>
  <xs:attribute name="sourceSummaryFieldName" type="xs:string" use="required"/>
</xs:complexType>
```

**Attributes:**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>summaryFieldName</td>
<td>The name of the managed property for which to generate log output.</td>
</tr>
<tr>
<td>sourceSummaryFieldName</td>
<td>The name of the managed property with which to override.</td>
</tr>
</tbody>
</table>
2.8.4 Simple Types

2.8.4.1 ST_catalogType

Referenced by: CT_catalog

A simple type that specifies the context catalog type.

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_catalogType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="integer"/>
    <xs:enumeration value="text"/>
  </xs:restriction>
</xs:simpleType>
```

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>integer</td>
<td>The context catalog contains numeric information that is internally represented as integers.</td>
</tr>
<tr>
<td>text</td>
<td>The context catalog contains string data.</td>
</tr>
</tbody>
</table>

2.8.4.2 ST_contextType

Referenced by: CT_context

A simple type that specifies the property context type associated with how to apply occurrence boost for this property context.

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_contextType">
  <xs:restriction base="xs:token">
    <xs:enumeration value="external"/>
    <xs:enumeration value="simple"/>
    <xs:enumeration value="normal"/>
  </xs:restriction>
</xs:simpleType>
```

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>normal</td>
<td>Use the normal occurrence boost for this property context. Do not use the external occurrence boost. For more information about configuring normal occurrence boost, see section 2.8.3.20.</td>
</tr>
<tr>
<td>external</td>
<td>Use external occurrence boost for this property context. For more information about configuring external normal occurrence boost, see section 2.8.3.17.</td>
</tr>
<tr>
<td>simple</td>
<td>Disable occurrence boost for this property context.</td>
</tr>
</tbody>
</table>

2.8.4.3 ST_substringRange

Referenced by: CT_index

A simple type that specifies substring search support.
This simple type is defined as follows:

```xml
<xs:simpleType name="ST_SubstringRange">
  <xs:restriction base="xs:integer">
    <xs:minInclusive value="0"/>
    <xs:maxInclusive value="63"/>
  </xs:restriction>
</xs:simpleType>
```

Valid values are as follows:

- **0**: Substring search not supported for this managed property or full-text index field.
- **1–31**: N-gram value for substring search matching across token boundaries.
- **33–63**: N-gram value for substring search not matching across token boundaries, where the value of N is the specified value minus 32.

For more information about index schema details, see sections 1.3.2.1 and 1.3.2.

### 2.8.4.4 ST_dummy

Referenced by: **CT_staticRankClass**

A simple type that specifies an implementation-specific parameter with a fixed value.

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_dummy">
  <xs:restriction base="xs:string">
    <xs:enumeration value="dummy"/>
  </xs:restriction>
</xs:simpleType>
```

### 2.8.4.5 ST_dummyfield

Referenced by: **CT_rankField**

A simple type that specifies an implementation-specific parameter with a fixed value.

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_dummyfield">
  <xs:restriction base="xs:string">
    <xs:enumeration value="dummyfield"/>
  </xs:restriction>
</xs:simpleType>
```

### 2.8.4.6 ST_alwaysZero

Referenced by: **CT_staticRankClassList, CT_rankField, CT_rankProfile**

A simple type that specifies an implementation-specific parameter with a fixed value.
This simple type is defined as follows:

```xml
<xs:simpleType name="ST_alwaysZero">
  <xs:restriction base="xs:string">
    <xs:enumeration value="0"/>
  </xs:restriction>
</xs:simpleType>
```

### 2.8.4.7 ST_tuneFactor

Referenced by: **CT_rankProfile**

A simple type that specifies an implementation-specific parameter with a fixed value.

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_tuneFactor">
  <xs:restriction base="xs:string">
    <xs:enumeration value="1.00"/>
  </xs:restriction>
</xs:simpleType>
```

### 2.8.4.8 ST_always32

Referenced by: **CT_rankField**

A simple type that specifies an implementation-specific parameter with a fixed value.

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_always32">
  <xs:restriction base="xs:string">
    <xs:enumeration value="32"/>
  </xs:restriction>
</xs:simpleType>
```

### 2.8.4.9 ST_yesno

Referenced by: **CT_catalog, CT_dynamicRankParameters, CT_proximityBoost, CT_attributeVector**

A simple type that specifies the Boolean condition values "yes" and "no".

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_yesno">
  <xs:restriction base="xs:string">
    <xs:enumeration value="yes"/>
    <xs:enumeration value="no"/>
  </xs:restriction>
</xs:simpleType>
```
2.8.4.10  ST_onoff

Referenced by: CT_index, CT_summaryField

A simple type that specifies the Boolean condition values "on" and "off".

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_onoff">
  <xs:restriction base="xs:string">
    <xs:enumeration value="on"/>
    <xs:enumeration value="off"/>
  </xs:restriction>
</xs:simpleType>
```

2.8.4.11  ST_alwaysOff

Referenced by: CT_index

A simple type that specifies an implementation-specific parameter with a fixed value.

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_alwaysOff">
  <xs:restriction base="xs:string">
    <xs:enumeration value="off"/>
  </xs:restriction>
</xs:simpleType>
```

2.8.4.12  ST_direction

Referenced by: CT_proximityBoost

A simple type that specifies a direction.

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_direction">
  <xs:restriction base="xs:string">
    <xs:enumeration value="forward"/>
    <xs:enumeration value="backward"/>
  </xs:restriction>
</xs:simpleType>
```

The following table lists the applicable values.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>forward</td>
<td>Forward direction.</td>
</tr>
<tr>
<td>backward</td>
<td>Backward direction.</td>
</tr>
</tbody>
</table>

[MS-FSSCFG] — v20120630
Search Configuration File Format

Copyright © 2012 Microsoft Corporation.

Release: July 16, 2012
2.8.4.13  ST_freshnessBoostDateTimeResolution

Referenced by: CT_freshnessBoostDateTimeResolution

A simple type that specifies valid **datetime** resolution for freshness relevance boost.

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_freshnessBoostDateTimeResolution">
  <xs:restriction base="xs:string">
    <xs:enumeration value="second"/>
    <xs:enumeration value="minute"/>
    <xs:enumeration value="hour"/>
    <xs:enumeration value="day"/>
    <xs:enumeration value="year"/>
  </xs:restriction>
</xs:simpleType>
```

The following table lists the applicable values.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>second</td>
<td>The <strong>datetime</strong> resolution in seconds.</td>
</tr>
<tr>
<td>minute</td>
<td>The <strong>datetime</strong> resolution in minutes.</td>
</tr>
<tr>
<td>hour</td>
<td>The <strong>datetime</strong> resolution in hours.</td>
</tr>
<tr>
<td>day</td>
<td>The <strong>datetime</strong> resolution in days.</td>
</tr>
<tr>
<td>year</td>
<td>The <strong>datetime</strong> resolution in years.</td>
</tr>
</tbody>
</table>

2.8.4.14  ST_attributeTypes

Referenced by: CT_attributeVector

A simple type that specifies the data type for an attribute vector.

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_attributeTypes">
  <xs:restriction base="xs:token">
    <xs:enumeration value="string"/>
    <xs:enumeration value="int64"/>
  </xs:restriction>
</xs:simpleType>
```

The following table lists the applicable values.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>int64</td>
<td>An attribute vector type for all numeric managed properties.</td>
</tr>
<tr>
<td>string</td>
<td>An attribute vector type for string managed properties.</td>
</tr>
</tbody>
</table>
2.8.4.15  **ST_summaryFieldTypes**

Referenced by: **CT_summaryField**

A simple type that specifies a document summary type, as specified in section 2.1.3.

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_summaryFieldTypes">
  <xs:restriction base="xs:token">
    <xs:enumeration value="string"/>
    <xs:enumeration value="longstring"/>
    <xs:enumeration value="data"/>
  </xs:restriction>
</xs:simpleType>
```

The following table lists the applicable values.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>The length of the document summary string does not exceed 64 kilobytes.</td>
</tr>
<tr>
<td>longstring</td>
<td>The length of the document summary string can exceed 64 kilobytes.</td>
</tr>
<tr>
<td>data</td>
<td>Used only for internal document summary representation inside the index.</td>
</tr>
</tbody>
</table>

2.8.4.16  **ST_summaryClassTypes**

Referenced by: **CT_summaryClass**

A simple type that specifies the type of summary class.

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_summaryClassTypes">
  <xs:restriction base="xs:token">
    <xs:enumeration value="in"/>
    <xs:enumeration value="out"/>
  </xs:restriction>
</xs:simpleType>
```

The following table lists the applicable values.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>in</td>
<td>The input summary class. This is the summary class that represents all managed properties and that is mapped to a document summary.</td>
</tr>
<tr>
<td>out</td>
<td>The output summary class. This represents one summary class used in a query result.</td>
</tr>
</tbody>
</table>

2.8.4.17  **ST_alwaysInteger**

Referenced by: **CT_summaryClassList**

A simple type that specifies an implementation-specific parameter with a fixed value.
This simple type is defined as follows:

```xml
<xs:simpleType name="ST_alwaysInteger">
  <xs:restriction base="xs:token">
    <xs:enumeration value="integer"/>
  </xs:restriction>
</xs:simpleType>
```

2.8.5  Context Catalog Structure

The context catalog structure MUST contain the following catalog elements:

- One catalog sub-element named **bt1**, as specified in section 2.8.5.1.1.
- One catalog sub-element named **bi1**, as specified in section 2.8.5.2.1.
- One catalog sub-element for each **FullTextIndex** element specified in the index schema, as specified in section 1.3.2.2.
- One catalog sub-element named **meta**, as specified in section 2.8.5.1.2.
- One catalog sub-element named **anchortext**, as specified in section 2.8.5.3.2.
- One catalog sub-element named **assocqueries**, as specified in section 2.8.5.3.3.

Context catalogs of type **text** MUST NOT contain more than eight property contexts. Context catalogs of type **integer** and synthetic context catalogs do not have this limitation.

2.8.5.1  Synthetic Context Catalogs

The **catalog** attributes MUST be set to the following values:

- **type**: text
- **synthetic**: yes
- **wildcard**: yes

The **context** attributes MUST be set to the following value:

- **type**: simple

The **index** attributes MUST be set to the following value:

- **posIndex**: on

All index elements within the same synthetic context catalog MUST have the same value for the attributes **posIndex** and **subStringIndex**.

2.8.5.1.1  bt1 Context Catalog

The **bt1** context catalog contains all non-numeric managed properties.

2.8.5.1.2  meta Context Catalog

The **meta** context catalog contains the following metadata internal properties:
• collection
• contentid
• contentids

This catalog element MUST be formatted according to the following XML.

```xml
<catalog name="meta" type="text" synthetic="yes" wildcard="no">
  <context name="collection" type="simple"/>
  <context name="contentid" type="simple"/>
  <context name="contentids" type="simple"/>
  <index name="collection" phraseIndex="off" posIndex="on" prefixSearch="off">
    <contextRef name="collection"/>
  </index>
  <index name="contentid" phraseIndex="off" posIndex="on" prefixSearch="off">
    <contextRef name="contentid"/>
  </index>
  <index name="contentids" phraseIndex="off" posIndex="on" prefixSearch="off">
    <contextRef name="contentids"/>
  </index>
</catalog>
```

2.8.5.2 Numeric Catalogs

The catalog attributes MUST be set to the following values:

• type: integer
• synthetic: no
• wildcard: no

The context attributes MUST be set to the following value:

• type: normal

The index attributes MUST be set to the following value:

• posIndex: on

2.8.5.2.1 bi1 Catalog

The bi1 context catalog contains all numeric managed properties.

2.8.5.3 Ranked Context Catalogs

Ranked context catalogs support queries with dynamic ranking. The catalog attributes MUST be set to the following values:

• type: text
• synthetic: no
• wildcard: yes

The index attributes MUST be set to the following value:
- **posIndex**: on

### 2.8.5.3.1 Full-Text Index Field Context Catalogs

Context catalogs can support queries against full-text index fields. There MUST be one catalog element for each `FullTextIndex` element specified in the index schema. The catalog element for each `FullTextIndex` index schema MUST be formatted according to the following XML, where the full-text index field name is **content**.

```xml
<catalog name="bcatcontent" type="text" synthetic="no" wildcard="yes">
  <context name="bconf1" type="normal"/>
  <context name="bconf2" type="normal"/>
  <context name="bconf3" type="normal"/>
  <context name="bconf4" type="normal"/>
  <context name="bconf5" type="normal"/>
  <context name="bconf6" type="normal"/>
  <context name="bconf7" type="normal"/>
  <context name="bconf8" type="external"/>
  <index name="bidxcontentlvl1" phraseIndex="off" posIndex="on" prefixSearch="off" drillSubIndex="bidxcontentlvl2">
    <contextRef name="bconf1"/>
    <contextRef name="bconf2"/>
    <contextRef name="bconf3"/>
    <contextRef name="bconf4"/>
    <contextRef name="bconf5"/>
    <contextRef name="bconf6"/>
    <contextRef name="bconf7"/>
    <contextRef name="bconf8"/>
    <alias name="content"/>
  </index>
  <index name="bidxcontentlvl2" phraseIndex="off" posIndex="on" prefixSearch="off" drillSubIndex="bidxcontentlvl3">
    <contextRef name="bconf3"/>
    <contextRef name="bconf4"/>
    <contextRef name="bconf5"/>
    <contextRef name="bconf6"/>
    <contextRef name="bconf7"/>
    <contextRef name="bconf8"/>
  </index>
  <index name="bidxcontentlvl3" phraseIndex="off" posIndex="on" prefixSearch="off" drillSubIndex="bidxcontentlvl4">
    <contextRef name="bconf5"/>
    <contextRef name="bconf6"/>
    <contextRef name="bconf7"/>
    <contextRef name="bconf8"/>
  </index>
  <index name="bidxcontentlvl4" phraseIndex="off" posIndex="on" prefixSearch="off">
    <contextRef name="bconf7"/>
    <contextRef name="bconf8"/>
  </index>
</catalog>
```

The **catalog name** attribute MUST use the following naming convention.

```
name="bcat<full-text index field name>"
```
The index name and drillSubIndex attributes MUST follow the naming convention.

```
name="bidx<full-text index field name>lvl<field importance level>"
drillSubIndex="bidx<full-text index field name>lvl<field importance level>"
```

In the preceding syntax, <full-text index field name> is the name of the FullTextIndex element in the index schema, and <field importance level> is the field importance level, as specified in the index schema.

The property contexts are one of eight reserved names: bconf1, bconf2, bconf3, bconf4, bconf5, bconf6, bconf7, and bconf8. These eight property contexts are associated with the field importance level for the managed property. The bconf7 and bconf8 property contexts are included in all field importance levels. The bconf5 and bconf6 property contexts are included in field importance levels 1, 2, and 3. The bconf3 and bconf4 property contexts are included in field importance levels 1 and 2. The bconf1 and bconf2 property contexts are included in field importance level 1.

For more information about mapping of managed properties to property contexts and field importance levels, see section 2.10.

The context and contextRef name attribute MUST specify a valid full-text index context.

### 2.8.5.3.2 anchortext Catalog

The anchortext catalog element MUST be formatted according to the following XML.

```
<catalog name="anchortext" type="text" synthetic="no" wildcard="no">
  <context name="canchortext" type="external"/>
  <index name="complete" phraseIndex="off" posIndex="off" prefixSearch="off">
    <contextRef name="canchortext"/>
  </index>
</catalog>
```

### 2.8.5.3.3 assocqueries Catalog

The assocqueries catalog element MUST be formatted according to the following XML.

```
<catalog name="assocqueries" type="text" synthetic="no" wildcard="no">
  <context name="cassocqueries" type="external"/>
  <index name="complete" phraseIndex="off" posIndex="off" prefixSearch="off">
    <contextRef name="cassocqueries"/>
  </index>
</catalog>
```

### 2.9 index.cf

The following table provides information about the file.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Middleware Protocol storage path</td>
<td>RTSearch/webcluster/</td>
</tr>
<tr>
<td>Type of data</td>
<td>Configuration information derived from index schema.</td>
</tr>
</tbody>
</table>
This file contains detailed configuration of the index structure given a specific index schema. The file content MUST derive from the corresponding values in indexConfig.xml. Section 2.9.2 specifies the detailed requirements.

The configuration data in this file is derived from indexConfig.xml, as specified in section 2.8. The file MAY be ignored, because all the configuration information in index.cf is also in indexConfig.xml.

### 2.9.1 ABNF Grammar

The configuration file MUST be according to the following ABNF grammar. In addition to the grammar, the file format allows blank lines and comment lines that begin with the number sign (#).

```plaintext
index-cf   = *crlf catalog def-index alias [attrv] [drilling]
crlf       = LF / (CR LF)
true-false = "true" / "false"
validname  = 1*(DIGIT / ALPHA / "[" / "]" / "." / ":")
validname-list = (validname *(SP validname))
catalog     = 1*(catalog-ent *crlf)
catalog-ent = catalog-def dictionary [wildcards] catalog-schema 1*crlf
catalog-def = "catalog" SP validname SP "type" SP catalogtype crlf
catalogtype = "text" / (("textsynthetic" / "integer") SP context-number )
context-number = 1*DIGIT
dictionary = "dictionary exact" crlf
wildcards   = "wildcards" crlf
catalog-schema = 1*contexts 1*index-entry
contexts     = context-type SP context-spec crlf
context-type = "contexts" / "externalcontexts" / "simplecontexts"
context-spec = "all" / validname-list
index-entry = index contains mccontexts
index        = "index" SP validname *index-attrs crlf
index-attrs  = SP index-attr
index-attr   = "withprefix" / substring / "nopositions"
substring    = "withsubstring" SP validsubstring
validsubstring = 1*DIGIT
contains     = "contains" SP context-spec crlf
mccontexts  = "mccontext" SP validname crlf
def-index    = "defaultindex" SP validname 1*crlf
alias        = 1*alias-entry 1*crlf
alias-entry  = "alias" SP validname SP alias-index crlf
alias-index  = validname "." validname
attrv        = 1*attr-entry 1*crlf
attr-entry   = "attributevector" SP attr-name SP attr-type SP attr-pars crlf
attr-name    = ("batv" / "bavn") validname
attr-type    = "string" / "int64"
```
2.9.2 Configuration Parameter Details

The configuration parameters in this file are derived from the configuration in indexConfig.xml. The tables throughout this section specify the corresponding parameters in indexConfig.xml.

2.9.2.1 Context Catalog Configuration

The context catalog configuration section specifies context catalog entries that are derived from index schema.

The following table provides syntax details for ABNF rules.

<table>
<thead>
<tr>
<th>ABNF rule</th>
<th>Syntax details</th>
</tr>
</thead>
<tbody>
<tr>
<td>catalog-def</td>
<td>MUST be the value of the name attribute, as specified in section 2.8.3.3.</td>
</tr>
<tr>
<td>catalogtype</td>
<td>The context catalog type. MUST be set according to the value of the type and synthetic attributes, as specified in section 2.8.3.3:</td>
</tr>
<tr>
<td></td>
<td>text: Full-text index field context catalog.</td>
</tr>
<tr>
<td></td>
<td>textsynthetic: Synthetic context catalog.</td>
</tr>
<tr>
<td></td>
<td>integer: Integer context catalog.</td>
</tr>
<tr>
<td>context-number</td>
<td>MUST be the number of property contexts in the context catalog, given by the number of &lt;context&gt; elements in the &lt;catalog&gt; element.</td>
</tr>
<tr>
<td></td>
<td>This parameter applies for context catalogs of type integer and textsynthetic, as specified in indexConfig.xml.</td>
</tr>
<tr>
<td>dictionary</td>
<td>The context dictionary. Type MUST be exact.</td>
</tr>
<tr>
<td>wildcards</td>
<td>MUST be present if wildcard is set to &quot;yes&quot;, as specified in section 2.8.3.3.</td>
</tr>
<tr>
<td>context-type</td>
<td>The value MUST correspond to the value of the type attribute, as specified in section 2.8.3.3:</td>
</tr>
<tr>
<td></td>
<td>contexts: MUST be set if type is set to &quot;normal&quot;.</td>
</tr>
<tr>
<td></td>
<td>simplecontexts: MUST be set if type is set to &quot;simple&quot;.</td>
</tr>
<tr>
<td></td>
<td>externalcontexts: MUST be set if type is set to &quot;external&quot;.</td>
</tr>
<tr>
<td>context-spec</td>
<td>MUST be a list of property context element names in the catalog element, as specified in section 2.8.3.4.</td>
</tr>
<tr>
<td></td>
<td>The &quot;all&quot; value MUST be used for the special catalog named &quot;msyntcat&quot;.</td>
</tr>
<tr>
<td>index</td>
<td>MUST contain a valid property index name, as specified in indexConfig.xml. For more information, see section 2.8.</td>
</tr>
<tr>
<td>index-attr</td>
<td>withprefix: MUST be set if prefixSearch is set, as specified in section 2.8.3.5.</td>
</tr>
<tr>
<td></td>
<td>withsubstring: MUST be set if subStringSearch is set, as specified in section 2.8.3.5.</td>
</tr>
<tr>
<td></td>
<td>nopositions: MUST be set if the posIndex attribute is set to &quot;off&quot;, as specified in section 2.8.</td>
</tr>
</tbody>
</table>
### ABNF rule

<table>
<thead>
<tr>
<th>ABNF rule</th>
<th>Syntax details</th>
</tr>
</thead>
<tbody>
<tr>
<td>contains</td>
<td>A list of property contexts specified in this property index. MUST be the set of contextRef elements within the index element, as specified in section 2.8.3.5.</td>
</tr>
<tr>
<td>mccontexts</td>
<td>Specifies the most common property context. MUST be set to the first property context specified within the &quot;contains&quot; clause.</td>
</tr>
</tbody>
</table>

The following special context catalogs MUST be specified in the file and MUST have the following content.

```
catalog msynthcat type text
dictionary exact
wildcards
contexts all
index all withprefix
contains all
catalog anchortext type text
dictionary exact
externalcontexts canchortext
index complete nopositions
contains canchortext
mccontext canchortext

catalog assocqueries type text
dictionary exact
externalcontexts cassocqueries
index complete nopositions
contains cassocqueries
mccontext cassocqueries

catalog meta type textsynthetic 3
dictionary exact
simplecontexts collection contentid contentids
index collection
contains collection
mccontext collection
index contentid
contains contentid
mccontext contentid
index contentids
contains contentids
mccontext contentids
```

The main synthetic catalog is msynthcat. The anchortext, assocqueries, and meta properties correspond to the definitions in indexConfig.xml.

The other catalog definitions MUST correspond to catalog definitions in indexConfig.xml, as specified in the previous tables.

#### 2.9.2.2 Default Index Configuration

The default index configuration section defines the default index for the particular index schema, as described in the following table.
2.9.2.3  Index Alias Configuration

The index alias configuration section contains a number of property index name aliases derived from the index schema, as described in the following table.

<table>
<thead>
<tr>
<th>ABNF rule</th>
<th>Syntax details</th>
</tr>
</thead>
<tbody>
<tr>
<td>alias-entry</td>
<td>alias &lt;aliasname&gt; &lt;alias-index&gt;</td>
</tr>
<tr>
<td>alias-name</td>
<td>MUST be the name of an alias element in the index element, as specified in</td>
</tr>
<tr>
<td>alias-index</td>
<td>section 2.8.3.5.</td>
</tr>
<tr>
<td>alias-name</td>
<td>MUST be the name attribute of the index element with the specified alias</td>
</tr>
<tr>
<td>alias-index</td>
<td>name.</td>
</tr>
</tbody>
</table>

2.9.2.4  Attribute Vector Configuration

The attribute vector configuration section contains a number of attribute vector tables derived from index schema refiner definitions. One attribute vector definition MUST exist for each attributeVector element specified in indexConfig.xml. The following table provides syntax details for ABNF rules.

<table>
<thead>
<tr>
<th>ABNF rule</th>
<th>Syntax details</th>
</tr>
</thead>
<tbody>
<tr>
<td>attr-name</td>
<td>MUST be a valid attribute vector name, as specified in section 2.8.3.30.</td>
</tr>
<tr>
<td>attr-type</td>
<td>MUST be a valid attribute vector type, as specified in section 2.8.3.30.</td>
</tr>
<tr>
<td>attr-multivalued</td>
<td>&lt;attr-multivalued&gt; &lt;attr-sortsigned&gt;</td>
</tr>
<tr>
<td>attr-multivalued</td>
<td>MUST be the multi attribute, as specified in section 2.8.3.30.</td>
</tr>
<tr>
<td>attr-sortsigned</td>
<td>MUST be the signedVal attribute, as specified in section 2.8.3.30.</td>
</tr>
</tbody>
</table>

2.9.2.5  Drilling Configuration

The drilling configuration section defines the relation between the drilling levels, derived from index schema, as described in the following table.

<table>
<thead>
<tr>
<th>ABNF rule</th>
<th>Syntax details</th>
</tr>
</thead>
<tbody>
<tr>
<td>drill-entry</td>
<td>&lt;from&gt; &lt;to&gt;</td>
</tr>
<tr>
<td>from</td>
<td>MUST be the name of an index element that has a drillSubIndex attribute with</td>
</tr>
<tr>
<td></td>
<td>value &quot;to&quot;.</td>
</tr>
</tbody>
</table>

2.10  fixml_mappings.xml

The following table provides information about the file.
<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Middleware Protocol storage path</td>
<td>RTSearch/webcluster/</td>
</tr>
<tr>
<td>Type of data</td>
<td>Configuration information derived from index schema.</td>
</tr>
<tr>
<td>File format</td>
<td>XML schema file.</td>
</tr>
</tbody>
</table>

This file contains the mapping configuration for FAST Index Markup Language (FIXML) files. The *item processing* service uses this mapping to apply mapping of managed properties into the FIXML object. For more information, refer to [MS-FSFIXML].

### 2.10.1 Global Elements

#### 2.10.1.1 Mappings

The *mappings* element contains a set of managed property mapping specifications for creation of FIXML.

```xml
<xs:element name="mappings" type="CT_mappings"/>
```

### 2.10.2 Global Attributes

None.

### 2.10.3 Complex Types

#### 2.10.3.1 CT_mappings

Referenced by: <mappings>

A complex type that is a container for a set of map elements.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_mappings">
  <xs:sequence>
    <xs:element minOccurs="1" name="map" type="CT_map" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="sclass" type="xs:string" use="required"/>
</xs:complexType>
```

**map**: A CT_map element.

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sclass</td>
<td>An input summary class name. MUST be the name associated with the latest index schema reference, as specified in section 2.8.3.31.</td>
</tr>
</tbody>
</table>
2.10.3.2 **CT_map**

**Referenced by:** **CT_mappings**

A complex type that specifies the mapping configuration for the index field.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_map">
  <xs:sequence minOccurs="0" maxOccurs="1">
    <xs:element name="ignore-value" type="CT_ignore-value"/>
  </xs:sequence>
  <xs:attribute name="type" type="ST_type" use="required"/>
  <xs:attribute name="src" type="xs:string" use="required"/>
  <xs:attribute name="dst" type="xs:string" use="required"/>
  <xs:attribute name="dstcatalog" type="xs:string" use="optional"/>
  <xs:attribute name="maxsize" type="xs:int" use="optional" default="64"/>
  <xs:attribute name="keepbreaks" type="ST_yesno" use="optional"/>
  <xs:attribute name="phrasebreak" type="ST_yesno" use="optional"/>
  <xs:attribute name="fieldseparationlength" type="xs:int" use="optional"/>
  <xs:attribute name="phraseseparator" type="xs:string" use="optional"/>
  <xs:attribute name="multi" type="ST_yesno" use="optional"/>
  <xs:attribute name="defaultvalue" type="xs:string" use="optional"/>
  <xs:attribute name="separator" type="xs:string" use="optional"/>
</xs:complexType>
```

**ignore-value:** A **CT_ignore-value** element. The element MUST be present if the index schema **RefinerConfiguration** attribute **DefaultValue** is present.

**Attributes:**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>type</strong></td>
<td>An <strong>ST_type</strong> attribute that specifies destination index field type in the FIXML object.</td>
</tr>
<tr>
<td><strong>src</strong></td>
<td>The name of the source property within the item processing service. The name MUST be the managed property or internal property name unless otherwise specified in section 2.10.3.2.1.</td>
</tr>
<tr>
<td><strong>dst</strong></td>
<td>The name of destination property in the FIXML object. The name depends on the destination index field type and MUST be set as specified in section 2.10.3.2.1.</td>
</tr>
<tr>
<td><strong>dstcatalog</strong></td>
<td>The destination context catalog for <strong>type</strong> set to &quot;context&quot;. MUST be present if <strong>type</strong> is set to &quot;context&quot;. MUST NOT be present if <strong>type</strong> has any other value. The context catalog name depends on the destination index field type and MUST be set as specified in section 2.10.3.2.1.</td>
</tr>
<tr>
<td><strong>maxsize</strong></td>
<td>The maximum size of the source managed property in kilobytes. Larger managed properties are truncated before being mapped to the FIXML destination element. MUST be present if <strong>type</strong> is set to &quot;context&quot; or &quot;sfield&quot;. MUST NOT be present if <strong>type</strong> has any other value. For <strong>type</strong> set to &quot;context&quot;, <strong>maxsize</strong> MUST be set to the value of the index schema <strong>ManagedProperty</strong> attribute <strong>MaxIndexSize</strong>. For <strong>type</strong> set to &quot;sfield&quot;, <strong>maxsize</strong> MUST be set to the value of the index schema</td>
</tr>
</tbody>
</table>

[MS-FSSCFG] — v20120630
Search Configuration File Format

Copyright © 2012 Microsoft Corporation.

Release: July 16, 2012
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ManagedProperty attribute MaxResultSize.</td>
<td></td>
</tr>
<tr>
<td>keepbreaks</td>
<td>Keeps paragraph and section breaks in the document summary for generation of the dynamic teaser object. MUST be present if type is set to &quot;sfield&quot;. MUST NOT be present if type has any other value. MUST be set to &quot;yes&quot; if the index schema ManagedProperty attribute SummaryType is set to &quot;Dynamic&quot;. MUST be set to &quot;no&quot; if the index schema ManagedProperty attribute SummaryType is set to &quot;Static&quot;.</td>
</tr>
<tr>
<td>phrasebreak</td>
<td>Supports insertion of a phrase break. MUST be present if type is set to &quot;context&quot;. MUST NOT be present if type has any other value. MUST be set to &quot;yes&quot; if the index schema ManagedProperty attribute Type is set to &quot;Text&quot;. MUST be set to &quot;no&quot; if the index schema ManagedProperty attribute Type has any other value.</td>
</tr>
<tr>
<td>phraseseparator</td>
<td>A UTF-8 character that specifies a phrase break in the source property for inserting phrase breaks during indexing. MUST be present if type is set to &quot;context&quot;. MUST NOT be present if type has any other value. MUST be set to a semicolon (;) if the index schema ManagedProperty has IsMultiValued set to &quot;Yes&quot;. MUST be set to an empty string (&quot;&quot;&quot;) if the index schema ManagedProperty has IsMultiValued set to &quot;No&quot;.</td>
</tr>
<tr>
<td>fieldseparationlength</td>
<td>The number of word positions that the proximity distance added between the last word of one managed property and the first word of the next managed property within a full-text index field. MUST be present if type is set to &quot;context&quot;. MUST NOT be present if type has any other value. The value depends on the destination index field type and MUST be set as specified in section 2.10.3.2.1.</td>
</tr>
<tr>
<td>multi</td>
<td>This property supports multi-value strings for attribute vectors. MUST be present if type is set to &quot;attributevector&quot;. MUST NOT be present if type has any other value. The value depends on the destination index field type and MUST be set as specified in section 2.10.3.2.1.</td>
</tr>
<tr>
<td>defaultvalue</td>
<td>MUST be set to the index schema DefaultValue attribute, as specified in section 1.3.2.3.</td>
</tr>
<tr>
<td>separator</td>
<td>A UTF-8 character that specifies multi-value string separation in the source property for attribute vectors. MUST be present if type=&quot;attributevector&quot;. MUST NOT be present if type has any other value. MUST be set to a semicolon (;) if the index schema ManagedProperty has IsMultiValued set to &quot;Yes&quot;. MUST be set to the empty string &quot;&quot; if the index schema ManagedProperty has IsMultiValued set to &quot;No&quot;.</td>
</tr>
</tbody>
</table>
2.10.3.2.1 Map Elements for Managed Properties

The *mappings* element MUST contain the following map elements for each *ManagedProperty* class in the index schema, as specified in section 1.3.2.1. All other attributes of the *map* element MUST be set according to the specification in section 2.10.3.2.

One *map* element for each managed property of type *Text* or *Boolean*, with *Queryable* set to *true*. The *map* element MUST have the following attributes set to the specified values:

- **type**: context
- **dstcatalog**: bt1
- **dst**: bcon<managed property name>
- **fieldseparationlength**: 0

One *map* element for each managed property of type *Integer*, *Decimal*, *Float*, or *Datetime*, with *Queryable* set to *true*. The *map* element MUST have the following attributes set to the specified values:

- **type**: context
- **dstcatalog**: bi1
- **dst**: bcon<managed property name>
- **fieldseparationlength**: 0

One *map* element for each managed property containing a *FullTextIndexMapping* in the index schema. The *map* element MUST have the following attributes set to the specified values:

- **type**: context
- **dstcatalog**: bcat<full-text index field name>
- **dst**: bconf<importance level value>
- **fieldseparationlength**: 256

One *map* element for each managed property with *SummaryType* set to "Static". The *map* element MUST have the following attributes set to the specified values:

- **type**: sfield
- **dst**: bsum<managed property name>

One *map* element for each managed property with *SummaryType* = "Dynamic". The *map* element MUST have the following attributes set to the specified values:

- **type**: sfield
- **src**: res<managed property name>
- **dst**: bsuc<managed property name>

One *map* element for each managed property with *SortableType* = "Enabled". The *map* element MUST have the following attributes set to the specified values:
• **type:** attributevector
• **dst:** batv<managed property name>
• **multi:** no

One map element for each managed property containing a RefinerRef in the index schema. The map element MUST have the following attributes set to the specified values:
• **type:** attributevector
• **dst:** bavn<managed property name>
• **multi:** yes

### 2.10.3.2.2 Map Elements for Internal Properties

The file MUST include the following map elements that are associated with internal properties.

```xml
<map
type="context"
src="canchortext"
dst="canchortext"
dstcatalog="anchortext"/>
<map
type="context"
src="cassocqueries"
dst="cassocqueries"
dstcatalog="assocqueries"/>
```

For more information about internal properties, see section 2.1.5.

### 2.10.3.3 CT_ignore-value

Referenced by: CT_map

A complex type that specifies a value that will be ignored for attribute vectors. The source managed property that contains this value is equivalent to that managed property containing no value.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_ignore-value">
  <xs:attribute name="value" type="xs:string" use="required"/>
</xs:complexType>
```

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>value</strong></td>
<td>A string that specifies a value to ignore when generating attribute vectors for managed properties. The value MUST be the DefaultValue attribute in the RefinerConfiguration index schema, as specified in section 1.3.2.3.</td>
</tr>
</tbody>
</table>
2.10.4 Simple Types

2.10.4.1 ST_yesno

Referenced by: CT_map

A simple type that specifies a Boolean condition "yes" and "no".

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_yesno">
  <xs:restriction base="xs:string">
    <xs:enumeration value="yes"/>
    <xs:enumeration value="no"/>
  </xs:restriction>
</xs:simpleType>
```

2.10.4.2 ST_type

Referenced by: CT_map

A simple type that specifies destination index field type in the FIXML object. For more information about type rules, see section 2.10.3.2.

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_type">
  <xs:restriction base="xs:string">
    <xs:enumeration value="context"/>
    <xs:enumeration value="rfield"/>
    <xs:enumeration value="sfield"/>
    <xs:enumeration value="attributevector"/>
  </xs:restriction>
</xs:simpleType>
```

The following table lists the applicable values.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>context</td>
<td>A searchable index field within a property context, as specified in section 2.8.3.4.</td>
</tr>
<tr>
<td>sfield</td>
<td>A document summary field.</td>
</tr>
<tr>
<td>attributevector</td>
<td>An attribute vector.</td>
</tr>
</tbody>
</table>

2.11 rank.cf

The following table provides information about the file.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Middleware Protocol storage path</td>
<td>RTSearch/webcluster/</td>
</tr>
<tr>
<td>Type of data</td>
<td>Configuration information derived from index schema.</td>
</tr>
</tbody>
</table>
This file contains detailed rank configuration for a specific index schema. The file content MUST derive from the corresponding values in indexConfig.xml. Section 2.11.2 specifies the detailed requirements.

The configuration data in this file is derived from indexConfig.xml, as specified in section 2.8. The file MAY be ignored, because all the configuration information in rank.cf can also be found in indexConfig.xml.

### 2.11.1 ABNF Grammar

The configuration file MUST be according to the following ABNF grammar. In addition to the grammar, the file format allows blank lines and comment lines that begin with the number sign (#).

```abnf
rank-cf   = *comment *crlf rank-profile *crlf
crlf      = LF / (CR LF)
comment   = *("#" *(DIGIT / ALPHA / "(" / ")") )crlf)

; Data type definitions
; ---------------------
yesno     = "yes" / "no"
decimal2  = 1*DIGIT "." 2DIGIT
decimal3  = 1*DIGIT "." 3DIGIT
integer   = 1*DIGIT
prof-name = 1*(DIGIT / ALPHA)
validname = 1*(DIGIT / ALPHA / "{" / "}" / ")") / ")")
catalog-context = validname "." validname
boostbl-file = boostbl-path validname ".tbl"
boostbl-path = "$FASTSEARCH/etc/
boostbl-spec = SP validname SP (boostbl-file / "NULL") crlf
boostbl-file-r = boostbl-path-r validname ".tbl"
boostbl-path-r = "$FASTSEARCH/etc/resources/relevancy/boost-tables/
boostbl-spec-r = SP validname SP (boostbl-file-r / "NULL") crlf

; Rank profile main definition
; -----------------------------
rank-profile = profile-name prof-tuning cat(boost [freshness])
profile-name = "rankprofile" SP prof-name *crlf

; Rank profile tuning
; -------------------
prof-tuning = factor bias qual dyn bins superior cutoff prox clamp *crlf
factor     = "tunefactor" SP decimal2 crlf
bias       = "tunebias" SP integer crlf
dyn        = "dynamicranking" SP "on" crlf

; Static rank property configuration:
qual       = quality1 quality2 quality3 quality4
quality1   = "qualitycomponent" SP "batvhwboost" decimal3 crlf
quality2   = "qualitycomponent" SP "batvdocrank" decimal3 crlf
quality3   = "qualitycomponent" SP "batvsizerank" decimal3 crlf
```

---

[MS-FSSCFG] — v20120630
Search Configuration File Format

Copyright © 2012 Microsoft Corporation.

Release: July 16, 2012
quality4  - "qualitycomponent" SP "batvurldepthrank" decimal3 crlf

; Performance-related dynamic rank cutoff
; ----------------------------------------

bins    - bin posbin xnear
bin      - binlow binhigh binsize
posbin   - posbinlow posbinhigh posbinsize
xnear    - xnearposbinlow xnearposbinhigh xnearposbinsize
binlow   - "binlow" SP integer crlf
binhigh  - "binhigh" SP integer crlf
binsize  - "binsize" SP decimal2 crlf
posbinlow - "posbinlow" SP integer crlf
posbinhigh - "posbinhigh" SP integer crlf
posbinsize - "posbinsize" SP decimal2 crlf
xnearposbinlow - "xnearposbinlow" SP integer crlf
xnearposbinhigh - "xnearposbinhigh" SP integer crlf
xnearposbinsize - "xnearposbinsize" SP decimal2 crlf
superior - "superiorboost" SP "0" crlf
cutoff   - rankcutoff rankcutoffadvval
rankcutoff - "rankcutoff" SP "0" crlf
rankcutoffadvval - "rankcutoffadvval" SP integer crlf

; Proximity ranking configuration
; ----------------------------------------

prox    - firstocc proximity prox-phrase prox-pair prox-triple
firstocc - "firstoccproximity" SP yesno crlf
proximity - "proximity" SP "yes" crlf
prox-phrase - "phraseproximity" SP "yes" crlf
prox-pair - "proximitypairbeforefirstoccproximitytriple" SP yesno crlf
prox-triple - "proximitytriplebeforefirstoccproximityquad" SP yesno crlf
clamp    - "clampstaticrank" SP yesno crlf

; Catalog-specific boosting configuration
; ----------------------------------------

cat-boost - staticprops *catalogboost

; Static rank properties:
staticprops - anchor assoc
anchor - anchor1 anchor2
anchor1 - "extnumoccboostonly" SP "anchortext" SP yesno 1*crlf
anchor2 - "extnumoccboost" SP "anchortext" SP boostbl-file-r 1*crlf
assoc - assoc1 assoc2
assoc1 - "extnumoccboostonly" SP "assocqueries" SP yesno 1*crlf
assoc2 - "extnumoccboost" SP "assocqueries" SP boostbl-file-r 1*crlf

; Rank boost configuration of a given catalog:
catalogboost - operator-boost occboost prox-boost [drilling] *crlf

; Query operator boost:
operator-boost - andb orb phraseb rankb anyb nearb onearb
andb - "andboost" SP validname SP integer crlf
orb - "orboost" SP validname SP integer crlf
phraseb - "phraseboost" SP validname SP integer crlf
rankb          = "rankboost" SP validname SP integer crlf
anyb           = "anyboost" SP validname SP integer crlf
nearb          = "nearboost" SP validname SP integer crlf
onearb         = "orderednearboost" SP validname SP integer crlf

; Occurrence boost:
occboost       = numoccb firstoccb extnumoccb
numoccb        = "numoccboost" boostbl-spec
firstoccb      = "firstoccboost" boostbl-spec-r
extnumoccb     = "extnumoccboost" boostbl-spec-r

; Proximity boost:
prox-boost     = firstoccpn firstoccrpn proxn revproxn
firstoccpn     = firstoccp0 [firstoccp1 firstoccp2 firstoccp3 firstoccp4]
firstoccrpn    = firstoccrp0 [firstoccrp1 firstoccrp2 firstoccrp3 firstoccrp4]
proxn          = proxn0 [proxn1 proxn2 proxn3 proxn4]
revproxn       = revproxn0 [revproxn1 revproxn2 revproxn3 revproxn4]
firstoccp0     = "firstoccproximityboost0" boostbl-spec-r
firstoccp1     = "firstoccproximityboost1" SP validname SP "NULL" crlf
firstoccp2     = "firstoccproximityboost2" SP validname SP "NULL" crlf
firstoccp3     = "firstoccproximityboost3" SP validname SP "NULL" crlf
firstoccp4     = "firstoccproximityboost4" SP validname SP "NULL" crlf
firstoccrp0    = "firstoccrevproximityboost0" boostbl-spec-r
firstoccrp1    = "firstoccrevproximityboost1" SP validname SP "NULL" crlf
firstoccrp2    = "firstoccrevproximityboost2" SP validname SP "NULL" crlf
firstoccrp3    = "firstoccrevproximityboost3" SP validname SP "NULL" crlf
firstoccrp4    = "firstoccrevproximityboost4" SP validname SP "NULL" crlf
proxn0         = "proximityboost0" boostbl-spec-r
proxn1         = "proximityboost1" SP validname SP "NULL" crlf
proxn2         = "proximityboost2" SP validname SP "NULL" crlf
proxn3         = "proximityboost3" SP validname SP "NULL" crlf
proxn4         = "proximityboost4" SP validname SP "NULL" crlf
revproxn0      = "revproximityboost0" boostbl-spec-r
revproxn1      = "revproximityboost1" SP validname SP "NULL" crlf
revproxn2      = "revproximityboost2" SP validname SP "NULL" crlf
revproxn3      = "revproximityboost3" SP validname SP "NULL" crlf
revproxn4      = "revproximityboost4" SP validname SP "NULL" crlf

; Drilling configuration:
drilling       = divspec 1*ctxtboost
divspec        = "divtable" SP validname SP boostbl-file-r crlf
ctxtboost      = contextboost commonctxboost
contextboost   = "contextboost" catalog-context SP integer crlf
commonctxboost = "commoncontextboost" catalog-context SP int234 crlf
int234         = pairValue SP tripleValue SP quadValue
pairValue      = integer
tripleValue    = integer
quadValue      = integer

; Freshness configuration:
freshness      = fresh-file fresh-coeff
fresh-file     = "freshnessboostfile" SP validname SP resolution crlf
fresh-coeff    = "freshnessboostcoefficient" SP integer crlf
resolution     = "second" / "minute" / "hour" / "day" / "year"
## 2.11.2 Configuration Parameter Reference

The configuration parameters in this file are derived from the configuration in indexConfig.xml. Tables throughout this section specify the corresponding parameters in indexConfig.xml.

### 2.11.2.1 Rank Profile-Level Parameters

One rankprofile section MUST appear in the file for each <rankProfile> element in indexConfig.xml, as specified in section 2.8.3.11.

The following table lists the corresponding parameters in indexConfig.xml.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rankprofile</td>
<td>This parameter MUST have the same value as the value of the name attribute for the same rank profile, as specified in section 2.8.3.11.</td>
</tr>
<tr>
<td>tunefactor</td>
<td>These parameters MUST have same value as the corresponding tuneFactor and tuneBias attributes, as specified in section 2.8.3.11.</td>
</tr>
<tr>
<td>tunebias</td>
<td></td>
</tr>
<tr>
<td>qualitycomponent</td>
<td>The attrvector and coefficient parameters MUST have same value as the corresponding attributeVector and coefficient attributes for the same rank profile, as specified in section 2.8.3.14.</td>
</tr>
<tr>
<td>&lt;attrvector&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;coefficient&gt;</td>
<td></td>
</tr>
<tr>
<td>dynamicranking</td>
<td>MUST be set to &quot;on&quot;.</td>
</tr>
<tr>
<td>binlow</td>
<td>These parameters MUST have same value as the corresponding attributes for the same rank profile, as specified in section 2.8.3.15.</td>
</tr>
<tr>
<td>binhigh</td>
<td></td>
</tr>
<tr>
<td>binsize</td>
<td></td>
</tr>
<tr>
<td>posbinlow</td>
<td></td>
</tr>
<tr>
<td>posbinhoigh</td>
<td></td>
</tr>
<tr>
<td>posbinsize</td>
<td></td>
</tr>
<tr>
<td>xnearposbinlow</td>
<td></td>
</tr>
<tr>
<td>xnearposbinhoigh</td>
<td></td>
</tr>
<tr>
<td>xnearposbinsize</td>
<td></td>
</tr>
<tr>
<td>superiorboost</td>
<td></td>
</tr>
<tr>
<td>rankcutoff</td>
<td></td>
</tr>
<tr>
<td>rankcutoffadvval</td>
<td></td>
</tr>
<tr>
<td>firstoccproximity</td>
<td></td>
</tr>
<tr>
<td>proximity</td>
<td></td>
</tr>
<tr>
<td>phraseproximity</td>
<td></td>
</tr>
<tr>
<td>proximitypairbeforefirstoccproximitytriple</td>
<td></td>
</tr>
<tr>
<td>proximitytriplebeforefirstoccproximityquad</td>
<td></td>
</tr>
<tr>
<td>clampstaticrank</td>
<td></td>
</tr>
<tr>
<td>freshnessboostfile</td>
<td>The attrvector parameter MUST be equal to the value of the name attribute for the same rank profile, as specified in section 2.8.3.24.</td>
</tr>
<tr>
<td>&lt;attrvector&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;resolution&gt;</td>
<td>The resolution parameter MUST be equal to the value of the value attribute for the same rank profile, as specified in section 2.8.3.25.</td>
</tr>
</tbody>
</table>

Copyright © 2012 Microsoft Corporation.

Release: July 16, 2012
2.11.2.2 Context Catalog-Level Parameters

The structure of the section for context catalog–level parameters MUST follow the indexConfig.xml catalog structure, as specified in section 2.8.5. The parameters MUST be present and MUST contain the specified value for each corresponding instance in indexConfig.xml. If not otherwise specified in the following table, catalog MUST be the corresponding catalogName attribute for the corresponding rankedCatalog element in indexConfig.xml, as specified in section 2.8.3.18.

The following table lists the corresponding parameters in indexConfig.xml.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>freshnessboostcoefficient</td>
<td>This parameter MUST have the same value as the value attribute for the same rank profile, as specified in section 2.8.3.26.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>andboost</td>
<td>The value parameter MUST have the same value as the value attribute for the corresponding element, as specified in section 2.8.3.18.</td>
</tr>
<tr>
<td>orboost</td>
<td>The value parameter MUST have the same value as the value attribute for the corresponding element, as specified in section 2.8.3.18.</td>
</tr>
<tr>
<td>phraseboost</td>
<td>The value parameter MUST have the same value as the value attribute for the corresponding element, as specified in section 2.8.3.18.</td>
</tr>
<tr>
<td>rankboost</td>
<td>The value parameter MUST have the same value as the value attribute for the corresponding element, as specified in section 2.8.3.18.</td>
</tr>
<tr>
<td>anyboost</td>
<td>The value parameter MUST have the same value as the value attribute for the corresponding element, as specified in section 2.8.3.18.</td>
</tr>
<tr>
<td>nearboost</td>
<td>The value parameter MUST have the same value as the value attribute for the corresponding element, as specified in section 2.8.3.18.</td>
</tr>
<tr>
<td>orderednearboost</td>
<td>The value parameter MUST have the same value as the value attribute for the corresponding element, as specified in section 2.8.3.18.</td>
</tr>
<tr>
<td>numoccboost</td>
<td>The value parameter MUST have the same value as the value attribute for the corresponding element, as specified in section 2.8.3.18.</td>
</tr>
<tr>
<td>firstoccboost</td>
<td>The value parameter MUST have the same value as the value attribute for the corresponding element, as specified in section 2.8.3.18.</td>
</tr>
<tr>
<td>firstoccproximityboost0</td>
<td>The fileName parameter MUST have the same value as the fileName attribute, as specified in section 2.8.3.21, where firstOcc is set to &quot;yes&quot; and direction is set to &quot;forward&quot;.</td>
</tr>
<tr>
<td>firstoccproximityboost[1-4]</td>
<td>For every firstoccproximityboost0 entry, there MUST be four...</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>&lt;catalog&gt;</code> NULL</td>
<td>subsequent entries (<code>firstoccproximityboost1</code>, <code>firstoccproximityboost2</code>, <code>firstoccproximityboost3</code>, and <code>firstoccproximityboost4</code>) for the same <code>catalog</code> parameter, where the last parameter MUST be &quot;NULL&quot;.</td>
</tr>
<tr>
<td><code>firstoccrevproximityboost0</code> <code>&lt;catalog&gt;</code> <code>&lt;fileName&gt;</code></td>
<td>The <code>fileName</code> parameter MUST have the same value as the <code>fileName</code> attribute, as specified in section 2.8.3.21, where <code>firstOcc=&quot;yes&quot;</code> and <code>direction=&quot;backward&quot;</code>.</td>
</tr>
<tr>
<td><code>firstoccrevproximityboost[1-4]</code> <code>&lt;catalog&gt;</code> NULL</td>
<td>For every <code>firstoccrevproximityboost0</code> entry, there MUST be four subsequent entries (<code>firstoccrevproximityboost1</code>, <code>firstoccrevproximityboost2</code>, <code>firstoccrevproximityboost3</code>, and <code>firstoccrevproximityboost4</code>) for the same <code>catalog</code> parameter, where the last parameter MUST be &quot;NULL&quot;.</td>
</tr>
<tr>
<td><code>extnumoccbost only</code> <code>&lt;catalog&gt;</code> &quot;yes&quot;</td>
<td>The <code>catalog</code> parameter MUST have the same value as the <code>catalogName</code> attribute, as specified in section 2.8.3.17.</td>
</tr>
<tr>
<td><code>proximityboost0</code> <code>&lt;catalog&gt;</code> <code>&lt;fileName&gt;</code></td>
<td>The <code>fileName</code> parameter MUST have the same value as the <code>fileName</code> attribute, as specified in section 2.8.3.21, where <code>firstOcc</code> is set to &quot;no&quot; and `direction=&quot;forward&quot;.</td>
</tr>
<tr>
<td><code>proximityboost[1-4]</code> <code>&lt;catalog&gt;</code> NULL</td>
<td>For every <code>proximityboost0</code> entry, there MUST be four subsequent entries (<code>proximityboost1</code>, <code>proximityboost2</code>, <code>proximityboost3</code>, and <code>proximityboost4</code>) for the same <code>catalog</code> parameter, where the last parameter MUST be &quot;NULL&quot;.</td>
</tr>
<tr>
<td><code>revproximityboost0</code> <code>&lt;catalog&gt;</code> <code>&lt;fileName&gt;</code></td>
<td>The <code>fileName</code> parameter MUST have the same value as the <code>fileName</code> attribute, as specified in section 2.8.3.21, where <code>firstOcc</code> is set to &quot;no&quot; and <code>direction</code> is set to &quot;backward&quot;.</td>
</tr>
<tr>
<td><code>revproximityboost[1-4]</code> <code>&lt;catalog&gt;</code> NULL</td>
<td>For every <code>revproximityboost0</code> entry, there MUST be four subsequent entries (<code>revproximityboost1</code>, <code>revproximityboost2</code>, <code>revproximityboost3</code>, and <code>revproximityboost4</code>) for the same <code>catalog</code> parameter, where the last parameter MUST be &quot;NULL&quot;.</td>
</tr>
</tbody>
</table>
2.12  FieldProperties.xml

The following table provides information about the file.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Middleware Protocol storage path</td>
<td>Schema/webcluster/</td>
</tr>
<tr>
<td>Type of data</td>
<td>Configuration information derived from index schema.</td>
</tr>
<tr>
<td>File format</td>
<td>XML schema file.</td>
</tr>
</tbody>
</table>

This file MUST contain managed property configuration information that is derived from the index schema.

2.12.1  Global Elements

2.12.1.1  field-properties

The field-properties element is a container for field elements.

```xml
<xs:element name="field-properties" type="CT_field-properties"/>
```

2.12.2  Global Attributes

None.

2.12.3  Complex Types

2.12.3.1  CT_field-properties

Referenced by: <field-properties>

A complex type that is a container for field elements.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_field-properties">
  <xs:sequence>
    <xs:element name="field" minOccurs="1" maxOccurs="unbounded" type="CT_field"/>
  </xs:sequence>
  <xs:attribute name="default-index" type="xs:string" use="required"/>
</xs:complexType>
```

field: A CT_field element. MUST contain field elements for all managed properties specified within the index schema.

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>default-</td>
<td>The name of the default index. MUST be the name of the full-text index field specified in</td>
</tr>
</tbody>
</table>
Name | Description
--- | ---
index | the index schema with **IsDefault** set.

### 2.12.3.2 CT_field

**Referenced by:** CT_field-properties

A complex type that specifies item processing parameters for one managed property or full-text index field.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_field">
  <xs:sequence>
    <xs:element name="language-tokenization" minOccurs="0" maxOccurs="1" type="CT_language-tokenization"/>
    <xs:element name="substring-tokenization" minOccurs="0" maxOccurs="1" type="CT_substring-tokenization"/>
    <xs:element name="generic-tokenization" minOccurs="0" maxOccurs="1" type="CT_generic-tokenization"/>
    <xs:element name="result" type="CT_result"/>
  </xs:sequence>
  <xs:attribute name="alias" type="xs:string" use="required"/>
  <xs:attribute name="kind" type="ST_fieldKind" use="required"/>
  <xs:attribute name="indexed" type="ST_yesno" use="required"/>
  <xs:attribute name="type" type="ST_fieldType" use="required"/>
  <xs:attribute name="decimal-precision" type="xs:int" use="optional"/>
  <xs:attribute name="boundary" type="ST_yesno" use="required"/>
  <xs:attribute name="wildcard" type="ST_wildcardAtt" use="required"/>
  <xs:attribute name="defines-freshness" type="ST_yes" use="optional"/>
</xs:complexType>
```

**language-tokenization:** A CT_language-tokenization element.

**substring-tokenization:** A CT_substring-tokenization element.

**generic-tokenization:** A CT_generic-tokenization element.

**result:** A CT_result element.

**Attributes:**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alias</td>
<td>The name of the managed property or full-text index field. MUST be the managed property or full-text index field name in the index schema.</td>
</tr>
<tr>
<td>kind</td>
<td>An ST_fieldKind attribute that specifies the representation in the search index.</td>
</tr>
<tr>
<td>indexed</td>
<td><strong>yes:</strong> Is indexed.&lt;br&gt;<strong>no:</strong> Is not indexed, and is available only as a document summary.&lt;br&gt;MUST be set according to the Queryable parameter for the managed property in the index schema.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>type</td>
<td>An ST_fieldType attribute that specifies the data type.</td>
</tr>
<tr>
<td>decimal-precision</td>
<td>Decimal precision according to the index schema setting. MUST occur only in association with type set to &quot;decimal&quot;.</td>
</tr>
<tr>
<td>boundary</td>
<td>Boundary matching enabled. MUST be set to &quot;yes&quot; for all managed properties of type string, and &quot;no&quot; for all other field elements.</td>
</tr>
<tr>
<td>wildcard</td>
<td>An ST_wildcardAtt attribute that specifies wildcard search. MUST be set to &quot;full&quot; for all managed properties of type string and all full-text index fields, and &quot;no&quot; for all other field elements.</td>
</tr>
<tr>
<td>defines-freshness</td>
<td>MUST be set to &quot;yes&quot; for the managed property that will be used as the basis for freshness rank evaluation. MUST not be included for any other managed property.</td>
</tr>
</tbody>
</table>

2.12.3.3 CT_generic-tokenization

Referenced by: CT_field

A complex type that specifies language-independent linguistic processing, to be used for managed properties that are not language-aware.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_generic-tokenization">
  <xs:attribute name="separator" type="xs:string" use="required"/>
</xs:complexType>
```

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>separator</td>
<td>MUST be set to an empty string (&quot;&quot;).</td>
</tr>
</tbody>
</table>

2.12.3.4 CT_substring-tokenization

Referenced by: CT_field

A complex type that specifies tokenization for the substring search type.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_substring-tokenization">
  <xs:attribute name="N" type="xs:int" use="required"/>
</xs:complexType>
```

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>N-gram value for the substring. MUST have the same value as the subStringSearch attribute for this managed property, as specified in section 2.8.3.5.</td>
</tr>
</tbody>
</table>
### 2.12.3.5 CT_language-tokenization

**Referenced by: CT_field**

A complex type that specifies language-specific linguistic processing.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_language-tokenization">
  <xs:attribute name="lemmatization" type="ST_yesno" use="required"/>
  <xs:attribute name="mode" type="ST_tokenization_mode" use="optional"/>
</xs:complexType>
```

**Attributes:**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| lemmatization   | Stemming enabled yes or no. MUST be set for managed properties that have stemming enabled.  
                  | This corresponds to `LemmatizationEnabled` parameter in the index schema specified in section 1.3.2.1. |
| mode            | Special tokenization mode.                                                 |

### 2.12.3.6 CT_result

**Referenced by: CT_field**

A complex type that specifies how the result provides the document summary.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_result">
  <xs:attribute name="type" use="required" type="ST_resulttype"/>
  <xs:attribute name="max-size" type="xs:int" use="optional"/>
</xs:complexType>
```

**Attributes:**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>An <code>ST_resulttype</code> attribute that specifies the type of document summary to be provided.</td>
</tr>
</tbody>
</table>
| max-size | The maximum size of the document summary in kilobytes.  
          | This corresponds to the managed property `MaxResultSize` parameter in the index schema if `type` is set to "static" or `type` is set to "dynamic". For more information about related index schema concepts, see section 1.3.2.1.  
          | MUST NOT be present if `type` is set to "no". |
2.12.4 Simple Types

2.12.4.1 ST_resulttype

Referenced by: CT_result

A simple type that specifies the type of document summary to be provided. This corresponds to the SummaryType index schema configuration specified in section 1.3.2.1, with the following relation:

- **Disabled** is set to "no", **Static** is set to "static", and **Dynamic** is set to "dynamic".

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_resulttype">
  <xs:restriction base="xs:string">
    <xs:enumeration value="no"/>
    <xs:enumeration value="static"/>
    <xs:enumeration value="dynamic"/>
  </xs:restriction>
</xs:simpleType>
```

The following table defines the values.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>no</td>
<td>No document summary will be provided.</td>
</tr>
<tr>
<td>static</td>
<td>A static document summary.</td>
</tr>
<tr>
<td>dynamic</td>
<td>A document hit highlighted summary.</td>
</tr>
</tbody>
</table>

2.12.4.2 ST_yes

Referenced by: CT_field

A simple type that specifies the Boolean condition value "yes".

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_yesno">
  <xs:restriction base="xs:string">
    <xs:enumeration value="yes"/>
  </xs:restriction>
</xs:simpleType>
```

2.12.4.3 ST_yesno

Referenced by: CT_field, CT_language-tokenization

A simple type that specifies the Boolean condition values "yes" and "no".

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_yesno">
  <xs:restriction base="xs:string">
    <xs:enumeration value="yes"/>
  </xs:restriction>
</xs:simpleType>
```
### 2.12.4.4 ST_fieldKind

**Referenced by:** CT_field

A simple type that specifies how the search index represents this index field.

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_fieldKind">
  <xs:restriction base="xs:string">
    <xs:enumeration value="field"/>
    <xs:enumeration value="composite"/>
  </xs:restriction>
</xs:simpleType>
```

The following table lists the applicable values.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>A managed property as specified in the index schema.</td>
</tr>
<tr>
<td>composite</td>
<td>A full-text index field as specified in the index schema.</td>
</tr>
</tbody>
</table>

### 2.12.4.5 ST_fieldType

**Referenced by:** CT_field

A simple type that specifies the data type for the index field. MUST be set according to the managed property Type parameter in the index schema, as specified in section 1.3.2.1, by using the following data type mapping:

- **Text** to string
- **Integer** to int
- **Boolean** to string
- **Float** to float
- **Decimal** to decimal
- **Datetime** to datetime

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_fieldType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="string"/>
    <xs:enumeration value="int"/>
  </xs:restriction>
</xs:simpleType>
```
The following table lists the applicable values.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>A UTF-8 text data type for text search.</td>
</tr>
<tr>
<td>int</td>
<td>A 64-bit signed integer.</td>
</tr>
<tr>
<td>float</td>
<td>A 64-bit floating-point data type that uses base 2 for the exponent. The exponent uses 11 bits, and the mantissa uses 52 bits.</td>
</tr>
<tr>
<td>decimal</td>
<td>A fixed-point signed decimal data type.</td>
</tr>
<tr>
<td>datetime</td>
<td>A datetime data type. This data type is represented as a 64-bit unsigned integer in the search index.</td>
</tr>
</tbody>
</table>

2.12.4.6 ST_wildcardAtt

Referenced by: CT_field

A simple type that specifies wildcard search.

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_wildcardAtt">
  <xs:restriction base="xs:string">
    <xs:enumeration value="no"/>
    <xs:enumeration value="full"/>
  </xs:restriction>
</xs:simpleType>
```

The following table lists the applicable values.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>no</td>
<td>Wildcard search disabled.</td>
</tr>
<tr>
<td>full</td>
<td>Wildcard search enabled.</td>
</tr>
</tbody>
</table>

2.12.4.7 ST_tokenization_mode

Referenced by: CT_language-tokenization

A simple type that specifies the set of special tokenization modes supported by the product as specified in [MS-FSIN] section 2.

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_tokenization_mode">
</xs:simpleType>
```
The following table lists the applicable values.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>uri</td>
<td>Tokenization mode with the name &quot;uri&quot;.</td>
</tr>
<tr>
<td>site-url</td>
<td>Tokenization mode with the name &quot;site-url&quot;.</td>
</tr>
</tbody>
</table>

2.13 Boost Table Files

The following table provides information about the file.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Middleware Protocol storage path</td>
<td>RTSearch/webcluster/boost-tables</td>
</tr>
<tr>
<td>Type of data</td>
<td>Implementation-specific configuration information.</td>
</tr>
<tr>
<td>File format</td>
<td>Boost table text file.</td>
</tr>
</tbody>
</table>

This section specifies the file format for the boost table files used for dynamic ranking. The following subsections specify the content of the files. The files MUST be exactly as specified in the subsections.

2.13.1 Occurrence Boost Table Files

Occurrence boost table files provide a mapping from the occurrences of a term to the boost value of the occurrence. If term occurrence is n, the actual boost value is the numeric value at line n+1 in the boost file. The following table specifies the occurrence boost table files.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| `<model>_<ctxt>_numoccboost.tbl`                   | This file provides a mapping from the normal number of term occurrences to occurrence boost value, where:  
|                                                    | `<model>` is the name of a rank model as specified in the index schema according to section 1.3.2.4. The rank model enables implementation-specific rank tuning.  
|                                                    | `<ctxt>` is a property context of type external, according to section 2.8.3.4.  
|                                                    | See section 2.8.4.2 for more details about normal and external property contexts. |
| `<model>_<ctxt>_extnumoccboost.tbl`                | This file provides a mapping from number of term occurrences in property contexts that are specified as "external" to external occurrence boost value, where:  
|                                                    | `<model>` is a RankModelName, as specified in the index schema according to section 1.3.2.4.  
<p>|                                                    | <code>&lt;ctxt&gt;</code> is a property context of type external, according to section 2.8.3.4.  |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;model&gt;_&lt;ctxt&gt;_firstoccboost.tbl</td>
<td>This file provides a mapping from the first occurrence of a term to the first-occurrence boost value, where:&lt;br&gt;&lt;model&gt; is a RankModelName, as specified in the index schema according to section 1.3.2.4.&lt;br&gt;&lt;ctxt&gt; is a property context of type external, according to section 2.8.3.4.</td>
</tr>
</tbody>
</table>

### 2.13.2 Proximity Boost Table Files

Proximity boost table files provide a mapping from multi-term proximity to proximity for boost values. If proximity distance is n, the boost value is the numeric value at line n in the boost file. The following table specifies the proximity boost table files.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;model&gt;_&lt;ctxt&gt;<em>proximity_boost_firstocc</em>&lt;dir&gt;_0.tbl</td>
<td>This file provides a mapping from first-occurrence proximity distance to proximity boost value, where &lt;model&gt; is a RankModelName, as specified in the index schema according to section 1.3.2.4. The rank model enables implementation-specific rank tuning.&lt;br&gt;&lt;ctxt&gt; is a property context of type external, according to section 2.8.3.4.&lt;br&gt;&lt;dir&gt; specifies a value of &quot;fw&quot; for forward proximity boost, or it specifies a value of &quot;bw&quot; for backward proximity boost.</td>
</tr>
<tr>
<td>&lt;model&gt;_&lt;ctxt&gt;<em>proximity_boost_nofirstocc</em>&lt;dir&gt;_0.tbl</td>
<td>This file provides a mapping from occurrence proximity distance to proximity boost value, where &lt;model&gt; is a RankModelName, as specified in the index schema according to section 1.3.2.4.&lt;br&gt;&lt;ctxt&gt; is a property context of type external, according to section 2.8.3.4.&lt;br&gt;&lt;dir&gt; specifies a value of &quot;fw&quot; for forward proximity boost, or it specifies a value of &quot;bw&quot; for backward proximity boost.</td>
</tr>
</tbody>
</table>

### 2.13.3 Global Term Frequency Boost Table File

The boost table file for global term frequency provides rank inverse boost values for document term frequency adjustment against global term frequency. If global term frequency is t, the document frequency adjustment factor is the numeric value at line \( \log_2 t \) in the boost file. The following table specifies the boost table file for global term frequency.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;model&gt;_&lt;ctxt&gt;_divtable.tbl</td>
<td>This file provides a mapping from global term frequency to term</td>
</tr>
</tbody>
</table>
frequency division factor, where \(<model>\) is a \texttt{RankModelName}, as specified in the index schema according to section 1.3.2.4. 
\(<ctxt>\) is a property context of type \texttt{external}, according to section 2.8.3.4.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
|               | frequency division factor, where \(<model>\) is a \texttt{RankModelName}, as specified in the index schema according to section 1.3.2.4. 
\(<ctxt>\) is a property context of type \texttt{external}, according to section 2.8.3.4. |

2.14 \texttt{rankspace.xml}

The following table provides information about the file.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Middleware Protocol storage path</td>
<td>RTSearch/webcluster/</td>
</tr>
<tr>
<td>Type of data</td>
<td>Configuration information derived from index schema.</td>
</tr>
<tr>
<td>File format</td>
<td>XML schema file.</td>
</tr>
</tbody>
</table>

This file contains rank profile configuration information derived from the index schema that is associated with rank profiles.

2.14.1 Global Elements

2.14.1.1 rankspace

The \texttt{rankspace} element is a container for \texttt{ranking} elements.

\[
\text{<xs:element name="rankspace" type="CT_rankspace"/>}
\]

2.14.2 Global Attributes

None.

2.14.3 Complex Types

2.14.3.1 CT_rankspace

Referenced by: \texttt{rankspace}

A complex type that specifies a list of rank profiles defined in the system.

This complex type is defined as follows:

\[
\text{<xs:complexType name="CT_rankspace">}
\text{<xs:sequence>}
\text{<xs:element maxOccurs="unbounded" name="ranking" type="CT_ranking"/>}
\text{</xs:sequence>}
\text{</xs:complexType>}
\]

\texttt{ranking}: A \texttt{CT_ranking} element. MUST include one \texttt{ranking} element for each rank profile in the index schema.
Attributes: None.

### 2.14.3.2 CT_ranking

Referenced by: CT_rankspace

A complex type that specifies the mapping configuration of a rank profile.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_ranking">
  <xs:attribute name="name" type="xs:string" use="required"/>
  <xs:attribute name="description" type="ST_description" use="required"/>
  <xs:attribute name="descendingIndex" type="ST_alwaysZero" use="required"/>
</xs:complexType>
```

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>The name of rank profile. MUST be a rank profile name as specified in indexConfig.xml.</td>
</tr>
<tr>
<td>description</td>
<td>A fixed value. MUST be set as specified in the XML schema, and MUST be ignored.</td>
</tr>
<tr>
<td>descendingIndex</td>
<td>An implementation-specific parameter. MUST be set to 0.</td>
</tr>
</tbody>
</table>

### 2.14.4 Simple Types

#### 2.14.4.1 ST_description

Referenced by: CT_ranking

A simple type that specifies an implementation-specific parameter with a fixed value.

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_description">
  <xs:restriction base="xs:string">
    <xs:enumeration value="BLISS generated"/>
  </xs:restriction>
</xs:simpleType>
```

#### 2.14.4.2 ST_alwaysZero

Referenced by: CT_ranking

A simple type that specifies an implementation-specific parameter with a fixed value.

```xml
<xs:simpleType name="ST_alwaysZero">
  <xs:restriction base="xs:string">
    <xs:enumeration value="0"/>
  </xs:restriction>
</xs:simpleType>
```
2.15 resultspace.xml

The following table provides information about the file.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Middleware Protocol storage path</td>
<td>RTSearch/webcluster/</td>
</tr>
<tr>
<td>Type of data</td>
<td>Implementation-specific configuration information derived from index schema.</td>
</tr>
<tr>
<td>File format</td>
<td>XML schema file.</td>
</tr>
</tbody>
</table>

This file contains result view configuration information derived from the index schema to be used for mapping of result views.

2.15.1 Global Elements

2.15.1.1 resultspace

The resultspace element is a container for result-view elements.

```xml
<xs:element name="resultspace" type="CT_resultspace"/>
```

2.15.2 Global Attributes

None.

2.15.3 Complex Types

2.15.3.1 CT_resultspace

Referenced by: resultspace

A complex type that is a container for result view definitions.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_resultspace">
  <xs:sequence>
    <xs:element name="result-view" type="CT_result-view"/>
  </xs:sequence>
</xs:complexType>
```

result-view: One CT_result-view element.

Attributes: None.
2.15.3.2  CT_result-view

Referenced by: CT_resultspace

A complex type that specifies one result view.

This element contains result view mapping configuration, as derived from the index schema. It MUST contain all index fields of type bsum that are specified in the servedcontent summary class in summary.cf. This element MUST contain one field element for each document summary to be presented on the [MS-FSQR] protocol interface.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_result-view">
  <xs:sequence>
    <xs:element maxOccurs="unbounded" name="field" type="CT_field"/>
  </xs:sequence>
  <xs:attribute name="index" type="ST_index"/>
  <xs:attribute name="name" type="ST_name" use="required"/>
</xs:complexType>
```

field: A CT_field element.

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>index</td>
<td>An ST_index attribute that specifies an index identifier for this result view. MUST be set to 0.</td>
</tr>
<tr>
<td>name</td>
<td>MUST contain a value of &quot;DATASEARCHDEFAULT&quot;. The value MUST be ignored.</td>
</tr>
</tbody>
</table>

2.15.3.3  CT_field

Referenced by: CT_result-view

A complex type that specifies result view configuration for a particular managed property. It MUST contain all managed properties of type bsum that are specified in the servedcontent summary class in summary.cf. This element contains one field element for each document summary to be presented on the [MS-FSQR] protocol interface.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_field">
  <xs:attribute name="type" type="ST_type" use="required"/>
  <xs:attribute name="name" type="xs:string" use="required"/>
</xs:complexType>
```

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>The field type. MUST be &quot;string&quot; or &quot;integer&quot;.</td>
</tr>
<tr>
<td>name</td>
<td>The name of the managed property.</td>
</tr>
</tbody>
</table>

111 / 176
2.15.4 Simple Types

2.15.4.1 ST_index

Referenced by: CT_result-view

A simple type that specifies an implementation-specific parameter with a fixed value.

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_index">
  <xs:restriction base="xs:string">
    <xs:enumeration value="0"/>
  </xs:restriction>
</xs:simpleType>
```

2.15.4.2 ST_name

Referenced by: CT_result-view

A simple type that specifies an implementation-specific parameter with a fixed value.

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_name">
  <xs:restriction base="xs:string">
    <xs:enumeration value="DATASEARCHDEFAULT"/>
  </xs:restriction>
</xs:simpleType>
```

2.15.4.3 ST_type

Referenced by: CT_field

A simple type that specifies the type of document summary.

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_type">
  <xs:restriction base="xs:string">
    <xs:enumeration value="string"/>
    <xs:enumeration value="integer"/>
  </xs:restriction>
</xs:simpleType>
```

The following table lists the applicable values.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>The document summary type for text document summaries.</td>
</tr>
<tr>
<td>integer</td>
<td>The document summary type for numeric document summaries.</td>
</tr>
</tbody>
</table>
2.16  search_preload

The following table provides information about the file.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Middleware Protocol storage path</td>
<td>RTSearch/webcluster/</td>
</tr>
<tr>
<td>Type of data</td>
<td>Implementation-specific configuration information.</td>
</tr>
<tr>
<td>File format</td>
<td>File name text file.</td>
</tr>
</tbody>
</table>

This file contains a list of configuration file names that are preloaded upon system restart of the query matching service and query processing service. The file MUST contain a list of all boost tables to load (see section 2.13), followed by the content specified here.

fdispatch.addon
fsearch.addon
index.cf
rank.cf
summary.cf
summary.map
template.rc
templates/rtsearch/fdispatch_fhtml/error.templ
templates/rtsearch/fdispatch_fhtml/footer.templ
templates/rtsearch/fdispatch_fhtml/header.templ
templates/rtsearch/fdispatch_fhtml/next.templ
templates/rtsearch/fdispatch_fhtml/nohits.templ
templates/rtsearch/fdispatch_fhtml/prev.templ
templates/rtsearch/fdispatch_fhtml/result.templ
templates/rtsearch/fdispatch_fhtml/templates.rc

2.17  sources.xml

The following table provides information about the file.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Middleware Protocol storage path</td>
<td>QRServer/webcluster/etc/qrserver/</td>
</tr>
<tr>
<td>Type of data</td>
<td>Implementation-specific configuration information.</td>
</tr>
<tr>
<td>File format</td>
<td>Fixed XML file.</td>
</tr>
</tbody>
</table>

This file contains fixed implementation-specific configuration information related to service connection.

2.17.1  XML Content

The configuration file content is fixed and MUST be as specified in the following XML.

```xml
<?xml version="1.0" encoding="utf-8"?>
<source>
```
name="webcluster"
engine="file:etc/qrsrvr/webcluster.spec"
urlconfig="cs:///RTSearch/summary.cf"
rankconfig="cs:///RTSearch/rank.cf"
fieldspec="cs:///etc/qrsrvr/fieldspec.xml"
fieldmap="cs:///etc/qrsrvr/resultfield.map"
qtpipeline="office14"
requerycount="1"
defaultcatn="0">
<timeout query="12" docsum="17"/>
</source>
</sources>

2.18 summary.cf

The following table provides information about the file.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Middleware Protocol storage path</td>
<td>RTSearch/webcluster/</td>
</tr>
<tr>
<td>Type of data</td>
<td>Configuration information derived from index schema.</td>
</tr>
<tr>
<td>File format</td>
<td>ABNF text file.</td>
</tr>
</tbody>
</table>

The summary.cf configuration file specifies managed property names and managed property types used in query requests and returned in query results on the [MS-FSDQE] interface. The summary.cf configuration file represents a view of the index that can be returned in a query result. A summary class in the configuration file represents a view. The summary class **servedcontent** MUST be present in the configuration file. For more information, see section 2.8.3.31.

2.18.1 ABNF Grammar

The configuration file MUST be as specified in the following ABNF grammar.

```
summary-cf     = "idtype integer" crlf crlf 1*summaryclass
crlf           = LF / (CR LF)
name           = 1*(DIGIT / ALPHA)
fieldprefix    = "bsum" / "bsrc" / "bdlg"
built-in       = "internalid" / "contentid" / "contentids" / "collection" / "ranklog"
fieldname      = built-in / (fieldprefix name)
sumtype        = "string" / "longstring" / "data" / "longdata"
summaryclass   = class 1*field *crlf
class          = "class" SP name SP "id" SP 1*DIGIT crlf
field          = "field" SP fieldname SP "type" sumtype crlf
```
2.18.2 Configuration Parameter Reference

The \textit{idtype} parameter MUST be the value of the \texttt{fieldTypeUsedForId} attribute, as specified in section 2.8.3.31. This specifies the data type for the identifier associated with each summary class. Within the configuration file, each summary class MUST be specified as follows:

- class \texttt{<classname>} id \texttt{<ID>} field \texttt{<field specification>} field \texttt{<field specification>}

The \texttt{<ID>} element is an integer that specifies a summary class used by the protocol specified in \texttt{[MS-FSDQE]}. Within a summary class, each \texttt{<field specification>} element MUST be specified as follows:

- field \texttt{<prefix><fieldname> type <fieldtype>}

\texttt{prefix}: MUST use the naming conventions in section 2.1.2.

\texttt{fieldname}: MUST be the name of a managed property.

\texttt{fieldtype}: MUST be one of the supported document summary types, as specified in section 2.1.3.

2.18.3 Summary Classes

The summary.cf file MUST contain the summary classes, as specified in indexConfig.xml. For more information, see section 2.8.3.31.

2.19 summary.map

The following table provides information about the file.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Middleware Protocol storage path</td>
<td>RTSearch/webcluster/</td>
</tr>
<tr>
<td>Type of data</td>
<td>Implementation-specific configuration information.</td>
</tr>
<tr>
<td>File format</td>
<td>Name value text file.</td>
</tr>
</tbody>
</table>

This is an implementation-specific configuration file that the query matching service uses to configure which summary fields in the summary class to overwrite.

The file contains configuration information for the query matching service generated from the index schema.

The following table specifies the configuration parameters that MUST be in the file.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>defaultoutputclass \texttt{&lt;output class&gt;}</td>
<td>\texttt{&lt;output class&gt;} MUST be the identifier specified for the summary class \texttt{servedcontent} in summary.cf. This is the default summary class in use if no summary class is specified in a query.</td>
</tr>
<tr>
<td>override ranklog ranklog</td>
<td>An implementation-specific rank log configuration. Value MUST be as specified. This specifies that the summary field &quot;ranklog&quot; is overwritten by debug information from the ranking process.</td>
</tr>
<tr>
<td>override \texttt{&lt;output summary field&gt;}</td>
<td>The \texttt{override} keyword specifies that a summary field from the summary class SHOULD be overwritten when returning items. The third argument, in</td>
</tr>
</tbody>
</table>
### Parameter Description

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dynamicteaser &lt;input summary field&gt;</td>
<td>this case &quot;dynamicteaser&quot;, specifies with what the summary field SHOULD be overwritten. When the third field is equal to &quot;dynamicteaser&quot;, this specifies to overwrite the output summary field with a dynamic hit highlighted version of the specified input summary field.</td>
</tr>
<tr>
<td>override &lt;output summary field&gt; juniperlog &lt;input summary field&gt;</td>
<td>This specifies which summary field to overwrite with a debug log from the hit highlighting process.</td>
</tr>
</tbody>
</table>

### 2.20  `summaryclasses.xml`

The following table provides information about the file.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Middleware Protocol storage path</td>
<td>Schema/webcluster/</td>
</tr>
<tr>
<td>Type of data</td>
<td>Configuration information derived from index schema.</td>
</tr>
<tr>
<td>File format</td>
<td>XML schema file.</td>
</tr>
</tbody>
</table>

This file contains document summary configuration information derived from the index schema and used by the indexing service.

#### 2.20.1  Global Elements

##### 2.20.1.1  `summary-input-classes`

The `summary-input-classes` element is a container for input summary classes.

```xml
<x:s:element name="summary-input-classes" type="CT_summary-input-classes"/>
```

#### 2.20.2  Global Attributes

None.

#### 2.20.3  Complex Types

##### 2.20.3.1  `CT_summary-input-classes`

Referenced by: `summary-input-classes`

A complex type that specifies document summary classes.

This element MUST contain one or more `summaryClass` elements.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_summary-input-classes">
  <xs:sequence>
    <xs:element name="summaryClass" type="CT_summaryClass"/>
  </xs:sequence>
</xs:complexType>
```
summaryClass: A CT_summaryClass element that specifies a document summary class. This is an input summary class that is used to map managed properties to document summaries prior to indexing. The number of summaryClass elements MUST be equal to the set of input summary classes, as specified in indexConfig.xml. For details, refer to section 2.8.3.31.

Attributes: None.

2.20.3.2 CT_summaryClass

Referenced by: CT_summary-input-classes

A complex type that specifies one input summary class.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_summaryClass">
  <xs:sequence>
    <xs:element maxOccurs="unbounded" name="summaryField" type="CT_summaryField"/>
  </xs:sequence>
  <xs:attribute name="name" type="ST_className" use="required"/>
  <xs:attribute name="type" type="ST_classType" use="required"/>
</xs:complexType>
```

summaryField: A CT_summaryField element. The number of summaryField elements MUST be equal to all document summaries required to generate the associated output summary class servedcontent, as specified in summary.cf.

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>The name of the summary class. Refer to section 2.8.3.31 for naming rules for summary classes.</td>
</tr>
<tr>
<td>type</td>
<td>The type of summary class. MUST have the value &quot;in&quot;.</td>
</tr>
</tbody>
</table>

2.20.3.3 CT_summaryField

Referenced by: CT_summaryClass

A complex type that specifies one document summary.

This complex type is defined as follows:

```xml
<xs:complexType name="CT_summaryField">
  <xs:attribute name="name" type="xs:string" use="required"/>
  <xs:attribute name="type" type="ST_summaryType" use="required"/>
  <xs:attribute name="compression" type="ST_compression" use="optional"/>
</xs:complexType>
```

Attributes:
### 2.20.4 Simple Types

#### 2.20.4.1 ST_classType

Referenced by: CT_summaryClass

A simple type that specifies an implementation-specific parameter with a fixed value.

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_classType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="in"/>
  </xs:restriction>
</xs:simpleType>
```

#### 2.20.4.2 ST_className

Referenced by: CT_summaryClass

A simple type that specifies an implementation-specific parameter with a fixed value.

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_className">
  <xs:restriction base="xs:string">
    <xs:enumeration value="content"/>
  </xs:restriction>
</xs:simpleType>
```

#### 2.20.4.3 ST_summaryType

Referenced by: CT_summaryField

A simple type that specifies document summary type. MUST be one of the supported document summary types, as specified in section 2.1.3.

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_summaryType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="string"/>
    <xs:enumeration value="longstring"/>
    <xs:enumeration value="data"/>
  </xs:restriction>
</xs:simpleType>
```
The following table lists the applicable values.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>The length of the document summary string does not exceed 64 kilobytes.</td>
</tr>
<tr>
<td>longstring</td>
<td>The length of the document summary string can exceed 64 kilobytes.</td>
</tr>
<tr>
<td>data</td>
<td>Used only for internal document summary representation inside the index.</td>
</tr>
</tbody>
</table>

### 2.20.4.4 ST_compression

Referenced by: **CT_summaryField**

A simple type that specifies whether file compression SHOULD be used when storing document summaries in index files. A value of "on" means compression SHOULD be used.

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_compression">
  <xs:restriction base="xs:string">
    <xs:enumeration value="on"/>
    <xs:enumeration value="off"/>
  </xs:restriction>
</xs:simpleType>
```

For further details on the compression used, see the table in section 2.8.3.33.

### 2.21 ManagedPropertyBoosts.xml

Configuration parameters in the ManagedPropertyBoosts.xml are derived from the index schema and contain information about keyword rank boosts. ManagedPropertyBoosts.xml contains instruction on how much rank SHOULD be added by the search server if a keyword exists in an item in the result set of a search query.

The following table provides information about the file.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Middleware Protocol storage path</td>
<td>QRServer/webcluster/etc/qrserver/</td>
</tr>
<tr>
<td>Type of data</td>
<td>Configuration information derived from index schema</td>
</tr>
<tr>
<td>File format</td>
<td>XML schema file.</td>
</tr>
</tbody>
</table>

### 2.21.1 Global Elements

#### 2.21.1.1 field-boosts

The **field-boosts** element contains a list of rank profiles which have keyword rank boost information associated with them.
2.21.2 Global Attributes

None.

2.21.3 Complex Types

2.21.3.1 CT_FieldBoosts

Referenced by: field-boosts

A complex type that is a container for the rank profiles that have keyword rank boost specifications.

Child elements: CT_RankProfile

Attributes: None.

```xml
<xs:complexType name="CT_FieldBoosts">
  <xs:sequence>
    <xs:element name="rank-profile" type="CT_RankProfile" maxOccurs="unbounded" minOccurs="0" />
  </xs:sequence>
</xs:complexType>
```

2.21.3.2 CT_RankProfile

Referenced by: CT_FieldBoosts

A complex type that specifies the rank profile the underlying keyword rank adjustments apply to. The keyword rank boosts SHOULD be applied to all search queries sorted by this rank profile. The added keywords MUST not affect the recall of the search query.

Child elements:

A sequence of CT_BoostGroup elements.

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>The name of the rank profile.</td>
</tr>
<tr>
<td>index</td>
<td>The index of the rank profile in the rank.cf file (see section 2.11). This value is of the type ST_RankProfileIndex.</td>
</tr>
</tbody>
</table>

```xml
<xs:complexType name="CT_RankProfile">
  <xs:sequence>
    <xs:element name="boost" type="CT_BoostGroup" maxOccurs="unbounded" minOccurs="0" />
  </xs:sequence>
  <xs:attribute name="name" type="xs:string" use="required" /> 
  <xs:attribute name="index" type="ST_RankProfileIndex" use="required" /> 
</xs:complexType>
```
2.21.3.3 **CT_BoostGroup**

Referenced by: **CT_RankProfile**

This complex type is used to group multiple **CT_FieldBoost** elements by the amount which an item’s rank SHOULD be adjusted.

Child elements:

A sequence of **CT_FieldBoost** elements.

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>The amount of rank to increase or decrease an item's rank with if any of the keywords in the enclosed <strong>CT_FieldBoost</strong> elements exists in the specified managed property of an item.</td>
</tr>
</tbody>
</table>

```xml
<xs:complexType name="CT_BoostGroup">
  <xs:sequence>
    <xs:element name="field-boost" type="CT_FieldBoost" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="value" type="xs:int" use="required" />
</xs:complexType>
```

2.21.3.4 **CT_FieldBoost**

Referenced by: **CT_BoostGroup**

This complex type specifies the keyword (or sequence of keywords) which MUST exist in a specified managed property of an item for a rank adjustment to take place.

Child elements: None.

Attributes:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the managed property with which this keyword rank is associated.</td>
</tr>
<tr>
<td>keyword</td>
<td>Phrase to search for in the managed property of the item to decide whether or not to increase or decrease the item's rank. This can be one or more keywords, which MUST exist exactly as specified in the managed property.</td>
</tr>
</tbody>
</table>

```xml
<xs:complexType name="CT_FieldBoost">
  <xs:attribute name="name" type="xs:string" use="required" />
  <xs:attribute name="keyword" type="xs:string" use="required" />
</xs:complexType>
```

2.21.4 Simple Types

2.21.4.1 **ST_RankProfileIndex**

Referenced by: **CT_RankProfileIndex**
An integer with a value between 0 and 2147483647, which specifies the index of the rank profile in the rank.cf file (see section 2.11). The first rank profile in rank.cf has an index of 0, the second 1, and so on.

This simple type is defined as follows:

```xml
<xs:simpleType name="ST_RankProfileIndex">
  <xs:restriction base="xs:unsignedInt">
    <xs:minInclusive value="0"/>
    <xs:maxInclusive value="2147483647"/>
  </xs:restriction>
</xs:simpleType>
</xs:schema>
```

### 2.22 findexrc

findexrc is a static configuration file used by the indexing service. The following table provides information about the file.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Middleware Protocol storage path</td>
<td>RTSearch/webcluster/</td>
</tr>
<tr>
<td>Type of data</td>
<td>Implementation-specific configuration information.</td>
</tr>
<tr>
<td>File format</td>
<td>Name value text file.</td>
</tr>
</tbody>
</table>

This is an implementation-specific configuration file that the indexing service (see [MS-FSO] section 2.1.1.5) uses to configure how to index content. The static configuration specified here is the only supported configuration, and the configuration file MUST be exactly as specified in the following table. Other implementation SHOULD ignore the content of this file, and implement based on the specifications in [MS-FSIXDS].

The following table specifies the static configuration parameters that MUST be in the file.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hostname = localhost</td>
<td>Specifies the host name to be used internally in the indexing service. Value is not in use and MUST be ignored.</td>
</tr>
<tr>
<td>Dictformat = new</td>
<td>Not in use, and MUST be ignored.</td>
</tr>
<tr>
<td>threadedfusion</td>
<td>Specifies to generate boolocc, phraseocc, and posocc (see [MS-FSIXDS] section 2.1) files in parallel when indexing content. MUST be specified, but does not impact the search index output.</td>
</tr>
<tr>
<td>makeposocc</td>
<td>Parameter MUST be specified. Specifies that positional occurrence files are to be generated (see [MS-FSIXDS] section 2.1).</td>
</tr>
<tr>
<td>checkpointfiles = no</td>
<td>Parameter MUST be specified and set to &quot;no&quot;. Controls whether or not the indexing service deletes temporary files earlier in the indexing process.</td>
</tr>
<tr>
<td>syncfiles = no</td>
<td>Parameter MUST be specified and set to &quot;no&quot;. Controls whether or not to write files to disk so that an aborted indexing can be resumed.</td>
</tr>
<tr>
<td>compressboolocc</td>
<td>Parameter MUST be specified. Controls whether the Boolean occurrence files are compressed. Uncompressed Boolean occurrence files (see [MS-FSIXDS] section</td>
</tr>
</tbody>
</table>
### 2.23 template.rc

This file MUST exist and MUST be greater than 0 byte in size, but the content MUST be ignored. The file is referenced from the search_preload file (see section 2.16), and will as a consequence be downloaded to each search node in the system, even though the file is no longer in use.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Middleware Protocol storage path</td>
<td>RTSearch/webcluster/</td>
</tr>
<tr>
<td>Type of data</td>
<td>Implementation-specific configuration information.</td>
</tr>
<tr>
<td>File format</td>
<td>Fixed ignored text file</td>
</tr>
</tbody>
</table>

### 2.24 error.templ

This file MUST exist and MUST be greater than 0 byte in size, but the content MUST be ignored. The file is referenced from the search_preload file (see section 2.16), and will as a consequence be downloaded to each search node in the system, even though the file is no longer in use.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Middleware Protocol storage path</td>
<td>RTSearch/webcluster/templates/rtsearch/fdispatch_fhtml/</td>
</tr>
<tr>
<td>Type of data</td>
<td>Implementation-specific configuration information.</td>
</tr>
<tr>
<td>File format</td>
<td>Fixed ignored text file</td>
</tr>
</tbody>
</table>

### 2.25 footer.templ

This file MUST exist and MUST be greater than 0 byte in size, but the content MUST be ignored. The file is referenced from the search_preload file (see section 2.16), and will as a consequence be downloaded to each search node in the system, even though the file is no longer in use.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Middleware Protocol storage path</td>
<td>RTSearch/webcluster/templates/rtsearch/fdispatch_fhtml/</td>
</tr>
<tr>
<td>Type of data</td>
<td>Implementation-specific configuration information.</td>
</tr>
<tr>
<td>File format</td>
<td>Fixed ignored text file</td>
</tr>
</tbody>
</table>
### 2.26 header.templ

This file MUST exist and MUST be greater than 0 byte in size, but the content MUST be ignored. The file is referenced from the search_preload file (see section 2.16), and will as a consequence be downloaded to each search node in the system, even though the file is no longer in use.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Middleware Protocol storage path</td>
<td>RTSearch/webcluster/templates/rtsearch/fdispatch_fhtml/</td>
</tr>
<tr>
<td>Type of data</td>
<td>Implementation-specific configuration information.</td>
</tr>
<tr>
<td>File format</td>
<td>Fixed ignored text file</td>
</tr>
</tbody>
</table>

### 2.27 next.templ

This file MUST exist and MUST be greater than 0 byte in size, but the content MUST be ignored. The file is referenced from the search_preload file (see section 2.16), and will as a consequence be downloaded to each search node in the system, even though the file is no longer in use.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Middleware Protocol storage path</td>
<td>RTSearch/webcluster/templates/rtsearch/fdispatch_fhtml/</td>
</tr>
<tr>
<td>Type of data</td>
<td>Implementation-specific configuration information.</td>
</tr>
<tr>
<td>File format</td>
<td>Fixed ignored text file</td>
</tr>
</tbody>
</table>

### 2.28 nohits.templ

This file MUST exist and MUST be greater than 0 byte in size, but the content MUST be ignored. The file is referenced from the search_preload file (see section 2.16), and will as a consequence be downloaded to each search node in the system, even though the file is no longer in use.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Middleware Protocol storage path</td>
<td>RTSearch/webcluster/templates/rtsearch/fdispatch_fhtml/</td>
</tr>
<tr>
<td>Type of data</td>
<td>Implementation-specific configuration information.</td>
</tr>
<tr>
<td>File format</td>
<td>Fixed ignored text file</td>
</tr>
</tbody>
</table>

### 2.29 prev.templ

This file MUST exist and MUST be greater than 0 byte in size, but the content MUST be ignored. The file is referenced from the search_preload file (see section 2.16), and will as a consequence be downloaded to each search node in the system, even though the file is no longer in use.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Middleware Protocol storage path</td>
<td>RTSearch/webcluster/templates/rtsearch/fdispatch_fhtml/</td>
</tr>
</tbody>
</table>
### 2.30 result.templ

This file MUST exist and MUST be greater than 0 byte in size, but the content MUST be ignored. The file is referenced from the search_preload file (see section 2.16), and will as a consequence be downloaded to each search node in the system, even though the file is no longer in use.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type of data</strong></td>
<td>Implementation-specific configuration information.</td>
</tr>
<tr>
<td><strong>File format</strong></td>
<td>Fixed ignored text file</td>
</tr>
</tbody>
</table>

#### 2.31 templates.rc

This file MUST exist and MUST be greater than 0 byte in size, but the content MUST be ignored. The file is referenced from the search_preload file (see section 2.16), and will as a consequence be downloaded to each search node in the system, even though the file is no longer in use.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Configuration Middleware Protocol storage path</strong></td>
<td>RTSearch/webcluster/templates/rtsearch/fdispatch_fhtml/</td>
</tr>
<tr>
<td><strong>Type of data</strong></td>
<td>Implementation-specific configuration information.</td>
</tr>
<tr>
<td><strong>File format</strong></td>
<td>Fixed ignored text file</td>
</tr>
</tbody>
</table>
3 Structure Examples

In the following sections, the schema definition might differ from the processing rules imposed by the protocol. The XSD in this specification provides a base description of the file format. The text that introduces the XSD specifies additional restrictions that reflect protocol behavior. For example, the schema definition might allow for an element to be empty, null, or not present but the behavior of the protocol as specified restricts the same elements to being non-empty, present, and not null.

The structure examples in this section are based on the index schema abstract data model specified in section 1.3.2.

3.1 maptransform.xml

The example shows a data type configuration with one decimal data type specified in the schema. The example does not list elements that are fixed and mandatory.

Data type definitions for a decimal data type with two decimal places.

```xml
<?xml version="1.0" encoding="utf-8"?>
<transform-specification>
  <datatype-definitions>
    ...</datatype-definitions>
</transform-specification>
```

Number transformations for a set of numeric managed properties of type INT and DATETIME.

```xml
<field name="bconcrawltime" datatype="DATETIME"/>
<field name="bconprocessingtime" datatype="DATETIME"/>
<field name="bcondocdatetime" datatype="DATETIME"/>
<field name="bconsize" datatype="INT"/>
<field name="bconhwboost" datatype="INT"/>
<field name="bcondocrank" datatype="INT"/>
<field name="bconsiterank" datatype="INT"/>
<field name="bconurldepthrank" datatype="INT"/>
<field name="bconcreated" datatype="DATETIME"/>
```
3.2 fieldspec.xml

The example shows a definition of a set of managed properties enabled for full-text sorting and one rank profile that can be used for rank-based sorting.

The field list contains a set of fields enabled for full-text sort (sorttype set to "attv") and one rank profile named "default" (sorttype set to "rankprofile").

```xml
<?xml version="1.0" encoding="utf-8"?>
<fieldlist>
  <field name="title" sorttype="attv"/>
  <field name="crawltime" sorttype="attv"/>
  <field name="processingtime" sorttype="attv"/>
  <field name="docdatetime" sorttype="attv"/>
  <field name="size" sorttype="attv"/>
  <field name="hwboost" sorttype="attv"/>
  <field name="docrank" sorttype="attv"/>
  <field name="siterank" sorttype="attv"/>
  <field name="urldepthrank" sorttype="attv"/>
  <field name="created" sorttype="attv"/>
  <field name="lastmodifiedtime" sorttype="attv"/>
  <field name="default" sorttype="rankprofile"/>
</fieldlist>
```

3.3 configuration.attributes.xml

The example shows a refinement configuration for a string, an integer, and a datetime refiner. A sample string refiner for the managed property author. The cutoff for number of refinement bins evaluated in the query evaluation component is set to 1000. The sorting of the refiner is configured to be by frequency in descending order. A maximum of 100 refinement bins are returned.

```xml
<?xml version="1.0" encoding="utf-8"?>
<navigators>
  <navigator
    deephits="0"
    name="authornavigator"
    cutminbuckets="0"
    deep="yes"
    passive="no"
```
field="bavnauthor"
separator=""
cutmaxbuckets="1000"
cutfreq="0"
modifier="author"
type="string"
display=""
unit=""
multimode="needed">
  <score
    count="0"
    constant="1"
    buckets="0"
    entropy="1"
    offset="0"
    ratio="0"/>
  <string anchoring="complete">
    <sort by="frequency" order="descending"/>
    <filter buckets="100" frequency="1"/>
  </string>
</navigator>

An example integer navigator for the managed property named size. The distribution of the refinement bins is calculated such that approximately the same number of observations falls into each refinement bin. The maximum number of refinement bins is set to 4. The resolution is set to 1024 so that the boundaries of the refinement bins are multiples of 1 kilobyte. The divisor is set to 1024 so that the size is returned in kilobytes. The default value for the refiner is 0.

<navigator
  deephits="0"
  name="sizenavigator"
  deep="yes"
  passive="no"
  field="bavnsize"
  signed="yes"
  modifier="size"
  type="integer"
  display=""
  unit="">
  <integer>
    <discretize algorithm="equalfrequency">
      <equalfrequency intervals="4" resolution="1024"/>
    </discretize>
    <display divisor="1024">
      <first offset="0" format="Less than %s"/>
      <middle offset1="0" offset2="0" format="%s up to %s"/>
      <last offset="0" format="%s and up"/>
    </display>
  </integer>
  <score
    count="0"
    constant="1"
    buckets="0"
    entropy="1"
    offset="0"
    ratio="0"/>
  <ignore value="0"/>
An example datetime navigator for the managed property named `docdatetime`.

The distribution of the refinement bins is calculated such that approximately the same number of observations falls into each refinement bin. The maximum number of refinement bins is set to 4.

The resolution is set to 864000000000 so that the refinement bins are set up on day boundaries (24*60*60*100000000).

```xml
<navigator>
  <deephits="0"
   name="docdatetimenavigator"
   deep="yes"
   passive="no"
   field="bavndocdatetime"
   modifier="docdatetime"
   type="datetime"
   display=""
   unit="">
    <datetime>
      <integer>
        <discretize algorithm="equalfrequency">
          <equalfrequency intervals="4" resolution="864000000000"/>
        </discretize>
        <display divisor="1">
          <first offset="0" format="Before %s"/>
          <middle offset1="0" offset2="0" format="From %s to %s"/>
          <last offset="0" format="%s or later"/>
        </display>
      </integer>
    </datetime>
    <score>
      <count="0" constant="1" buckets="0" entropy="1" offset="0" ratio="0"/>
    </score>
  </navigator>
</navigators>
```

### 3.4 `fsearch.addon`

The example shows a configuration of the schema-dependent configuration parameters.

The index schema contains three managed properties that are configured for hit highlighted summary: `body`, `title`, and `notes`.

The default index is named `content`.

The index does not specify any latent attribute vectors.

Configuration of the three managed properties that are configured for a hit highlighted summary.
The **title** and **notes** properties are searchable as individual managed properties. The **body** property is not.

The **title** and **notes** properties have fallback to the managed property itself in case no hit highlighted summary can be created.

The **body** property has fallback to the managed property **teaser**.

```
title.matcher.indexes title;bt1bidxtitle;content;bcatcontent.bidxcontentlvl1
body.matcher.indexes content;bcatcontent.bidxcontentlvl1
notes.matcher.indexes notes;bt1bidxnotes;content;bcatcontent.bidxcontentlvl1
title.config.parent normalteaser
title.fallback0.field bsrcrtitle
title.fallback0.when always_process
body.config.parent normalteaser
body.fallback0.field bsumteaser
body.fallback0.when always
notes.config.parent normalteaser
notes.fallback0.field bsrcnotes
notes.fallback0.when always_process
```

Specifies the default index name.

```
juniper.config.default_index content
```

No attribute vectors are configured as "latent".

```
attributevectors.disable
```

### 3.5 indexConfig.xml

The example shows an index configuration derived from the index schema. Only the parts of the configuration file that are dependent on the actual schema are shown.

The synthetic text catalog. For brevity, the example shows only a subset of the managed properties. All managed properties have the same configuration in this section.

```xml
<catalog name="bt1" type="text" synthetic="yes" wildcard="yes">
  <context name="bcontentitle" type="simple"/>
  <context name="bcondescription" type="simple"/>
  <context name="bconanchortext" type="simple"/>
  ...
  <context name="bconprices" type="simple"/>
  <context name="bconextractedurls" type="simple"/>

  <index name="bidxtitle" phraseIndex="off" posIndex="on" prefixSearch="off">
    <contextRef name="bcontentitle"/>
    <alias name="title"/>
  </index>

  <index name="bidxdescription" phraseIndex="off" posIndex="on" prefixSearch="off">
    <contextRef name="bcondescription"/>
    <alias name="description"/>
  </index>
</catalog>
```
<catalog name="bidxcontentlvl1" catalogName="bcatcontent"/>

The default index is named "content".

This part specifies the generic parameters for a rank profile named "default". In this example, there are four managed properties that contribute to the static rank: **hwboost**, **docrank**, **siterank**, and **urldepthrank**. The stop-word threshold is set to 2,000,000. The proximity search threshold for **NEAR** and **ONEAR** operators is set to 200,000,000.

```
<rankProfileList>
  <rankProfile name="brpdefault" tuneFactor="1.00" tuneBias="0">
    <staticRankParameters>
      <qualityComponentList>
```
<qualityComponent attributeVector="batvhwboost" coefficient="1.000"/>
<qualityComponent attributeVector="batvdocrank" coefficient="1.000"/>
<qualityComponent attributeVector="batvsiterank" coefficient="1.000"/>
<qualityComponent attributeVector="batvurdepthrank" coefficient="1.000"/>
</qualityComponentList>
</staticRankParameters>
<dynamicRankParameters>
  binLow="0"
  binHigh="2000000"
  binSize="4294967295.00"
  posBinLow="0"
  posBinHigh="20000000"
  posBinSize="4294967295.00"
  xNearPosBinLow="0"
  xNearPosBinHigh="20000000"
  xNearPosBinSize="4294967295.00"
  superiorBoost="0"
  rankCutoff="0"
  rankCutoffAdvVal="0"
  firstOccProximity="yes"
  proximity="yes"
  phraseProximity="yes"
  proximityPairBeforeFirstOccProximityTriple="yes"
  proximityTripleBeforeFirstOccProximityQuad="yes"
  clampStaticRank="no">
Rank profile data for each context catalog. For the full-text context catalog, the implementation-specific rank model from the index schema gives the boost values.

<catalogRankList>
  <extNumOccBoostOnlyCatalog catalogName="anchortext" fileName="boost-tables/default_anchortext_extnumocccboost.tbl"/>
  <extNumOccBoostOnlyCatalog catalogName="assocqueries" fileName="boost-tables/default_assocqueries_extnumocccboost.tbl"/>
  <rankedCatalog catalogName="bcatcontent">
    <andBoost value="0"/>
    <orBoost value="500"/>
    <phraseBoost value="0"/>
    <rankBoost value="0"/>
    <anyBoost value="500"/>
    <nearBoost value="500"/>
    <orderedNearBoost value="500"/>
    <numOccBoost fileName="boost-tables/default_content_numocc_boost.tbl"/>
    <firstOccBoost fileName="boost-tables/default_content_firstocc_boost.tbl"/>
    <extNumOccBoost fileName="boost-tables/default_content_ext_numocc_boost.tbl"/>
    <proximityBoost fileName="boost-tables/default_content_proximity_boost_firstocc_fw_0.tbl" tableSet="0" firstOcc="yes" direction="forward"/>
    <proximityBoost fileName="boost-tables/default_content_proximity_boost_firstocc_bw_0.tbl" tableSet="0" firstOcc="yes" direction="backward"/>
    <proximityBoost fileName="boost-tables/default_content_proximity_boost_nofirstocc_fw_0.tbl" tableSet="0" firstOcc="no" direction="forward"/>
    <proximityBoost fileName="boost-tables/default_content_proximity_boost_nofirstocc_bw_0.tbl" tableSet="0" firstOcc="no" direction="backward"/>
</rankedCatalog>
</catalogRankList>
A set of attribute vectors, as specified in the index schema. The following attribute vectors correspond to the managed properties that are configured for full-text sorting. Note that signedValue is set to "yes" for signed numeric managed properties.

```
<attributeVectorList>
  <attributeVector name="batvtitle" type="string" multi="no" signedValue="no"/>
  <attributeVector name="batvcrawltime" type="int64" multi="no" signedValue="no"/>
  <attributeVector name="batvprocessingtime" type="int64" multi="no" signedValue="no"/>
  <attributeVector name="batvdocdatetime" type="int64" multi="no" signedValue="no"/>
  <attributeVector name="batvsize" type="int64" multi="no" signedValue="yes"/>
  <attributeVector name="batvhwboost" type="int64" multi="no" signedValue="yes"/>
  <attributeVector name="batvdocrank" type="int64" multi="no" signedValue="yes"/>
  <attributeVector name="batvsiterank" type="int64" multi="no" signedValue="yes"/>
  <attributeVector name="batvurldepthrank" type="int64" multi="no" signedValue="yes"/>
  <attributeVector name="batvcreated" type="int64" multi="no" signedValue="no"/>
  <attributeVector name="batvlastmodifiedtime" type="int64" multi="no" signedValue="no"/>
</attributeVectorList>
```

A set of attribute vectors, as specified in the index schema. The following attribute vectors correspond to the managed properties that have an associated refiner specified in the index schema. For this type of attribute vector, all values are treated as unsigned.

```
<attributeVector name="bavnauthor" type="string" multi="yes" signedValue="no"/>
<attributeVector name="bavnlanguages" type="string" multi="yes" signedValue="no"/>
```

---

[MS-FSSCFG] — v20120630
Search Configuration File Format

Copyright © 2012 Microsoft Corporation.

Release: July 16, 2012
The list of summary classes. For brevity, the example shows only a subset of the managed properties. In this example, only one input summary class is defined. This means that the initially configured index schema is not changed (only one generation).

```xml
<summaryClassList fieldTypeUsedForId="integer" defaultOutputClassName="servedcontent">
  <summaryClass name="content" type="in">
    <summaryField name="internalid" type="string" defaultValue=""/>
    <summaryField name="contentid" type="string" defaultValue=""/>
    ...
    <summaryField name="bsumurls" type="string" defaultValue=""/>
  </summaryClass>
  <summaryClass name="servedcontent" type="out">
    <summaryField name="internalid" type="string" defaultValue=""/>
    <summaryField name="contentid" type="string" defaultValue=""/>
    <summaryField name="bsumpersonnameteaser" type="string" defaultValue=""/>
    ...
    <summaryField name="ranklog" type="string" defaultValue=""/>
  </summaryClass>
</summaryClassList>
```

### 3.6 index.cf

The data in this configuration file is derived from indexConfig.xml and mirrors the features specified in that file.

The configuration for the full-text index field named "content". The configuration maps to the configuration as specified in section 2.8.5.3.1.

```xml
catalog bcatcontent type text
dictionary exact
```
The numeric context catalog lists the set of numeric managed properties defined in the index schema. The catalog includes 10 numeric managed properties.

catalog bi1 type integer 10
dictionary exact
contexts bconcrawltime bconprocessingtime bcondocdatetime bconsize bconhwboost bcondocrank bconsiterank bconurlldepthrank bconcreated bconlastmodifiedtime
index bidxcrawltime nopositions
contains bconcrawltime
mccontext bconcrawltime
index bidxprocessingtime nopositions
contains bconprocessingtime
mccontext bconprocessingtime
index bidxdocdatetime nopositions
contains bcondocdatetime
mccontext bcondocdatetime
index bidxsize nopositions
contains bconsize
mccontext bconsize
index bidxhwboost nopositions
contains bconhwboost
mccontext bconhwboost
index bidxdocrank nopositions
contains bcondocrank
mccontext bcondocrank
index bidxurldepthrank nopositions
contains bconurldepthrank
mccontext bconurldepthrank
index bidxcreated nopositions
contains bconcreated
mccontext bconcreated
index bidxlastmodifiedtime nopositions
contains bconlastmodifiedtime
mccontext bconlastmodifiedtime

The context catalog for synthetic text includes all managed properties for searchable text. The catalog includes 46 managed properties.
catalog btl type txsynthetic 46
dictionary exact
simplecontexts bcontitle bcondescription bconanchortext bconassocqueries bconkeywords bconcontenttype bconformat bconlanguage bconlanguages bconcharset
simplecontexts bconurls bcondomain bcontld bconpath bconurlkeywords bcondocacl bcondocaclystemid bconauthor bconcreatedby bcondocumentid bcondocsubject bconnotes bconsiteid
simplecontexts bconsitename bconsitetitle bconspsiteurl bconstatus bconcrawledpropertiescontent bconcrawledpropertynames bconcompanies bconlocations bconpersonnames bconconcepts simplecontexts bconemails bcontaxonomy bcondates bcontimes bconprices bconextractedurls

The property indexes in the context catalog named "bt1" correspond to the \texttt{simplecontexts} specified previously. For brevity, the example shows only a subset.

index bidxtitle withprefix
contains bcontitle
mccontext bcontitle
index bidxdescription withprefix
contains bcondescription
mccontext bcondescription
index bidxanchortext withprefix
contains bconanchortext
mccontext bconanchortext
index bidxassocqueries withprefix
contains bconassocqueries
mccontext bconassocqueries
index bidxkeywords withprefix
contains bconkeywords
mccontext bconkeywords
index bidxcontenttype withprefix
contains bconcontenttype
mccontext bconcontenttype

Specification of the default index that indicates the property index for the first field importance level.

defaultindex bcatcontent.bidxcontentlvl1

Alias definitions for the managed properties. For brevity, the example shows only a subset.

alias content bcatcontent.bidxcontentlvl1
alias title bt1.bidxtitle
alias description bt1.bidxdescription
alias anchortext bt1.bidxanchortext
alias assocqueries bt1.bidxassocqueries
alias keywords bt1.bidxkeywords
alias contenttype bt1.bidxcontenttype
alias format bt1.bidxformat
alias language bt1.bidxlanguage
alias languages bt1.bidxlanguages
alias charset bt1.bidxcharset

The list of attribute vectors corresponds to the definition in indexConfig.xml.
attributevector batvtitle string false false
attributevector batvcrawltime int64 false false
attributevector batvprocessintime int64 false false
attributevector batvdocdatetime int64 false false
attributevector batvsize int64 false true
attributevector batvhboost int64 false true
attributevector batvdocrank int64 false true
attributevector batvsiterank int64 false true
attributevector batvurldepthrank int64 false true
attributevector batvcreated int64 false false
attributevector batvlastmodifiedtime int64 false false
attributevector batvauthor string true false
attributevector batvlanguages string true false
attributevector batvcompanies string true false
attributevector batvlocations string true false
attributevector batvpersonnames string true false
attributevector batvconcepts string true false
attributevector batvemails string true false
attributevector batvdates string true false
attributevector batvtimes string true false
attributevector batvextractedurls string true false
attributevector batvprices string true false
attributevector batvformat string true false
attributevector batvcrawledpropertynames string true false
attributevector batvsize int64 true true
attributevector batvdocdatetime int64 true false

The drilling information indicates the four field importance levels defined.

link bcatcontent.bidxcontentlvl1 bcatcontent.bidxcontentlvl2
link bcatcontent.bidxcontentlvl2 bcatcontent.bidxcontentlvl3
link bcatcontent.bidxcontentlvl3 bcatcontent.bidxcontentlvl4

3.7 fixml_mappings.xml

This example provides the mapping of managed properties for the generation of the FIXML data object. It contains all managed properties and internal properties defined in the index schema.

The mapping entries for the managed property title. The first mapping entry defines the mapping to the synthetic context catalog. For the title field, the maximum size for the source managed property is set to 1024 kilobytes. The second mapping entry defines the mapping to the full-text index field named "content". The title is mapped to the reserved property context named "bconf7". Refer to section 2.8.5.3.1 for more information. The third mapping entry defines the mapping to the source for generation of a hit highlighted summary.

```xml
<?xml version="1.0" encoding="utf-8"?>
<mappings sclass="content">
<map
  src="title"
  dstcatalog="bt1"
  dst="bcontitle"
  phrasebreak="yes"
  fieldseparationlength="0"
  maxsize="1024"
  phraseseparator=""
```
The mapping entries for the managed property docdatetime. The first mapping entry defines the mapping to the numeric context catalog. The second mapping entry defines the mapping to the document summary. The third mapping entry defines the mapping to the attribute vector for full-text sorting. The fourth mapping entry defines the mapping to the attribute vector for the associated refiner. The default value is set to 0.
3.8 rank.cf

The data in this configuration file is derived from indexConfig.xml and mirrors the features specified in that file.

The rank profile named "default".

```plaintext
rankprofile brpdefault

tunefactor 1.00
tunebias 0
qualitycomponent batvhwboost 1.000
qualitycomponent batvdocrank 1.000
qualitycomponent batvsiteRank 1.000
qualitycomponent batvurldepthrank 1.000
dynamicranking on
binlow 0
binho 0
binsize 4294967295.00
posbinlow 0
posbinhigh 2000000
posbinsize 4294967295.00
xnearposbinlow 0
xnearposbinhigh 20000000
xnearposbinsize 4294967295.00
superiorboost 0
rankcutoff 0
rankcutoffadval 0
firstoccproximity yes
proximity yes
phraseproximity yes
proximitypairbeforefirstoccproximitytriple yes
proximitytriplebeforefirstoccproximityquad yes
clampstaticrank no

extnumoccboosonly anchortext yes
extnumoccboost anchortext boost-tables/default_anchortext_extnumoccboost.tbl

extnumoccboostonly assocqueries yes
extnumoccboost assocqueries boost-tables/default_assocqueries_extnumoccboost.tbl

The boost configuration for a rank profile named "content".

```plaintext
andboost bcatcontent 0
orboost bcatcontent 500
phraseboost bcatcontent 0
rankboost bcatcontent 0
anyboost bcatcontent 500
nearboost bcatcontent 500
orderednearboost bcatcontent 500
numoccboost bcatcontent boost-tables/default_content_numocc_boost.tbl
firstoccboost bcatcontent boost-tables/default_content_firstocc_boost.tbl
extnumoccboost bcatcontent boost-tables/default_content_ext_numocc_boost.tbl
```
This example provides the configuration of item and query processing features for the managed properties defined in the index schema.

The example shows field properties for a subset of the managed properties. The example lists the configuration for the managed properties title, body, teaser, description, and anchortext.

The title and body properties are configured for a hit highlighted summary (type="dynamic"). The teaser property is configured for returning a document summary. The description and anchortext properties are not configured for returning a document summary.

All the managed properties are configured for lemmatization and language-specific tokenization.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<field-properties default-index="content">
  <field alias="title" kind="field" indexed="yes" type="string" boundary="yes" wildcard="full">
    <language-tokenization lemmatization="yes"/>
    <result type="dynamic" max-size="64"/>
  </field>

  <field alias="body" kind="field" indexed="no" type="string" boundary="yes" wildcard="full">
    <language-tokenization lemmatization="yes"/>
    <result type="dynamic" max-size="1024"/>
  </field>

  <field alias="teaser" kind="field" indexed="no" type="string" boundary="yes" wildcard="full">
    <language-tokenization lemmatization="yes"/>
    <result type="dynamic" max-size="64"/>
  </field>
</field-properties>
```
3.10 Boost Table Files

This example provides a set of sample boost table files for relevance boosting.

**Term occurrence boost file**: default_content_numocc_boost.tbl (showing only the first and last part of boost values in the file):

7000
8250
9500
10750
12000
12416
...
19887
19899
19912
19924
19937
19949
19962
19974
19987
19999

**Proximity boost file** - default_content_proximity_boost_nofirstocc_fw_0.tbl:

1000
950
900
850
800
750
700
650
600
550
500
450
400
Boost table file for global term frequency - default_content_divtable.tbl:

3.11 rankspace.xml

This file lists a single rank profile named "default" that is defined in the index schema.

```xml
<rankspace>
  <ranking name="default" description="BLISS generated" descendingIndex="0"/>
</rankspace>
```
3.12 resultspace.xml

This file contains all document summaries in the output summary class. For brevity, the example shows only a subset of the document summary fields.

```xml
<?xml version="1.0" encoding="utf-8"?>
<resultspace>
  <result-view index="0" name="DATASEARCHDEFAULT">
    <field type="string" name="title"/>
    <field type="string" name="body"/>
    <field type="string" name="teaser"/>
    <field type="string" name="contenttype"/>
    <field type="string" name="format"/>
    <field type="string" name="language"/>
    <field type="string" name="languages"/>
    <field type="string" name="charset"/>
    <field type="string" name="urls"/>
    ...
    <field type="string" name="locationteaser"/>
    <field type="string" name="personnameteaser"/>
  </result-view>
</resultspace>
```

3.13 summary.cf

The data in this configuration file is derived from indexConfig.xml and mirrors the set of managed properties and internal properties specified in that file.

This file contains all document summaries in the summary classes. The input summary class is named **content**. For brevity, the example shows only a subset of the document summaries.

```plaintext
idtype integer
class content id 0
field internalid type string
field contentid type string
field contentids type string
field collection type string
field bsumaccount type string
field bsumassignedto type string
field bsumauthor type string
field bsumurls type string
```

The output summary class is named **servedcontent**. For brevity, the example shows only a subset of the summary fields.

```plaintext
class servedcontent id 1073741823
field internalid type string
field contentid type string
field contentids type string
field collection type string
field bsumtitle type longstring
```
field bsumpersonnameteaser type string
field ranklog type string

### 3.14 summaryclasses.xml

The data in this configuration file is derived from indexConfig.xml and mirrors the set of managed properties and internal properties specified in that file.

For brevity, the example shows only a subset of the document summaries.

The **bsumbody** compression is activated when the document summary data is stored in the index data structures.

```xml
<?xml version="1.0" encoding="utf-8"?>
<summary-input-classes>
  <summaryClass name="content" type="in">
    <summaryField name="bsumaccount" type="string" compression="off"/>
    <summaryField name="bsumassignedto" type="string" compression="off"/>
    <summaryField name="bsumauthor" type="string" compression="off"/>
    <summaryField name="bsumbody" type="longstring" compression="on"/>
  </summaryClass>
</summary-input-classes>
```
4 Security Considerations

None.
5 Appendix A: Full XML Schemas

For ease of implementation, this section provides the full Worldwide Web Consortium (W3C) XML schemas for the elements, attributes, complex types, and simple types described in the preceding sections.

5.1 maptransform.xsd

<?xml version="1.0" encoding="utf-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
<!-- ******************** Global elements ****************--- -->
<xs:element name="transform-specification" type="CT_transform-specification"/>

<!-- ******************** Complex types ****************--- -->
<xs:complexType name="CT_transform-specification">
<xs:sequence>
<xs:element name="datatype-definitions" type="CT_datatype-definitions"/>
<xs:element name="number-transformations" type="CT_number-transformations"/>
</xs:sequence>
</xs:complexType>

<xs:complexType name="CT_datatype-definitions">
<xs:sequence>
<xs:element name="datatype" type="CT_datatype" minOccurs="1" maxOccurs="unbounded"/>
</xs:sequence>
</xs:complexType>

<xs:complexType name="CT_datatype">
<xs:attribute name="name" type="xs:string" use="required"/>
<xs:attribute name="offsetbits" type="ST_offsetbits" use="required"/>
<xs:attribute name="signbits" type="ST_signbits" use="required"/>
<xs:attribute name="mantissabits" type="ST_mantissabits" use="required"/>
<xs:attribute name="expbase" type="ST_expbase" use="required"/>
<xs:attribute name="decimalplaces" type="ST_decimalplaces" default="0"/>
<xs:attribute name="toint" type="ST_toint"/>
</xs:complexType>

<xs:complexType name="CT_number-transformations">
<xs:sequence>
<xs:element name="field" type="CT_field" minOccurs="1" maxOccurs="unbounded"/>
</xs:sequence>
</xs:complexType>

<xs:complexType name="CT_field">
<xs:attribute name="name" type="xs:string" use="required"/>
<xs:attribute name="datatype" type="xs:string"/>
</xs:complexType>

<!-- ******************** Simple types ****************--- -->
<xs:simpleType name="ST_offsetbits">
</xs:simpleType>

[MS-FSSCFG] — v20120630
Search Configuration File Format

Copyright © 2012 Microsoft Corporation.
Release: July 16, 2012
5.2 fieldspec.xsd

<?xml version="1.0" encoding="utf-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<!-- *********************** Global elements *********************** -->

<xs:element name="fieldlist" type="CT_fieldlist"/>

</xs:schema>
<xs:complexType name="CT_fieldlist">
  <xs:sequence>
    <xs:element name="field" type="CT_field" minOccurs="1" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="CT_field">
  <xs:attribute name="name" type="xs:string" use="required"/>
  <xs:attribute name="sorttype" type="ST_sorttype"/>
</xs:complexType>

<!-- *********************** Simple types *********************** -->

<xs:simpleType name="ST_sorttype">
  <xs:restriction base="xs:string">
    <xs:enumeration value="attv"/>
    <xs:enumeration value="rankprofile"/>
  </xs:restriction>
</xs:simpleType>

</xs:schema>

5.3 configuration.attributes.xsd

<?xml version="1.0" encoding="utf-8" ?>
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
  xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <!-- *********************** Global elements *********************** -->
  <xs:element name="navigators" type="CT_navigators"/>

  <!-- *********************** Complex types *********************** -->
  <xs:complexType name="CT_navigators">
    <xs:sequence>
      <xs:element maxOccurs="unbounded" name="navigator" type="CT_navigator"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="CT_navigator">
    <xs:all>
      <xs:element name="datetime" type="CT_datetimeNav" minOccurs="0" maxOccurs="1"/>
      <xs:element name="integer" type="CT_numericNav" minOccurs="0" maxOccurs="1"/>
      <xs:element name="double" type="CT_numericNav" minOccurs="0" maxOccurs="1"/>
      <xs:element name="fixedpoint" type="CT_fixedpoint" minOccurs="0" maxOccurs="1"/>
      <xs:element name="string" type="CT_stringNav" minOccurs="0" maxOccurs="1"/>
      <xs:element name="score" type="CT_score" minOccurs="1" maxOccurs="1"/>
    </xs:all>
    <xs:attribute name="deephits" type="xs:int" use="required"/>
    <xs:attribute name="name" type="xs:string" use="required"/>
  </xs:complexType>
</xs:schema>
<xs:attribute name="cutminbuckets" type="xs:int" use="optional"/>
<xs:attribute name="deep" type="ST_yesno" use="required"/>
<xs:attribute name="field" type="xs:string" use="required"/>
<xs:attribute name="cutfreq" type="ST_alwaysZero" use="optional"/>
<xs:attribute name="field" type="xs:string" use="required"/>
<xs:attribute name="modifier" type="xs:string" use="required"/>
<xs:attribute name="type" type="ST_type" use="required"/>
<xs:attribute name="display" type="xs:string" use="required"/>
<xs:attribute name="unit" type="xs:string" use="required"/>
<xs:attribute name="multimode" type="ST_multimode" use="optional"/>
<xs:attribute name="signed" type="ST_yesno" use="optional"/>
</xs:complexType>

<xs:complexType name="CT_datetimeNav">
  <xs:sequence>
    <xs:element name="integer" type="CT_numericNav" minOccurs="1" maxOccurs="1"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="CT_fixedpoint">
  <xs:sequence>
    <xs:element name="integer" type="CT_numericNav" minOccurs="1" maxOccurs="1"/>
  </xs:sequence>
  <xs:attribute name="decimals" type="xs:int" use="required"/>
</xs:complexType>

<xs:complexType name="CT_stringNav">
  <xs:sequence>
    <xs:element name="sort" type="CT_sort"/>
    <xs:element name="filter" type="CT_filter"/>
  </xs:sequence>
  <xs:attribute name="anchoring" type="ST_anchoring" use="required"/>
</xs:complexType>

<xs:complexType name="CT_numericNav">
  <xs:sequence>
    <xs:element name="discretize" type="CT_discretize"/>
    <xs:element name="display" type="CT_display"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="CT_discretize">
  <xs:choice>
    <xs:element name="equalfrequency" type="CT_equalfrequency"/>
    <xs:element name="rangedivision" type="CT_rangedivision"/>
    <xs:element name="equalwidth" type="CT_equalwidth"/>
  </xs:choice>
  <xs:attribute name="algorithm" type="ST_algorithm" use="required"/>
</xs:complexType>

<xs:complexType name="CT_equalfrequency">
  <xs:attribute name="intervals" type="xs:int" use="required"/>
  <xs:attribute name="resolution" type="xs:int" use="required"/>
</xs:complexType>

<xs:complexType name="CT_equalwidth">
  <xs:attribute name="resolution" type="xs:int" use="required"/>
</xs:complexType>
<xs:complexType name="CT_rangedivision">
  <xs:attribute name="intervals" type="xs:int" use="required"/>
  <xs:attribute name="resolution" type="xs:int" use="required"/>
</xs:complexType>

<xs:complexType name="CT_display">
  <xs:sequence>
    <xs:element name="first" type="CT_firstLast"/>
    <xs:element name="middle" type="CT_middle"/>
    <xs:element name="last" type="CT_firstLast"/>
  </xs:sequence>
  <xs:attribute name="divisor" type="xs:float" use="required"/>
</xs:complexType>

<xs:complexType name="CT_firstLast">
  <xs:attribute name="offset" type="xs:int" use="required"/>
  <xs:attribute name="format" type="xs:string" use="required"/>
</xs:complexType>

<xs:complexType name="CT_middle">
  <xs:attribute name="offset1" type="xs:int" use="required"/>
  <xs:attribute name="offset2" type="xs:int" use="required"/>
  <xs:attribute name="format" type="xs:string" use="required"/>
</xs:complexType>

<xs:complexType name="CT_sort">
  <xs:attribute name="by" type="ST_by" use="required"/>
  <xs:attribute name="order" type="ST_order" use="required"/>
</xs:complexType>

<xs:complexType name="CT_filter">
  <xs:attribute name="buckets" type="xs:integer" use="required"/>
  <xs:attribute name="frequency" type="xs:integer" use="required"/>
</xs:complexType>

<xs:complexType name="CT_score">
  <xs:attribute name="count" type="ST_alwaysZero" use="required"/>
  <xs:attribute name="constant" type="ST_alwaysOne" use="required"/>
  <xs:attribute name="buckets" type="ST_alwaysZero" use="required"/>
  <xs:attribute name="entropy" type="ST_alwaysOne" use="required"/>
  <xs:attribute name="offset" type="ST_alwaysZero" use="required"/>
  <xs:attribute name="ratio" type="ST_alwaysZero" use="required"/>
</xs:complexType>

<xs:simpleType name="ST_type">
  <xs:restriction base="xs:string">
    <xs:enumeration value="string"/>
    <xs:enumeration value="datetime"/>
    <xs:enumeration value="integer"/>
    <xs:enumeration value="float"/>
    <xs:enumeration value="fixedpoint"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="ST_multimode">
</xs:simpleType>
<xs:restriction base="xs:string">
  <xs:enumeration value="needed"/>
</xs:restriction>
</xs:simpleType>

<xs:simpleType name="ST_anchoring">
  <xs:restriction base="xs:string">
    <xs:enumeration value="auto"/>
    <xs:enumeration value="none"/>
    <xs:enumeration value="complete"/>
    <xs:enumeration value="prefix"/>
    <xs:enumeration value="suffix"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="ST_algorithm">
  <xs:restriction base="xs:string">
    <xs:enumeration value="equalfrequency"/>
    <xs:enumeration value="equalwidth"/>
    <xs:enumeration value="rangedivision"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="ST_by">
  <xs:restriction base="xs:string">
    <xs:enumeration value="auto"/>
    <xs:enumeration value="name"/>
    <xs:enumeration value="frequency"/>
    <xs:enumeration value="number"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="ST_order">
  <xs:restriction base="xs:string">
    <xs:enumeration value="ascending"/>
    <xs:enumeration value="descending"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="ST_alwaysOne">
  <xs:restriction base="xs:string">
    <xs:enumeration value="1"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="ST_alwaysZero">
  <xs:restriction base="xs:string">
    <xs:enumeration value="0"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="ST_yesno">
  <xs:restriction base="xs:string">
    <xs:enumeration value="yes"/>
    <xs:enumeration value="no"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="ST_alwaysno"/>
5.4  indexConfig.xsd

```xml
<?xml version="1.0" encoding="utf-8" ?>
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
  xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <!-- ******************** Global elements ********************* -->
  <xs:element name="FastIndexingConfig" type="CT_FastIndexingConfig"/>
  <!-- **************************** Complex types ************************-->  
  <xs:complexType name="CT_FastIndexingConfig">
    <xs:sequence>
      <xs:element name="catalogList" type="CT_catalogList"/>
      <xs:element name="defaultIndex" type="CT_defaultIndex"/>
      <xs:element name="staticRankClassList" type="CT_staticRankClassList"/>
      <xs:element name="rankProfileList" type="CT_rankProfileList"/>
      <xs:element name="attributeVectorList" type="CT_attributeVectorList"/>
      <xs:element name="summaryClassList" type="CT_summaryClassList"/>
      <xs:element name="summaryFieldOverrideList" type="CT_summaryFieldOverrideList"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="CT_catalogList">
    <xs:sequence>
      <xs:element maxOccurs="unbounded" name="catalog" type="CT_catalog"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="CT_catalog">
    <xs:sequence>
      <xs:element maxOccurs="unbounded" name="context" type="CT_context"/>
      <xs:element maxOccurs="unbounded" name="index" type="CT_index"/>
    </xs:sequence>
    <xs:attribute name="name" type="xs:string" use="required"/>
    <xs:attribute name="type" type="ST_catalogType" use="required"/>
    <xs:attribute name="synthetic" type="ST_yesno" use="required"/>
    <xs:attribute name="wildcard" type="ST_yesno" use="required"/>
  </xs:complexType>
  <xs:complexType name="CT_context">
    <xs:attribute name="name" type="xs:string" use="required"/>
    <xs:attribute name="type" type="ST_contextType" use="required"/>
  </xs:complexType>
  <xs:complexType name="CT_index">
    <xs:attribute name="name" type="xs:string" use="required"/>
    <xs:attribute name="type" type="ST_indexType" use="required"/>
  </xs:complexType>
</xs:schema>
```
<xs:element minOccurs="0" name="alias" type="CT_alias"/>
</xs:sequence>
<xs:attribute name="name" type="xs:string" use="required"/>
<xs:attribute name="SubStringSearch" type="ST_SubstringRange" use="optional"/>
<xs:attribute name="phraseIndex" type="ST_onoff" use="required"/>
<xs:attribute name="posIndex" type="ST_onoff" use="required"/>
<xs:attribute name="prefixSearch" type="ST_alwaysOff" use="required"/>
<xs:attribute name="drillSubIndex" type="xs:string" use="optional"/>
</xs:complexType>
<xs:complexType name="CT_contextRef">
<xs:attribute name="name" type="xs:string" use="required"/>
</xs:complexType>
<xs:complexType name="CT_alias">
<xs:attribute name="name" type="xs:string" use="optional"/>
</xs:complexType>
<xs:complexType name="CT_defaultIndex">
<xs:attribute name="indexName" type="xs:string" use="required"/>
<xs:attribute name="catalogName" type="xs:string" use="required"/>
</xs:complexType>
<xs:complexType name="CT_staticRankClassList">
<xs:sequence>
<xs:element name="staticRankClass" type="CT_staticRankClass"/>
</xs:sequence>
<xs:attribute name="bitsUsedForId" type="ST_alwaysZero" use="required"/>
</xs:complexType>
<xs:complexType name="CT_staticRankClass">
<xs:sequence>
<xs:element name="rankField" type="CT_rankField"/>
</xs:sequence>
<xs:attribute name="name" type="ST_dummy" use="required"/>
</xs:complexType>
<xs:complexType name="CT_rankField">
<xs:attribute name="name" type="ST_dummyfield" use="required"/>
<xs:attribute name="bitsUsed" type="ST_always32" use="required"/>
<xs:attribute name="defaultValue" type="ST_alwaysZero" use="required"/>
</xs:complexType>
<xs:complexType name="CT_rankProfileList">
<xs:sequence>
<xs:element name="rankProfile" type="CT_rankProfile"/>
</xs:sequence>
</xs:complexType>
<xs:complexType name="CT_rankProfile">
<xs:sequence>
<xs:element name="staticRankParameters" type="CT_staticRankParameters"/>
<xs:element name="dynamicRankParameters" type="CT_dynamicRankParameters"/>
<xs:element name="freshnessBoostParameters" type="CT_freshnessBoostParameters" minOccurs="0"/>
</xs:sequence>
<xs:attribute name="name" type="xs:string" use="required"/>
<xs:attribute name="tuneFactor" type="ST_tuneFactor" use="required"/>
<xs:attribute name="tuneBias" type="ST_alwaysZero" use="required"/>
<xs:complexType name="CT_staticRankParameters">
  <xs:sequence>
    <xs:element name="qualityComponentList" type="CT_qualityComponentList"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="CT_qualityComponentList">
  <xs:sequence>
    <xs:element maxOccurs="unbounded" name="qualityComponent" type="CT_qualityComponent"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="CT_qualityComponent">
  <xs:attribute name="attributeVector" type="xs:string" use="required"/>
  <xs:attribute name="coefficient" type="xs:decimal" use="required"/>
</xs:complexType>

<xs:complexType name="CT_dynamicRankParameters">
  <xs:sequence>
    <xs:element name="catalogRankList" type="CT_catalogRankList"/>
  </xs:sequence>
  <xs:attribute name="binLow" type="xs:unsignedInt" use="required"/>
  <xs:attribute name="binHigh" type="xs:unsignedInt" use="required"/>
  <xs:attribute name="binSize" type="xs:decimal" use="required"/>
  <xs:attribute name="posBinLow" type="xs:unsignedInt" use="required"/>
  <xs:attribute name="posBinHigh" type="xs:unsignedInt" use="required"/>
  <xs:attribute name="posBinSize" type="xs:decimal" use="required"/>
  <xs:attribute name="xNearPosBinLow" type="xs:unsignedInt" use="required"/>
  <xs:attribute name="xNearPosBinHigh" type="xs:unsignedInt" use="required"/>
  <xs:attribute name="xNearPosBinSize" type="xs:decimal" use="required"/>
  <xs:attribute name="superiorBoost" type="xs:unsignedInt" use="required"/>
  <xs:attribute name="rankCutoff" type="xs:unsignedByte" use="required"/>
  <xs:attribute name="rankCutoffAdvVal" type="xs:unsignedByte" use="required"/>
  <xs:attribute name="firstOccProximity" type="ST_yesno" use="required"/>
  <xs:attribute name="proximity" type="ST_yesno" use="required"/>
  <xs:attribute name="phraseProximity" type="ST_yesno" use="required"/>
  <xs:attribute name="proximityPairBeforeFirstOccProximityTriple" type="ST_yesno" use="required"/>
  <xs:attribute name="proximityTripleBeforeFirstOccProximityQuad" type="ST_yesno" use="required"/>
  <xs:attribute name="clampStaticRank" type="ST_yesno" use="required"/>
</xs:complexType>

<xs:complexType name="CT_catalogRankList">
  <xs:sequence>
    <xs:element maxOccurs="unbounded" name="extNumOccBoostOnlyCatalog" type="CT_extNumOccBoostOnlyCatalog"/>
    <xs:element name="rankedCatalog" type="CT_rankedCatalog"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="CT_extNumOccBoostOnlyCatalog">
  <xs:attribute name="catalogName" type="xs:string" use="required"/>
  <xs:attribute name="fileName" type="xs:string" use="required"/>
</xs:complexType>
<xs:complexType name="CT_rankedCatalog">
  <xs:sequence>
    <xs:element name="andBoost" type="CT_boostValue"/>
    <xs:element name="orBoost" type="CT_boostValue"/>
    <xs:element name="phraseBoost" type="CT_boostValue"/>
    <xs:element name="rankBoost" type="CT_boostValue"/>
    <xs:element name="anyBoost" type="CT_boostValue"/>
    <xs:element name="nearBoost" type="CT_boostValue"/>
    <xs:element name="orderedNearBoost" type="CT_boostValue"/>
    <xs:element name="numOccBoost" type="CT_occBoost"/>
    <xs:element name="firstOccBoost" type="CT_occBoost"/>
    <xs:element name="maxOccur" type="unbounded" name="proximityBoost"
      type="CT_proximityBoost"/>
    <xs:element name="divTableBoost" type="CT_divTableBoost"/>
    <xs:element name="contextBoostList" type="CT_contextBoostList"/>
  </xs:sequence>
  <xs:attribute name="catalogName" type="xs:string" use="required"/>
</xs:complexType>

<xs:complexType name="CT_boostValue">
  <xs:attribute name="value" type="xs:unsignedInt" use="required"/>
</xs:complexType>

<xs:complexType name="CT_occBoost">
  <xs:attribute name="fileName" type="xs:string" use="required"/>
</xs:complexType>

<xs:complexType name="CT_proximityBoost">
  <xs:attribute name="fileName" type="xs:string" use="required"/>
  <xs:attribute name="tableSet" type="xs:unsignedByte" use="required"/>
  <xs:attribute name="firstOcc" type="ST_yesno" use="required"/>
  <xs:attribute name="direction" type="ST_direction" use="required"/>
</xs:complexType>

<xs:complexType name="CT_divTableBoost">
  <xs:attribute name="fileName" type="xs:string" use="required"/>
</xs:complexType>

<xs:complexType name="CT_contextBoostList">
  <xs:sequence>
    <xs:element maxOccurs="unbounded" name="contextBoost"
      type="CT_contextBoost"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="CT_contextBoost">
  <xs:attribute name="contextName" type="xs:string" use="required"/>
  <xs:attribute name="value" type="xs:unsignedInt" use="required"/>
  <xs:attribute name="pairValue" type="xs:unsignedInt" use="required"/>
  <xs:attribute name="tripleValue" type="xs:unsignedInt" use="required"/>
  <xs:attribute name="quadValue" type="xs:unsignedInt" use="required"/>
</xs:complexType>

<xs:complexType name="CT_freshnessBoostParameters">
  <xs:sequence>
    <xs:element name="freshnessBoostFileRef" type="CT_freshnessBoostFileRef"/>
    <xs:element name="freshnessBoostDateTimeResolution" type="CT_freshnessBoostDateTimeResolution"/>
  </xs:sequence>
</xs:complexType>
<xs:element name="freshnessBoostCoefficient" type="CT_freshnessBoostCoefficient"/>
</xs:sequence>
</xs:complexType>

<xs:complexType name="CT_freshnessBoostFileRef">
  <xs:attribute name="name" type="xs:string" use="required"/>
</xs:complexType>

<xs:complexType name="CT_freshnessBoostDateTimeResolution">
  <xs:attribute name="value" type="xs:string" use="required"/>
</xs:complexType>

<xs:complexType name="CT_freshnessBoostCoefficient">
  <xs:attribute name="value" type="xs:unsignedByte" use="required"/>
</xs:complexType>

<xs:complexType name="CT_attributeVectorList">
  <xs:sequence>
    <xs:element minOccurs="0" maxOccurs="unbounded" name="attributeVector" type="CT_attributeVector"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="CT_attributeVector">
  <xs:attribute name="name" type="xs:string" use="required"/>
  <xs:attribute name="type" type="ST_attributeTypes" use="required"/>
  <xs:attribute name="multi" type="ST_yesno" use="required"/>
  <xs:attribute name="signedValue" type="ST_yesno" use="required"/>
  <xs:attribute name="alphaSortPath" type="xs:string" use="optional"/>
  <xs:attribute name="alphaSortMasterFile" type="xs:string" use="optional"/>
</xs:complexType>

<xs:complexType name="CT_summaryClassList">
  <xs:sequence>
    <xs:element maxOccurs="unbounded" name="summaryClass" type="CT_summaryClass"/>
  </xs:sequence>
  <xs:attribute name="fieldTypeUsedForId" type="ST_alwaysInteger" use="required"/>
  <xs:attribute name="defaultOutputClassName" type="xs:string" use="required"/>
</xs:complexType>

<xs:complexType name="CT_summaryClass">
  <xs:sequence>
    <xs:element maxOccurs="unbounded" name="summaryField" type="CT_summaryField"/>
  </xs:sequence>
  <xs:attribute name="name" type="xs:string" use="required"/>
  <xs:attribute name="type" type="ST_summaryClassTypes" use="required"/>
</xs:complexType>

<xs:complexType name="CT_summaryField">
  <xs:attribute name="name" type="xs:string" use="required"/>
  <xs:attribute name="type" type="ST_summaryFieldTypes" use="required"/>
  <xs:attribute name="defaultValue" type="xs:string" use="required"/>
  <xs:attribute name="compression" type="ST_onoff" use="optional"/>
</xs:complexType>
<xs:complexType name="CT_summaryFieldOverrideList">
  <xs:sequence>
    <xs:choice maxOccurs="unbounded">
      <xs:element name="overrideWithRankLog" type="CT_overrideWithRankLog"/>
      <xs:element name="overrideWithDynamicTeaser" type="CT_overrideWithDynamicTeaser"/>
      <xs:element name="overrideWithJuniperLog" type="CT_overrideWithJuniperLog"/>
      <xs:element name="overrideWithDynamicTeaserMetric" type="CT_overrideWithDynamicTeaserMetric"/>
    </xs:choice>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="CT_overrideWithRankLog">
  <xs:attribute name="summaryFieldName" type="xs:string" use="required"/>
</xs:complexType>

<xs:complexType name="CT_overrideWithDynamicTeaser">
  <xs:attribute name="summaryFieldName" type="xs:string" use="required"/>
  <xs:attribute name="sourceSummaryFieldName" type="xs:string" use="required"/>
</xs:complexType>

<xs:complexType name="CT_overrideWithJuniperLog">
  <xs:attribute name="summaryFieldName" type="xs:string" use="required"/>
  <xs:attribute name="sourceSummaryFieldName" type="xs:string" use="required"/>
</xs:complexType>

<xs:complexType name="CT_overrideWithDynamicTeaserMetric">
  <xs:attribute name="summaryFieldName" type="xs:string" use="required"/>
  <xs:attribute name="sourceSummaryFieldName" type="xs:string" use="required"/>
</xs:complexType>

<!-- *********************** Simple types *********************** -->

<xs:simpleType name="ST_catalogType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="integer"/>
    <xs:enumeration value="text"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="ST_contextType">
  <xs:restriction base="xs:token">
    <xs:enumeration value="external"/>
    <xs:enumeration value="simple"/>
    <xs:enumeration value="normal"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="ST_SubstringRange">
  <xs:restriction base="xs:integer">
    <xs:minInclusive value="0"/>
    <xs:maxInclusive value="63"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="ST_dummy">
  <xs:restriction base="xs:string">
    <xs:enumeration value=""/>
  </xs:restriction>
</xs:simpleType>
<xs:enumeration value="dummy"/>
</xs:restriction>
</xs:simpleType>

<xs:simpleType name="ST_dummyfield">
  <xs:restriction base="xs:string">
    <xs:enumeration value="dummyfield"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="ST_alwaysZero">
  <xs:restriction base="xs:string">
    <xs:enumeration value="0"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="ST_tuneFactor">
  <xs:restriction base="xs:string">
    <xs:enumeration value="1.00"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="ST_always32">
  <xs:restriction base="xs:string">
    <xs:enumeration value="32"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="ST_yesno">
  <xs:restriction base="xs:string">
    <xs:enumeration value="yes"/>
    <xs:enumeration value="no"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="ST_onoff">
  <xs:restriction base="xs:string">
    <xs:enumeration value="on"/>
    <xs:enumeration value="off"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="ST_alwaysOff">
  <xs:restriction base="xs:string">
    <xs:enumeration value="off"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="ST_direction">
  <xs:restriction base="xs:string">
    <xs:enumeration value="forward"/>
    <xs:enumeration value="backward"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="ST_freshnessBoostDateTimeResolution">
  <xs:restriction base="xs:string">
    <xs:enumeration value="second"/>
    <xs:enumeration value="minute"/>
  </xs:restriction>
</xs:simpleType>
<xs:enumeration value="hour"/>
<xs:enumeration value="day"/>
<xs:enumeration value="year"/>
</xs:restriction>
</xs:simpleType>

<xs:simpleType name="ST_attributeTypes">
<xs:restriction base="xs:token">
<xs:enumeration value="string"/>
<xs:enumeration value="int64"/>
</xs:restriction>
</xs:simpleType>

<xs:simpleType name="ST_summaryFieldTypes">
<xs:restriction base="xs:token">
<xs:enumeration value="string"/>
<xs:enumeration value="longstring"/>
<xs:enumeration value="data"/>
</xs:restriction>
</xs:simpleType>

<xs:simpleType name="ST_summaryClassTypes">
<xs:restriction base="xs:token">
<xs:enumeration value="in"/>
<xs:enumeration value="out"/>
</xs:restriction>
</xs:simpleType>

<xs:simpleType name="ST_alwaysInteger">
<xs:restriction base="xs:token">
<xs:enumeration value="integer"/>
</xs:restriction>
</xs:simpleType>

</xs:schema>

5.5  fixml_mappings.xsd

<?xml version="1.0" encoding="utf-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
<!-- ******************** Global elements ******************** -->
<xs:element name="mappings" type="CT_mappings"/>

<!-- ******************** Complex types ******************** -->
<xs:complexType name="CT_mappings">
<xs:sequence>
<xs:element minOccurs="1" name="map" type="CT_map" maxOccurs="unbounded"/>
</xs:sequence>
<xs:attribute name="sclass" type="xs:string" use="required"/>
</xs:complexType>

<xs:complexType name="CT_map">
<xs:sequence minOccurs="0" maxOccurs="1">
<xs:element name="ignore-value" type="CT_ignore-value"/>
</xs:sequence>
</xs:complexType>
</xs:schema>
<xs:attribute name="type" type="ST_type" use="required"/>
<xs:attribute name="src" type="xs:string" use="required"/>
<xs:attribute name="dst" type="xs:string" use="required"/>
<xs:attribute name="dstcatalog" type="xs:string" use="optional"/>
<xs:attribute name="maxsize" type="xs:int" use="optional" default="64"/>
<xs:attribute name="keepbreaks" type="ST_yesno" use="optional"/>
<xs:attribute name="phrasebreak" type="ST_yesno" use="optional"/>
<xs:attribute name="fieldseparationlength" type="xs:int" use="optional"/>
<xs:attribute name="phraseseparator" type="xs:string" use="optional"/>
<xs:attribute name="multi" type="ST_yesno" use="optional"/>
<xs:attribute name="defaultvalue" type="xs:string" use="optional"/>
<xs:attribute name="separator" type="xs:string" use="optional"/>
</xs:complexType>

<xs:complexType name="CT_ignore-value">
  <xs:attribute name="value" type="xs:string" use="required"/>
</xs:complexType>

<!-- **************************** Simple types ****************--- -->
<xs:simpleType name="ST_yesno">
  <xs:restriction base="xs:string">
    <xs:enumeration value="yes"/>
    <xs:enumeration value="no"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="ST_type">
  <xs:restriction base="xs:string">
    <xs:enumeration value="context"/>
    <xs:enumeration value="rfield"/>
    <xs:enumeration value="sfield"/>
    <xs:enumeration value="attributevector"/>
  </xs:restriction>
</xs:simpleType>

5.6 fieldProperties.xsd

<?xml version="1.0" encoding="utf-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <!-- **************************** Global elements ****************--- -->
  <xs:element name="field-properties" type="CT_field-properties"/>

  <!-- **************************** Complex types ****************--- -->
  <xs:complexType name="CT_field-properties">
    <xs:sequence>
      <xs:element name="field" minOccurs="1" maxOccurs="unbounded" type="CT_field"/>
    </xs:sequence>
    <xs:attribute name="default-index" type="xs:string" use="required"/>
  </xs:complexType>

  <xs:complexType name="CT_field">
    <xs:sequence>
      <!-- Field properties here -->
    </xs:sequence>
  </xs:complexType>
</xs:schema>
<xs:element name="language-tokenization" minOccurs="0" maxOccurs="1" type="CT_language-tokenization"/>
<xs:element name="substring-tokenization" minOccurs="0" maxOccurs="1" type="CT_substring-tokenization"/>
<xs:element name="generic-tokenization" minOccurs="0" maxOccurs="1" type="CT_generic-tokenization"/>
<xs:element name="result" type="CT_result"/>
</xs:sequence>
<xs:attribute name="alias" type="xs:string" use="required"/>
<xs:attribute name="kind" type="ST_fieldKind" use="required"/>
<xs:attribute name="indexed" type="ST_yesno" use="required"/>
<xs:attribute name="type" type="ST_fieldType" use="required"/>
<xs:attribute name="decimal-precision" type="xs:int" use="optional"/>
<xs:attribute name="boundary" type="ST_yesno" use="required"/>
<xs:attribute name="wildcard" type="ST_wildcardAtt" use="required"/>
<xs:attribute name="defines-freshness" type="ST_yes" use="optional"/>
</xs:complexType>

<xs:complexType name="CT_language-tokenization">
<xs:attribute name="lemmatization" type="ST_yesno" use="required"/>
</xs:complexType>

<xs:complexType name="CT_substring-tokenization">
<xs:attribute name="N" type="xs:int" use="required"/>
</xs:complexType>

<xs:complexType name="CT_generic-tokenization">
<xs:attribute name="separator" type="xs:string" use="required"/>
</xs:complexType>

<xs:complexType name="CT_result">
<xs:attribute name="type" use="required" type="ST_resulttype"/>
<xs:attribute name="max-size" type="xs:int" use="optional"/>
</xs:complexType>

<!-- ******************** Simple types ******************** -->

<xs:simpleType name="ST_resulttype">
<xs:restriction base="xs:string">
<xs:enumeration value="no"/>
<xs:enumeration value="static"/>
<xs:enumeration value="dynamic"/>
</xs:restriction>
</xs:simpleType>

<xs:simpleType name="ST_yes">
<xs:restriction base="xs:string">
<xs:enumeration value="yes"/>
</xs:restriction>
</xs:simpleType>

<xs:simpleType name="ST_yesno">
<xs:restriction base="xs:string">
<xs:enumeration value="yes"/>
<xs:enumeration value="no"/>
</xs:restriction>
</xs:simpleType>

<xs:simpleType name="ST_fieldKind">
<xs:restriction base="xs:string">
</xs:restriction>
</xs:simpleType>
5.7 rankspace.xsd

```xml
<?xml version="1.0" encoding="utf-8" ?>
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <!-- ******************************* Global elements *********************** -->
  <xs:element name="rankspace" type="CT_rankspace"/>

  <!-- ************************ Complex types ******************************* -->
  <xs:complexType name="CT_rankspace">
    <xs:sequence>
      <xs:element maxOccurs="unbounded" name="ranking" type="CT_ranking"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="CT_ranking">
    <xs:attribute name="name" type="xs:string" use="required"/>
    <xs:attribute name="description" type="ST_description" use="required"/>
    <xs:attribute name="descendingIndex" type="ST_alwaysZero" use="required"/>
  </xs:complexType>

  <!-- ******************************* Simple types ******************** -->
  <xs:simpleType name="ST_description">
    <xs:restriction base="xs:string">
      <xs:enumeration value="BLISS generated"/>
    </xs:restriction>
  </xs:simpleType>
</xs:schema>
```
<xs:simpleType name="ST_alwaysZero">
  <xs:restriction base="xs:string">
    <xs:enumeration value="0"/>
  </xs:restriction>
</xs:simpleType>

5.8 resultspace.xsd

<?xml version="1.0" encoding="utf-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <!-- *********************** Global elements *********************** -->
  <xs:element name="resultspace" type="CT_resultspace"/>
  <!-- *********************** Complex types *********************** -->
  <xs:complexType name="CT_resultspace">
    <xs:sequence>
      <xs:element name="result-view" type="CT_result-view"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="CT_result-view">
    <xs:sequence>
      <xs:element minOccurs="unbounded" name="field" type="CT_field"/>
    </xs:sequence>
    <xs:attribute name="index" type="ST_index"/>
    <xs:attribute name="name" type="ST_name" use="required"/>
  </xs:complexType>
  <xs:complexType name="CT_field">
    <xs:attribute name="type" type="ST_type" use="required"/>
    <xs:attribute name="name" type="xs:string" use="required"/>
  </xs:complexType>
  <!-- *********************** Simple types *********************** -->
  <xs:simpleType name="ST_index">
    <xs:restriction base="xs:string">
      <xs:enumeration value="0"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="ST_name">
    <xs:restriction base="xs:string">
      <xs:enumeration value="DATASEARCHDEFAULT"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="ST_type">
    <xs:restriction base="xs:string">
      <xs:enumeration value="string"/>
      <xs:enumeration value="integer"/>
    </xs:restriction>
  </xs:simpleType>
</xs:schema>
5.9 summaryclasses.xsd

<?xml version="1.0" encoding="utf-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <!-- *********************** Global elements *********************** -->
  <xs:element name="summary-input-classes" type="CT_summary-input-classes"/>
  <!-- *********************** Complex types *********************** -->
  <xs:complexType name="CT_summary-input-classes">
    <xs:sequence>
      <xs:element name="summaryClass" type="CT_summaryClass" />
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="CT_summaryClass">
    <xs:sequence>
      <xs:element maxOccurs="unbounded" name="summaryField" type="CT_summaryField"/>
    </xs:sequence>
    <xs:attribute name="name" type="ST_className" use="required"/>
    <xs:attribute name="type" type="ST_classType" use="required"/>
  </xs:complexType>
  <xs:complexType name="CT_summaryField">
    <xs:attribute name="name" type="xs:string" use="required"/>
    <xs:attribute name="type" type="ST_summaryType" use="required"/>
    <xs:attribute name="compression" type="ST_compression" use="optional"/>
  </xs:complexType>
  <!-- *********************** Simple types *********************** -->
  <xs:simpleType name="ST_classType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="in"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="ST_className">
    <xs:restriction base="xs:string">
      <xs:enumeration value="content"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="ST_summaryType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="string"/>
      <xs:enumeration value="longstring"/>
      <xs:enumeration value="data"/>
    </xs:restriction>
  </xs:simpleType>
</xs:schema>
<xs:simpleType name="ST_compression">
    <xs:restriction base="xs:string">
        <xs:enumeration value="on"/>
        <xs:enumeration value="off"/>
    </xs:restriction>
</xs:simpleType>

5.10 ManagedPropertyBoosts.xsd

<?xml version="1.0" encoding="utf-8"?>
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
    xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <!-- Global elements -->
    <xs:element name="field-boosts" type="CT_FieldBoosts"/>
    <!-- Complex types -->
    <xs:complexType name="CT_FieldBoosts">
        <xs:sequence>
            <xs:element name="rank-profile" type="CT_RankProfile" maxOccurs="unbounded" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="CT_RankProfile">
        <xs:sequence>
            <xs:element name="boost" type="CT_BoostGroup" maxOccurs="unbounded" minOccurs="0"/>
        </xs:sequence>
        <xs:attribute name="name" type="xs:string" use="required"/>
        <xs:attribute name="index" type="ST_RankProfileIndex" use="required"/>
    </xs:complexType>
    <xs:complexType name="CT_BoostGroup">
        <xs:sequence>
            <xs:element name="field-boost" type="CT_FieldBoost" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="value" type="xs:int" use="required"/>
    </xs:complexType>
    <xs:complexType name="CT_FieldBoost">
        <xs:attribute name="name" type="xs:string" use="required"/>
        <xs:attribute name="keyword" type="xs:string" use="required"/>
    </xs:complexType>
    <!-- Simple types -->
    <xs:simpleType name="ST_RankProfileIndex">
        <xs:restriction base="xs:unsignedInt">
            <xs:minInclusive value="0"/>
            <xs:maxInclusive value="2147483647"/>
        </xs:restriction>
    </xs:simpleType>
6 Appendix B: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include released service packs:

- Microsoft® FAST™ Search Server 2010

Exceptions, if any, are noted below. If a service pack or Quick Fix Engineering (QFE) number appears with the product version, behavior changed in that service pack or QFE. The new behavior also applies to subsequent service packs of the product unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

Unless otherwise specified, any statement of optional behavior in this specification that is prescribed using the terms SHOULD or SHOULD NOT implies product behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term MAY implies that the product does not follow the prescription.
7 Change Tracking

No table of changes is available. The document is either new or has had no changes since its last release.
8 Index

A

ABNF grammar
    index.cf 83
    rank.cf 93
    summary.cf 114
Abstract data model
    index schema 13
    index schema - classes (section 1.3.2.1 14, section 1.3.2.2 15, section 1.3.2.3 15, section
    1.3.2.4 16, section 1.3.2.5 18, section 1.3.2.6 18)
Applicability 19
Attribute vector configuration 86

B

Boost Table Files example 141
Boost table files structure 106
boost table files structure – global term frequency
    boost table file 107
boost table files structure – occurrence boost table files 106
boost table files structure – proximity boost table files 107

C

Change tracking 168
Classes – abstract data model
    FullTextIndex 15
    FullTextIndexRank 18
    ImportanceLevel 18
    ManagedProperty 14
    RankProfile 16
    RefinerConfiguration 15
Common concepts and type definitions
    data type definition and maps 22
    document summary types 24
    index field prefix naming conventions 23
    internal properties 24
    managed properties 24
Common concepts and type definitions structure 22
Common data types and fields (section 2 20, section 2 20)
Complex types – configuration.attributes.xml structure
    CT_datetimeNav 36
    CT_discretize 40
    CT_display 39
    CT_equalfrequency 41
    CT_equalwidth 42
    CT_filter 38
    CT_firstLast 39
    CT_fixedpoint 37
    CT_middle 40
    CT_navigator 34
    CT_navigators 34
    CT_numericNav 37
    CT_rangedivision 41
    CT_score 42
    CT_sort 38
    CT_stringNav 37
Complex types – FieldProperties.xml structure
    CT_field 100
    CT_field-properties 99
    CT_generic-tokenization 101
    CT_language-tokenization 102
    CT_result 102
    CT_substring-tokenization 101
Complex types – fieldspec.xml structure
    CT_field 32
    CT_fieldlist 32
Complex types – fixml_mappings.xml structure
    CT_ignore-value 91
    CT_map 88
    CT_mappings 87
Complex types – indexConfig.xml structure
    CT_alias 55
    CT_attributeVector 67
    CT_attributeVectorList 67
    CT_boostValue 62
    CT_catalog 53
    CT_catalogList 53
    CT_catalogRankList 60
    CT_context 54
    CT_contextBoost 66
    CT_contextBoostList 66
    CT_contextRef 55
    CT_defaultIndex 56
    CT_divTableBoost 64
    CT_dynamicRankParameters 59
    CT_extNumOccBoostOnlyCatalog 61
    CT_FastIndexingConfig 52
    CT_freshnessBoostCoefficient 65
    CT_freshnessBoostDateTimeResolution 65
    CT_freshnessBoostFileRef 65
    CT_freshnessBoostParameters 64
    CT_index 54
    CT_occBoost 63
    CT_overrideWithDynamicTeaser 71
    CT_overrideWithDynamicTeaserMetric 71
    CT_overrideWithJuniperLog 72
    CT_overrideWithRankLog 72
    CT_proximityBoost 63
    CT_qualityComponent 58
    CT_qualityComponentList 58
    CT_rankedCatalog 61
    CT_rankProfile 57
    CT_rankProfileList 56
    CT_staticRankClassList 56
    CT_staticRankParameters 57
    CT_summaryClass 69
    CT_summaryClassList 68
    CT_summaryField 69
    CT_summaryFieldOverrideList 70
Complex types – maptransform.xml structure
    CT_datatype 26

[MS-FSSCFG] — v20120630
Search Configuration File Format

Copyright © 2012 Microsoft Corporation.

Release: July 16, 2012
Configuration parameter details – index.cf 84
Configuration parameter details – index.cf structure
attribute vector configuration 86
catalog configuration 84
default index configuration 85
drilling configuration 86
index alias configuration 86
Configuration parameter details – resultfield.map 33
Configuration parameter reference – rank.cf 96
Configuration parameter reference – rank.cf structure
catalog configuration-level parameters 97
profile-level parameters 96
Configuration parameter reference – summary.cf 115
Configuration parameters derived from index schema – fsearchaddon 50
configuration.attributes.xml example 127
configuration.attributes.xml global attributes 34
configuration.attributes.xml structure 33
configuration.attributes.xml structure complex types
CT_datatype-definitions 26
CT_field 29
CT_number-transformations 28
CT_transform-specification 25
Complex types – rankspace.xml structure
CT_rankspace 108
Complex types – resultspace.xml structure
CT_field 111
CT_resultspace 110
CT_result-view 111
Complex types – summaryclasses.xml structure
CT_summaryClass 117
CT_summaryField 117
CT_summary-input-classes 116
Configuration parameter details – fdpdispatchaddon 47
Configuration parameter details – index.cf 84
Configuration parameter details – index.cf structure
attribute vector configuration 86
catalog configuration 84
default index configuration 85
drilling configuration 86
index alias configuration 86
Context catalog configuration 84
Context catalog structure – indexConfig.xml 79
Context catalog structure – indexConfig.xml structure
numeric catalogs 80
ranked context catalogs 80
synthetic context catalogs 79
Context catalog-level parameters 97
CT_alias type 55
CT_attributeVector type 67
CT_attributeVectorList type 67
CT_boostValue type 62
CT_catalog type 53
CT_catalogList type 53
CT_catalogRankList type 60
CT_context type 54
CT_contextBoost type 66
CT_contextBoostList type 66
CT_contextRef type 55
CT_datatype type 26
CT_datatype-definitions type 26
CT_datetimeNav type 36
CT_defaultIndex type 56
CT_discretize type 40
CT_display type 39
CT_divTableBoost type 64
CT_dynamicRankParameters type 59
CT_equalfrequency type 41
CT_equalwidth type 42
CT_extNumOccBoostOnlyCatalog type 61
CT_freshnessBoostParameters type 64
CT_freshnessBoostCoefficient type 65
CT_freshnessBoostDateTimeResolution type 65
CT_freshnessBoostFileRef type 65
CT_generic-tokenization type 101
CT_ignore-value type 91
CT_index type 54
CT_language-tokenization type 102
CT_map type 88
CT_mappings type 87
CT_middle type 40
CT_navigator type 34
CT_navigators type 34
CT_number-transformations type 28
CT_numericNav type 37
CT_occBoost type 63
CT_overrideWithDynamicTeaser type 71
CT_overrideWithDynamicTeaserMetric type 71
CT_overrideWithJuniperLog type 72
CT_overrideWithRankLog type 72
CT_proximityBoost type 63
CT_qualityComponent type 58
CT_qualityComponentList type 58
CT_rangedivision type 41
CTRankedCatalog type 61
CT_ranking type 109
CT_rankProfile type 57
CT_rankProfileList type 56
CT_resultspace type 108
CT_result type 102
CT_resultspace type 110
CT_result-view type 111
CT_score type 42
CT_score type 42
CT_stringNav type 37
CT_substring-tokenization type 101
CT_summaryClass type (section 2.8.3.32 69, section 2.20.3.2 117)
CT_summaryClassList type 68
CT_summaryField type (section 2.8.3.33 69, section 2.20.3.3 117)
CT_summaryFieldOverridelist type 70
CT_summary-input-classes type 116
CT_transform-specification type 25

D

Data type definition and maps 22
Data types and fields - common (section 2 20, section 2 20)
Default index configuration 85
Details
ABNF grammar – index.cf 83
ABNF grammar – rank.cf 93
ABNF grammar – summary.cf 114
boost table files structure 106
common concepts and type definitions structure 22
common data types and fields (section 2 20, section 2 20)
configuration parameter – fdispatch.addon 47
configuration parameter – index.cf 84
configuration parameter – resultfield.map 33
configuration parameter reference – rank.cf 96
configuration parameter reference – summary.cf 115
configuration parameters derived from index
schema – fsearch.addon 50
configuration.attributes.xml structure 33
context catalog structure – indexConfig.xml 79
data type definition and maps 22
document summary types 24
fixml_mappings.xml structure 31
fieldspec.xml structure 31
file content – fdispatch.addon 47
file content – resultfield.map 33
fsearch.addon structure 48
fixml_mappings.xml structure 86
global attributes – configuration.attributes.xml 34
global attributes – FieldProperties.xml 99
global attributes – fieldspec.xml 31
global attributes – fixml_mappings.xml 87
global attributes – indexConfig.xml 52
global attributes – maptransform.xml 25
global attributes – rankspace.xml 108
global attributes – resultspace.xml 110
global attributes – summaryclasses.xml 116
global term frequency boost table file – boost
table files structure 107
index field prefix naming conventions 23
index.cf structure 82
indexConfig.xml structure 51
internal properties 24
managed properties 24
maptransform.xml structure 25
occurrence boost table files – boost table files
structure 106
proximity boost table files – boost table files
structure 107
rank.cf structure 92
rankspace.xml structure 108
resultfield.map structure 33
resultspace.xml structure 110
search_preload structure 113
sources.xml structure 113
static hit highlighted summary parameters –
fsearch.addon 48
summary classes – summary.cf 115
summary.cf structure 114
summary.map structure 115
summaryclasses.xml structure 116
XML content – sources.xml structure 113
Document summary types 24
Drilling configuration 86

E

Examples 126
Boost Table Files 141
configuration.attributes.xml 127
FieldProperties.xml 140
fieldspec.xml 127
fixml_mappings.xml 137
fsearch.addon 129
index.cf 134
indexConfig.xml 130
maptransform.xml 126
overview 126
rank.cf 139
rankspace.xml 142
resultspace.xml 143
summary.cf 143
summaryclasses.xml 144

[MS-FSSCFG] – v20120630
Search Configuration File Format

Copyright © 2012 Microsoft Corporation.

Release: July 16, 2012
Product behavior 167
Properties
  internal 24
  managed 24
Proximity boost table files – boost table files structure 107

R

Rank profile-level parameters 96
rank.cf ABNF grammar 93
rank.cf configuration parameter reference 96
rank.cf example 139
rank.cf structure 92
rank.cf structure configuration parameter reference context catalog-level parameters 97
rank profile-level parameters 96
Ranked context catalogs 80
RankProfile class 16
rankspace element 108
rankspace.xml example 142
rankspace.xml global attributes 108
rankspace.xml structure 108
rankspace.xml structure complex types CT_ranking 109
CT_rankspace 108
rankspace.xml structure global elements rankspace 108
rankspace.xml structure simple types ST_alwaysZero 109
ST_description 109
References 11
  informative 12
  normative 11
RefinerConfiguration class 15
Relationship to protocols and other structures 18
resultfield.map configuration parameter details 33
resultfield.map file content 33
resultfield.map structure 33
resultspace element 110
resultspace.xml example 143
resultspace.xml global attributes 110
resultspace.xml structure 110
resultspace.xml structure complex types CT_field 111
CT_resultspace 110
CT_result-view 111
resultspace.xml structure global elements resultspace 110
resultspace.xml structure simple types ST_index 112
ST_name 112
ST_type 112

S

Schemas - XML
  configuration.attributes.xsd 148
  fieldProperties.xsd 160
  fieldspec.xsd 147
  fixml_mappings.xsd 159
  indexConfig.xsd 152
  maptransform.xsd 146
  overview 146
  rankspace.xsd 162
  resultspace.xsd 163
  summaryclasses.xsd 164
Search index
  managed properties 24
  search_preload structure 113
  Security - implementer considerations 145
Simple types – configuration.attributes.xml structure
  ST_algorithm 44
  ST_alwaysno 46
  ST_alwaysOne 46
  ST_alwaysZero 46
  ST anchoring 44
  ST by 45
  ST multimode 43
  ST_order 45
  ST_type 43
  ST yesno 46
Simple types – FieldProperties.xml structure
  ST_fieldKind 104
  STFieldType 104
  ST_resulttype 103
  ST_wildcardAtt 105
  ST yes 103
  ST yesno 103
Simple types – fieldspec.xml structure
  ST_sorttype 32
Simple types – fixml_mappings.xml structure
  ST_type 92
  ST yesno 92
Simple types – indexConfig.xml structure
  ST_always32 75
  ST_alwaysInteger 78
  ST_alwaysOff 76
  ST_alwaysZero 74
  ST_attributeTypes 77
  ST_catalogType 73
  ST_contextType 73
  ST_direction 76
  ST dummy 74
  ST dummyfield 74
  ST freshnessBoostDateTimeResolution 77
  ST onoff 76
  ST substringRange 73
  ST summaryClassTypes 78
  ST summaryFieldTypes 78
  ST tuneFactor 75
  ST yesno 75
Simple types – maptransform.xml structure
  ST_decimalplaces 31
  ST expbase 30
  ST exponentbits 30
  ST mantissabits 30
  ST offsetbits 29
  ST signbits 30
  ST toint 31
Simple types – rankspace.xml structure
  ST_alwaysZero 109
Simple types – resultspace.xml structure

- **ST_description**

- **ST_index**

- **ST_name**

- **ST_type**

Simple types – summaryclasses.xml structure

- **ST_className**

- **ST_classType**

- **ST_compression**

- **ST_summaryType**

sources.xml structure

- **sources.xml structure – XML content**

Simple types – summaryclasses.xml structure

- **ST_className**

- **ST_classType**

- **ST_compression**

- **ST_summaryType**

sources.xml structure

- **sources.xml structure – XML content**

Structures

- **boost table files**

- **common concepts and type definitions**

- **configuration.attributes.xml**

- **dispatch.addon**

- **FieldProperties.xml**

- **fieldspec.xml**

- **fixml_mappings.xml**

- **fsearch.addon**

- **index.cf**

- **indexConfig.xml**

- **maptransform.xml**

- **overview (section 2.20, section 2.20)**

- **rank.cf**

- **rankspace.xml**

- **resultfield.map**

- **resultspace.xml**

- **search_preload**

- **sources.xml**

- **summary.cf**

- **summary.map**

- **summaryclasses.xml**

- **Summary classes – summary.cf**

Summary types

- **document summary types**

- **summary.cf ABNF grammar**

- **summary.cf configuration parameter reference**

- **summary.cf example**

- **summary.cf structure**

- **summary.cf summary classes**

- **summary.map structure**

- **summaryclasses.xml example**

- **summaryclasses.xml global attributes**

- **summaryclasses.xml structure**

- **summaryclasses.xml structure complex types**

- **summaryclasses.xml structure global elements**

Summary classes – summary.types

- **summary.classes.xml structure simple types**

- **ST_className**

- **ST_classType**

- **ST_compression**

- **ST_summaryType**

- **summary-input-classes**

Synthetic context catalogs

- **CT_summaryClass**

- **CT_summaryField**

- **CT_summary-input-classes**

Vendor-extendible fields

- **Vendor-extensible fields**

Versioning

- **Versioning**

X
**XML content – source.xml structure** 113
**XML schema** 146
  - `configuration.attributes.xsd` 148
  - `fieldProperties.xsd` 160
  - `fieldspec.xsd` 147
  - `fixml_mappings.xsd` 159
  - `indexConfig.xsd` 152
  - `maptransform.xsd` 146
  - `overview` 146
  - `rankspace.xsd` 162
  - `resultspace.xsd` 163
  - `summaryclasses.xsd` 164