[MS-SEARCH]:

Search Protocol

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. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.
- **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

Date	Revision History	Revision Class	Comments
4/4/2008	0.1	New	Initial Availability
6/27/2008	1.0	Major	Revised and edited the technical content
10/6/2008	1.01	Editorial	Revised and edited the technical content
12/12/2008	1.02	Editorial	Revised and edited the technical content
7/13/2009	1.03	Major	Revised and edited the technical content
8/28/2009	1.04	Editorial	Revised and edited the technical content
11/6/2009	1.05	Editorial	Revised and edited the technical content
2/19/2010	2.0	Major	Updated and revised the technical content
3/31/2010	2.01	Major	Updated and revised the technical content
4/30/2010	2.02	Editorial	Revised and edited the technical content
6/7/2010	2.03	Editorial	Revised and edited the technical content
6/29/2010	2.04	Editorial	Changed language and formatting in the technical content.
7/23/2010	2.05	Minor	Clarified the meaning of the technical content.
9/27/2010	2.05	None	No changes to the meaning, language, or formatting of the technical content.
11/15/2010	2.05	None	No changes to the meaning, language, or formatting of the technical content.
12/17/2010	2.05	None	No changes to the meaning, language, or formatting of the technical content.
3/18/2011	2.5.1	Editorial	Changed language and formatting in the technical content.
6/10/2011	2.5.1	None	No changes to the meaning, language, or formatting of the technical content.
1/20/2012	3.0	Major	Significantly changed the technical content.
4/11/2012	3.0	None	No changes to the meaning, language, or formatting of the technical content.
7/16/2012	3.0	None	No changes to the meaning, language, or formatting of the technical content.
9/12/2012	3.0	None	No changes to the meaning, language, or formatting of the technical content.
10/8/2012	3.1	Minor	Clarified the meaning of the technical content.
2/11/2013	3.1	None	No changes to the meaning, language, or formatting of the technical content.
7/30/2013	3.1	None	No changes to the meaning, language, or formatting of the technical content.
11/18/2013	3.2	Minor	Clarified the meaning of the technical content.

Date	Revision History	Revision Class	Comments
2/10/2014	3.3	Minor	Clarified the meaning of the technical content.
4/30/2014	3.4	Minor	Clarified the meaning of the technical content.
7/31/2014	3.5	Minor	Clarified the meaning of the technical content.
10/30/2014	4.0	Major	Significantly changed the technical content.
3/16/2015	5.0	Major	Significantly changed the technical content.
2/26/2016	6.0	Major	Significantly changed the technical content.

Table of Contents

1 Introduction

This document specifies the Search Protocol that enables clients to make queries against an Enterprise Search service, the protocol server responding with a list of items that are relevant to the search query. This protocol also allows protocol clients to request query suggestions for a given search query.

Sections 1.5, 1.8, 1.9, 2, and 3 of this specification are normative. All other sections and examples in this specification are informative.

1.1 Glossary

This document uses the following terms:

ASCII: The American Standard Code for Information Interchange (ASCII) is an 8-bit character-encoding scheme based on the English alphabet. ASCII codes represent text in computers, communications equipment, and other devices that work with text. ASCII refers to a single 8-bit ASCII character or an array of 8-bit ASCII characters with the high bit of each character set to zero.

Augmented Backus-Naur Form (ABNF): A modified version of Backus-Naur Form (BNF), commonly used by Internet specifications. ABNF notation balances compactness and simplicity with reasonable representational power. ABNF differs from standard BNF in its definitions and uses of naming rules, repetition, alternatives, order-independence, and value ranges. For more information, see [RFC5234].

best bet: A URL that a site collection administrator assigns to a keyword as being relevant for that keyword. See also **visual best bet**.

binary large object (BLOB): A discrete packet of data that is stored in a database and is treated as a sequence of uninterpreted bytes.

bucket: A collection of items that were requested by a search application during a crawl. An item can be a person, a document, or any other type of item that can be crawled.

child element: In an XML document, an element that is subordinate to and is contained by another element, which is referred to as the parent element.

document vector: A set of name/value pairs that stores the most important terms and corresponding relevance weights for an indexed item.

duplicate: A search result that is identified as having identical or near identical content.

duplicate result removal: An operation to compare the similarity of items and remove duplicates from search results.

file extension: The sequence of characters in a file's name between the end of the file's name and the last "." character. Vendors of applications choose such sequences for the applications to uniquely identify files that were created by those applications. This allows file management software to determine which application should be used to open a file.

folder: A file system construct. File systems organize a volume's data by providing a hierarchy of objects, which are referred to as folders or directories, that contain files and can also contain other folders.

globally unique identifier (GUID): A term used interchangeably with universally unique identifier (UUID) in Microsoft protocol technical documents (TDs). Interchanging the usage of these terms does not imply or require a specific algorithm or mechanism to generate the value. Specifically, the use of this term does not imply or require that the algorithms described in

[RFC4122] or [C706] must be used for generating the **GUID**. See also universally unique identifier (UUID).

Graphics Interchange Format (GIF): A compression format that supports device-independent transmission and interchange of bitmapped image data. The format uses a palette of up to 256 distinct colors from the 24-bit RGB color space. It also supports animation and a separate palette of 256 colors for each frame. The color limitation makes the GIF format unsuitable for reproducing color photographs and other images with gradients of color, but it is well-suited for simpler images such as graphics with solid areas of color.

hit highlighted summary: A summary that appears on the search results page for each query result. It displays an excerpt from the item that contains the query text and applies highlight formatting to that query text.

inflectional form: A variant of a root token that has been modified according to the linguistic rules of a given language. For example, inflections of the verb "swim" in English include "swim," "swims," "swimming," and "swam."

item: A unit of content that can be indexed and searched by a search application.

Joint Photographic Experts Group (JPEG): A raster graphics file format for displaying high-resolution color graphics. JPEG graphics apply a user-specified compression scheme that can significantly reduce the file sizes of photo-realistic color graphics. A higher level of compression results in lower quality, whereas a lower level of compression results in higher quality. JPEG-format files have a .jpg or .jpeg file name extension.

language code identifier (LCID): A 32-bit number that identifies the user interface human language dialect or variation that is supported by an application or a client computer.

list: A container within a SharePoint site that stores list items. A list has a customizable schema that is composed of one or more fields.

list item: An individual entry within a SharePoint list. Each list item has a schema that maps to fields in the list that contains the item, depending on the content type of the item.

managed property: A specific property that is part of a metadata schema. It can be exposed for use in search queries that are executed from the user interface.

multivalue property: A property that can contain multiple values of the same type.

noise word: See stop word.

Office SharePoint Server Search service: A farm-wide service that either responds to query requests from front-end web servers or crawls items.

Portable Network Graphics (PNG): A bitmap graphics file format that uses lossless data compression and supports variable transparency of images (alpha channels) and control of image brightness on different computers (gamma correction). PNG-format files have a .png file name extension.

post-query suggestions: An alternative search query that is related to the search query that was executed.

predicate: A statement that is associated with a crawled item and is used to determine whether a document is returned in query results. Its value depends on the state of the full-text index catalog.

pre-query suggestions: A search query that is related to the search query that the user is typing.

property identifier: A unique integer or a 16-bit, numeric identifier that is used to identify a specific attribute (1) or property.

query: A formalized instruction to a data source to either extract data or perform a specified action. A query can be in the form of a query expression, a method-based query, or a combination of the two. The data source can be in different forms, such as a relational database, XML document, or in-memory object. See also **search query**.

query context: A component of a promotion that specifies the contexts in which a promotion is applied. Examples include the site where the query originates and a user's role or location.

query refinement: A process that is used to drill into query results by using aggregated statistical data, such as the distribution of managed property values in query results.

query result: A result that is returned for a query. It contains the title and URL of the item, and can also contain other managed properties and a hit-highlighted summary.

query text: The textual, string portion of a query.

rank: An integer that represents the relevance of a specific item for a search query. It can be a combination of static rank and dynamic rank. See also static rank and dynamic rank.

ranking model: In a search query, a set of weights and numerical parameters that are used to compute a ranking score for each item. All items share the same ranking model for a specific set of search results. See also **rank**.

refinement token: A Base-64 encoded string that represents a single refinement modifier that can be used to refine a search query. The string includes the name of the refiner, refinement name, and refinement value.

refiner: A configuration that is used for **query refinement** and is associated with one managed property.

result provider: A component or application that serves a query to a search provider and translates the resulting data into a result set.

retrievable property: A managed property that is stored in a metadata index.

root element: The top-level element in an XML document. It contains all other elements and is not contained by any other element, as described in [XML].

search application: A unique group of search settings that is associated, one-to-one, with a shared service provider.

search query: A complete set of conditions that are used to generate search results, including query text, sort order, and ranking parameters.

search scope: A list of attributes that define a collection of items.

search setting context: An administrative setting that is used to specify when a search setting for a keyword is applied to a search query, based on the query context.

securable object: An object that can have unique security permissions associated with it.

shallow refinement: A type of query refinement that is based on the aggregation of managed property statistics for only some results of a search query. The number of refined results varies according to implementation. See also deep refinement.

site: A group of related pages and data within a SharePoint site collection. The structure and content of a site is based on a site definition. Also referred to as SharePoint site and web site.

site collection administrator: A user who has administrative permissions for a site collection.

SOAP: A lightweight protocol for exchanging structured information in a decentralized, distributed environment. **SOAP** uses **XML** technologies to define an extensible messaging framework, which provides a message construct that can be exchanged over a variety of underlying protocols. The framework has been designed to be independent of any particular programming model and other implementation-specific semantics. SOAP 1.2 supersedes SOAP 1.1. See [SOAP1.2-1/2003].

SOAP action: The HTTP request header field used to indicate the intent of the **SOAP** request, using a URI value. See [SOAP1.1] section 6.1.1 for more information.

SOAP body: A container for the payload data being delivered by a **SOAP message** to its recipient. See [SOAP1.2-1/2007] section 5.3 for more information.

SOAP fault: A container for error and status information within a **SOAP message**. See [SOAP1.2-1/2007] section 5.4 for more information.

SOAP message: An **XML** document consisting of a mandatory SOAP envelope, an optional SOAP header, and a mandatory **SOAP body**. See [SOAP1.2-1/2007] section 5 for more information.

SOAP Message: The data encapsulated in a SOAP envelope that flows back and forth between a protocol client and a web service, as described in [SOAP1.1].

sort order: A set of rules in a search query that defines the ordering of rows in the search result. Each rule consists of a managed property, such as modified date or size, and a direction for order, such as ascending or descending. Multiple rules are applied sequentially.

subsite: A complete website that is stored in a named subdirectory of another website. The parent website can be the top-level site of a site collection or another subsite. Also referred to as subweb.

thesaurus: A file that contains a list of synonym sets. Each synonym set contains two or more terms that have the same meaning. When a search query is processed, the search is expanded to include the synonyms if the query text matches a term in the thesaurus. For example, with a synonym set of "cat, feline," a search for cat will retrieve items that contain either "cat" or "feline."

token: A word in an item or a search query that translates into a meaningful word or number in written text. A token is the smallest textual unit that can be matched in a search query. Examples include "cat", "AB14", or "42".

Unicode: A character encoding standard developed by the Unicode Consortium that represents almost all of the written languages of the world. The **Unicode** standard [UNICODE5.0.0/2007] provides three forms (UTF-8, UTF-16, and UTF-32) and seven schemes (UTF-8, UTF-16, UTF-16 BE, UTF-16 LE, UTF-32, UTF-32 LE, and UTF-32 BE).

Unicode code point: Any value in the Unicode codespace, which is a range of integers from "0" to "10FFFF16". Each code point is a unique positive integer that maps to a specific character.

Uniform Resource Locator (URL): A string of characters in a standardized format that identifies a document or resource on the World Wide Web. The format is as specified in [RFC1738].

visual best bet: A URL that specifies the address of an image and is assigned to a keyword by a site collection administrator as being relevant for that keyword. See also **best bet**.

WSDL operation: A single action or function of a web service. The execution of a WSDL operation typically requires the exchange of messages between the service requestor and the service provider.

XML: The Extensible Markup Language, as described in [XML1.0].

XML namespace: A collection of names that is used to identify elements, types, and attributes in XML documents identified in a URI reference [RFC3986]. A combination of XML namespace and local name allows XML documents to use elements, types, and attributes that have the same names but come from different sources. For more information, see [XMLNS-2ED].

XML namespace prefix: An abbreviated form of an **XML namespace**, as described in [XML].

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

1.2 References

Links to a document in the Microsoft Open Specifications library point to the correct section in the most recently published version of the referenced document. However, because individual documents in the library are not updated at the same time, the section numbers in the documents may not match. You can confirm the correct section numbering by checking the Errata.

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.

[IEEE754] IEEE, "IEEE Standard for Binary Floating-Point Arithmetic", IEEE 754-1985, October 1985, http://ieeexplore.ieee.org/servlet/opac?punumber=2355

[MS-DSDIFFGRAM] Microsoft Corporation, "SharePoint Web Services: DataSet DiffGram Structure".

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, http://www.rfc-editor.org/rfc/rfc2119.txt

[RFC2616] Fielding, R., Gettys, J., Mogul, J., et al., "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999, http://www.rfc-editor.org/rfc/rfc2616.txt

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, May 2000, http://www.rfc-editor.org/rfc/rfc2818.txt

[RFC3339] Klyne, G. and Newman, C., "Date and Time on the Internet: Timestamps", RFC 3339, July 2002, http://www.ietf.org/rfc/rfc3339.txt

[RFC4646] Phillips, A., and Davis, M., Eds., "Tags for Identifying Languages", BCP 47, RFC 4646, September 2006, http://www.rfc-editor.org/rfc/rfc4646.txt

[RFC5234] Crocker, D., Ed., and Overell, P., "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008, http://www.rfc-editor.org/rfc/rfc5234.txt

[SOAP1.1] Box, D., Ehnebuske, D., Kakivaya, G., et al., "Simple Object Access Protocol (SOAP) 1.1", May 2000, http://www.w3.org/TR/2000/NOTE-SOAP-20000508/

[SOAP1.2/1] Gudgin, M., Hadley, M., Mendelsohn, N., Moreau, J., and Nielsen, H.F., "SOAP Version 1.2 Part 1: Messaging Framework", W3C Recommendation, June 2003, http://www.w3.org/TR/2003/REC-soap12-part1-20030624

[SOAP1.2/2] Gudgin, M., Hadley, M., Mendelsohn, N., Moreau, J., and Nielsen, H.F., "SOAP Version 1.2 Part 2: Adjuncts", W3C Recommendation, June 2003, http://www.w3.org/TR/2003/REC-soap12-part2-20030624

[UNICODE3.1] The Unicode Consortium, "Unicode Data 3.1.0", February 2001, http://www.unicode.org/Public/3.1-Update/UnicodeData-3.1.0.txt

[UNICODE] The Unicode Consortium, "The Unicode Consortium Home Page", 2006, http://www.unicode.org/

[WSDL] Christensen, E., Curbera, F., Meredith, G., and Weerawarana, S., "Web Services Description Language (WSDL) 1.1", W3C Note, March 2001, http://www.w3.org/TR/2001/NOTE-wsdl-20010315

[XML10] World Wide Web Consortium, "Extensible Markup Language (XML) 1.0 (Third Edition)", February 2004, http://www.w3.org/TR/2004/REC-xml-20040204/

[XMLNS] Bray, T., Hollander, D., Layman, A., et al., Eds., "Namespaces in XML 1.0 (Third Edition)", W3C Recommendation, December 2009, http://www.w3.org/TR/2009/REC-xml-names-20091208/

[XMLSCHEMA1] Thompson, H., Beech, D., Maloney, M., and Mendelsohn, N., Eds., "XML Schema Part 1: Structures", W3C Recommendation, May 2001, http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/

[XMLSCHEMA2] Biron, P.V., Ed. and Malhotra, A., Ed., "XML Schema Part 2: Datatypes", W3C Recommendation, May 2001, http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/

1.2.2 Informative References

[MS-FQL2] Microsoft Corporation, "Fast Query Language Version 2 Protocol".

[MS-QSSWS] Microsoft Corporation, "Search Query Shared Services Protocol".

[MSDN-RANKMETHOD] Microsoft Corporation, "RANKMETHOD Term in Enterprise Search SQL Syntax", http://msdn.microsoft.com/en-us/library/ms550247.aspx

1.3 Overview

The Search Protocol enables clients to make **search queries** against an Enterprise Search service. The protocol client sends a search query to the protocol server, and the protocol server responds with a list of **items** that are relevant to the search query. This protocol also allows clients to request query suggestions for a given search query, the protocol server responding with a list of **pre-query suggestions** or **post-query suggestions** as requested by the client.

1.4 Relationship to Other Protocols

This protocol uses the **SOAP** message protocol for formatting request and response messages, as described in [SOAP1.1], [SOAP1.2/1] and [SOAP1.2/2]. It transmits those messages by using HTTP, as described in [RFC2616], or Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS), as described in [RFC2818].

The following diagram shows the underlying messaging and transport stack used by the protocol:

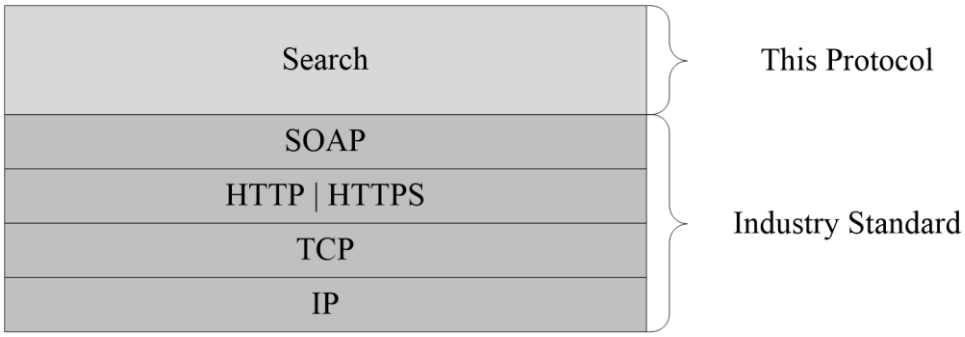

Figure 1: This protocol in relation to other protocols

1.5 Prerequisites/Preconditions

This protocol operates against a **site** that is identified by a **URL** that is known by protocol clients. The protocol client uses the protocol server endpoint as specified in section 2.1.

The protocol client needs sufficient privileges to access the site.

This protocol assumes that authentication has been performed by the underlying protocols.

1.6 Applicability Statement

This protocol was designed for returning results sets containing less than or equal to 10,000 rows.

1.7 Versioning and Capability Negotiation

This protocol uses multiple transports with SOAP as specified in section 2.1.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

In the following sections, the schema definition might differ from the processing rules imposed by the protocol. The WSDL in this specification matches the WSDL that shipped with the product and provides a base description of the schema. The text that introduces the WSDL might specify differences that reflect actual Microsoft product 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**, **not null**, and **present**.

2.1 Transport

Protocol servers MUST support SOAP over HTTP. Protocol servers SHOULD additionally support SOAP over HTTPS for securing communication with protocol clients, as specified in [RFC2818].

Protocol messages MUST be formatted as specified either in [SOAP1.1] section 4 or in [SOAP1.2/1] section 5. Protocol server faults MUST be returned either using HTTP Status Codes, as specified in [RFC2616] section 10 or using **SOAP faults**, as specified either in [SOAP1.1] section 4.4 or in [SOAP1.2/1] section 5.4.

The protocol server endpoint is formed by appending the suffix "/_vti_bin/search.asmx" to the URL of the site. For example, if the URL of the site (2) were http://www.contoso.com/Repository, the protocol server endpoint would be http://www.contoso.com/Repository/_vti_bin/search.asmx.<1>

2.2 Common Message Syntax

This section contains common definitions that are used by this protocol. The syntax of the definitions uses XML schema, as specified in [XMLSCHEMA1] and [XMLSCHEMA2], and WSDL, as specified in [WSDL].

2.2.1 Namespaces

This specification defines and references various **XML namespaces** using the mechanisms specified in [XMLNS]. Although this specification associates a specific **XML namespace prefix** for each XML namespace that is used, the choice of any particular XML namespace prefix is implementation-specific and not significant for interoperability.

Prefix	Namespace URI	Reference
s0	urn:Microsoft.Search	
Soap	http://schemas.xmlsoap.org/wsdl/soap/	[SOAP1.1]
Tns	http://microsoft.com/webservices/OfficeServer/QueryService	
S	http://www.w3.org/2001/XMLSchema	[XMLSCHEMA1] [XMLSCHEMA2]
Soap12	http://schemas.xmlsoap.org/wsdl/soap12/	[SOAP1.2/1] [SOAP1.2/2]
(none)	http://microsoft.com/webservices/OfficeServer/QueryService	
Wsdl	http://schemas.xmlsoap.org/wsdl/	[WSDL]

Prefix	Namespace URI	Reference
T	urn:Microsoft.Search.Types	
Rrq	urn:Microsoft.Search.Registration.Request	
Rrs	urn:Microsoft.Search.Registration.Response	
Q	urn:Microsoft.Search.Query	
D	urn:Microsoft.Search.Response.Document	
R	urn:Microsoft.Search.Response	
Diffgr	urn:schemas-microsoft-com:xml-diffgram-v1	[MS-DSDIFFGRAM]

2.2.2 Messages

This specification does not define any common WSDL message definitions.

2.2.3 Elements

The following table summarizes the set of common XML schema element definitions defined by this specification. XML schema element definitions that are specific to a particular operation are described with the operation.

Element	Description
QueryPacket	This element contains all the information that makes up a search query.

2.2.3.1 QueryPacket

A QueryPacket is the content of the protocol client's QuerySoapIn, QueryExSoapIn and GetQuerySuggestionsSoapIn **SOAP messages**. It defines the search query in the necessary detail for the protocol server to run it and return search results in the QuerySoapOut, QueryExSoapOut or GetQuerySuggestionsSoapOut SOAP messages.

If the protocol server fails to interpret the search query, it MUST respond with an ERROR_BAD_QUERY error code, defined in section 2.2.5.5, in the case of the Query operation, or an equivalent SOAP fault, in the case of the QueryEx and **GetQuerySuggestions** (section 3.1.4.2) operations.

```
<s:element name="QueryPacket">
  <s:complexType>
    <s:all>
      <s:element name="Query">
        <s:complexType>
          <s:all>
            <s:element name="QueryId" type="t:GUIDType" minOccurs="0"/>
            <s:element name="OriginatorId" type="t:GUIDType" minOccurs="0"/>
            <s:element name="SupportedFormats">
              <s:complexType>
```

```
                <s:sequence>
                  <s:any processContents="skip" minOccurs="0" maxOccurs="unbounded"/>
                </s:sequence>
              </s:complexType>
            </s:element>
            <s:element name="Context">
              <s:complexType>
                <s:sequence>
                  <s:element name="QueryText">
                    <s:complexType>
                      <s:simpleContent>
                        <s:extension base="s:string">
                          <s:attribute name="language" type="s:language" use="optional"/>
                          <s:attribute name="type" type="q:QueryType" use="optional"
default="STRING"/>
                        </s:extension>
                      </s:simpleContent>
                    </s:complexType>
                  </s:element>
                  <s:element name="LanguagePreference" type="s:language" minOccurs="0"/>
                  <s:element name="Requery" minOccurs="0">
                    <s:complexType>
                      <s:sequence>
                        <s:any processContents="skip" minOccurs="0" maxOccurs="unbounded"/>
                      </s:sequence>
                    </s:complexType>
                  </s:element>
                  <s:element name="OriginatorContext" minOccurs="0">
                    <s:complexType>
                      <s:sequence>
                        <s:any processContents="skip" minOccurs="0" maxOccurs="unbounded"/>
                      </s:sequence>
                    </s:complexType>
                  </s:element>
                </s:sequence>
              </s:complexType>
            </s:element>
            <s:element name="Range" minOccurs="0">
              <s:complexType>
                <s:sequence>
                  <s:element name="StartAt" type="t:StartAtType" default="1" minOccurs="0"/>
                  <s:element name="Count" type="s:unsignedInt" minOccurs="0"/>
                </s:sequence>
                <s:attribute name="id" type="s:string" use="optional"/>
              </s:complexType>
            </s:element>
            <s:element name="Keywords" minOccurs="0">
              <s:complexType>
                <s:sequence>
                  <s:any minOccurs="0" maxOccurs="unbounded" processContents="skip"/>
                </s:sequence>
                <s:anyAttribute processContents="skip"/>
              </s:complexType>
            </s:element>
            <s:element name="OfficeContext" minOccurs="0">
              <s:complexType>
                <s:sequence>
                  <s:any minOccurs="0" maxOccurs="unbounded" processContents="skip"/>
                </s:sequence>
                <s:anyAttribute processContents="skip"/>
              </s:complexType>
            </s:element>
            <s:element name="Properties" minOccurs="0">
              <s:complexType>
                <s:sequence>
                  <s:element name="Property" minOccurs="0" maxOccurs="unbounded">
                    <s:complexType>
                      <s:attribute name="name" type="s:string"/>
                    </s:complexType>
```

```
                </s:element>
              </s:sequence>
            </s:complexType>
          </s:element>
          <s:element name="SortByProperties" minOccurs="0">
            <s:complexType>
              <s:sequence>
                <s:element name="SortByProperty" minOccurs="0" maxOccurs="unbounded">
                  <s:complexType>
                    <s:attribute name="name" type="s:string"/>
                    <s:attribute name="direction" type="q:DirectionType" use="optional"/>
                  </s:complexType>
                </s:element>
              </s:sequence>
            </s:complexType>
          </s:element>
          <s:element name="ImplicitAndBehavior" type="s:boolean" minOccurs="0"/>
          <s:element name="RelevanceModel" type="t:GUIDType" minOccurs="0"/>
          <s:element name="EnableStemming" type="s:boolean" minOccurs="0"/>
          <s:element name="EnableNicknames" type="s:boolean" minOccurs="0"/>
          <s:element name="EnablePhonetic" type="s:boolean" minOccurs="0"/>
          <s:element name="TrimDuplicates" minOccurs="0">
              <s:complexType>
              <s:simpleContent>
                <s:extension base="s:boolean">
                    <s:attribute name="onproperty" type="s:string"/>
                    <s:attribute name="keepcount" type="s:unsignedInt"/>
                    <s:attribute name="includeid" type="s:unsignedInt"/>
                  </s:extension>
                </s:simpleContent>
              </s:complexType>
            </s:element>
          <s:element name="IncludeSpecialTermResults" type="s:boolean" minOccurs="0"/>
          <s:element name="PreQuerySuggestions" type="s:boolean" minOccurs="0"/>
          <s:element name="HighlightQuerySuggestions" type="s:boolean" minOccurs="0"/>
          <s:element name="CapitalizeFirstLetters" type="s:boolean" minOccurs="0"/>
           <s:element name="ResultProvider" type="s:string" minOccurs="0" />
           <s:element name="ResubmitFlags" minOccurs="0">
             <s:complexType>
               <s:sequence>
                 <s:element name="ResubmitFlag" minOccurs="0"
maxOccurs="unbounded">
                   <s:complexType>
                     <s:attribute name="value" type="s:string"/>
                   </s:complexType>
                 </s:element>
               </s:sequence>
             </s:complexType>
           </s:element>
           <s:element name="EnableSpellcheck" type="s:string" minOccurs="0" />
           <s:element name="UserContext" minOccurs="0">
            <s:complexType>
              <s:attribute name="includeuserprofile" type="s:boolean" use="optional"/>
                <s:sequence>
                  <s:element name="UserContextData" type="s:string" minOccurs="0"
/>
                </s:sequence>
              </s:complexType>
            </s:element>
            <s:element name="FindSimilar" minOccurs="0">
              <s:complexType>
                <s:sequence>
                  <s:element name="SimilarType" type="s:string" minOccurs="0"/>
                  <s:element name="SimilarTo" type="t:SimilarToType"
minOccurs="0"/>
```

```
                    <s:element name="SortSimilar" type="s:boolean" minOccurs="0"/>
                  </s:sequence>
                </s:complexType>
              </s:element>
              <s:element name="IncludeRefinementResults" minOccurs="0">
                <s:complexType>
                  <s:sequence>
                    <s:element name="Refiners" minOccurs="0">
                      <s:complexType>
                        <s:sequence>
                          <s:element name="Refiner" type="s:string" minOccurs="0" />
                        </s:sequence>
                      </s:complexType>
                    </s:element>
                    <s:element name="MaxShallowRefinementHits" type="s:int"
minOccurs="0"/>
                  </s:sequence>
                </s:complexType>
              </s:element>
              <s:element name="RefinementFilters" minOccurs="0">
                <s:complexType>
                  <s:sequence>
                    <s:element name="RefinementFilter" type="s:string" minOccurs="0"
maxOccurs="unbounded">
                    </s:element>
                  </s:sequence>
                </s:complexType>
              </s:element>
            <s:element name="IgnoreAllNoiseQuery" type="s:boolean" minOccurs="0"/>
            <s:element name="IncludeRelevantResults" type="s:boolean" minOccurs="0"/>
            <s:element name="IncludeHighConfidenceResults" type="s:boolean" minOccurs="0"/>
          </s:all>
          <s:attribute name="domain" type="t:String255" use="optional"/>
        </s:complexType>
      </s:element>
    </s:all>
    <s:attribute name="revision" type="s:unsignedInt" use="optional"/>
    <s:attribute name="build" type="t:String255" use="optional"/>
  </s:complexType>
</s:element>
```

revision:<2> The revision of the protocol client. The Search Protocol does not limit the format or meaning of this attribute in any way beyond the limitations of its WSDL definition.

build:<3> The build number of the protocol client, a **String255** Simple Type (see section 2.2.5.8). The Search Protocol does not limit the format or meaning of this attribute in any way beyond the limitations of its WSDL definition.

Query: The parent element for the child elements that define the query.

Query.domain: The domain of the query, a **String255** Simple Type (see section 2.2.5.8). If present in the case of the Query operation, MUST be returned to the protocol client in the response unchanged. In all other ways, the domain attribute MUST be ignored by the protocol server.

Query.QueryId: A **GUID** that uniquely identifies a search query request to the Query Web service. A **GUIDType** Simple Type (see section 2.2.5.2).

Query.OriginatorId: A GUID that uniquely identifies the protocol client. A **GUIDType** Simple Type (see section 2.2.5.2).

Query.SupportedFormats: Specifies the result formats supported by the protocol client. It is currently unused and its contents MUST be ignored by the protocol server.

Query.Context: The parent element for the search query issued to the Query web service.

Query.Context.QueryText: The **query text**. The format of the query text depends on the presence and value of the **type** attribute. If the **type** attribute is not present, or its value is "STRING", the search query is specified as a SharePoint Search keyword query. If the **type** attribute is present, and its value is "MSSQLFT", the search query is specified as a SharePoint Search SQL query. If the **type** attribute is present, and its value is "FQL", the search query is specified as a **FAST Query Language (FQL) query** (described in [MS-FQL2]).

Query.Context.QueryText.language: The language for the query text. The protocol server SHOULD use this information to influence its interpretation of the query text. If this attribute is not present, the protocol server SHOULD revert to a default value, such as the language of the underlying operating system. The format of this field is the standard format for language fields in **XML**. An example would be "en-us" for US English.

Query.Context.QueryText.type: The type of query, a **QueryType** Simple Type (see section 2.2.5.3). If present, the value MUST be "STRING" if the query text is specified as a **SharePoint Search keyword query** (described in sections 2.2.10 and 2.2.11) or "MSSQLFT" if it is specified as a **SharePoint Search SQL query** (described in sections 2.2.12 and 2.2.13), or "FQL" if it is specified as a **FAST Query Language (FQL) query** (described in [MS-FQL2]). If this attribute is not present, the protocol server MUST behave as if "STRING" had been specified. If present for the **GetQuerySuggestions** (section 3.1.4.2) operation, the value MUST be "STRING".

Query.Context.LanguagePreference: The language for the context. The format of this field is the standard format for language fields in XML. For example, "en-us" represents US English.

Query.Context.Requery: Additional information about the search query in case this search query is a retry of a previous search query. It is unused and MUST be ignored by the protocol server.

Query.Context.OriginatorContext: Additional information about the protocol client. It is unused and MUST be ignored by the protocol server.

Query.Range: The range of the search results. Before sending the results of the search query to the protocol client, the protocol server MUST behave as if all search results are generated and ordered into a list. The **Range** element allows the protocol client to choose a subset of the search results from this list to be returned by the protocol server. If the **Range** element is not present, the defaults for **Query.Range.StartAt** and **Query.Range.Count** MUST apply.

Query.Range.id: The range identifier. This optional attribute is currently unused and, if present, MUST be ignored by the protocol server.

Query.Range.StartAt: The starting point of a range of search results, a **StartAtType** Simple Type (see section 2.2.5.4). Specifies the index of a search result in the list of all search results the protocol server returns as the first search result. The protocol server MUST use 1 as the index of the first search result. If this element is not present, the protocol server MUST use 1 as the default. If the total number of search results for a search query is less than the value of the **StartAt** element, the protocol server MUST return the status code ERROR_NO_RESULTS_FOUND in the case of the **Query** operation. The protocol server MUST ignore this element for the **GetQuerySuggestions** (section 3.1.4.2) operation.

Query.Range.Count: The number of search results the protocol client wants to receive, starting at the index specified in the **StartAt** element. The protocol server returns at most this many consecutive search results from the list of all search results, beginning at the index specified in the **StartAt** element. If this element is not present, the protocol server MUST use 10 as a default. If the total number of search results for a search query is greater than or equal to the value of the **StartAt** element, but less than the value of the **StartAt** element and the **Count** element combined, the protocol server MUST return as many search results as it can. Otherwise, the protocol server MUST

return **Count** search results. For the **GetQuerySuggestions** (section 3.1.4.2) operation, this element is the number of query suggestions that the protocol client wants to receive.

Query.Keywords: Specifies more information about the **tokens** in the query text. It is currently unused and MUST be ignored by the protocol server.

Query.OfficeContext: Specifies more information about the protocol client. It is currently unused and MUST be ignored by the protocol server.

Query.Properties: The names of properties that the protocol server returns for each search result if the item in the search result has a value for that property. It only applies to search queries of type "STRING" and "FQL". If the **type** attribute of the **QueryText** element is "MSSQLFT", the contents of this element MUST be ignored.

In the case of the QueryEx operation, if this element has no **child elements**, and **ResultProvider** is "SharepointSearch", the protocol server MUST return the following default properties, if available:

- WorkId
- Rank
- Title
- Author
- Size
- Path
- Description
- Write
- SiteName
- CollapsingStatus
- HitHighlightedSummary
- HitHighlightedProperties
- ContentClass
- IsDocument
- PictureThumbnailURL

In the case of the QueryEx operation, if this element has no child elements, and **ResultProvider** is "FASTSearch", the protocol server MUST return the following default properties, if available:

- WorkId
- Rank
- Title
- Author
- Size
- Path
- Description

- Write
- SiteName
- CollapsingStatus
- HitHighlightedSummary
- HitHighlightedProperties
- ContentClass
- IsDocument
- PictureThumbnailURL
- Url
- ServerRedirectedUrl
- FileExtension
- SpSiteUrl
- docvector
- fcocount
- fcoid
- PictureWidth
- PictureHeight

For a description of these properties, see section 3.1.1.1.

All properties specified here MUST be **retrievable properties**. If not all specified properties are retrievable properties, the protocol server MUST return the error code "ERROR_SERVER" in the case of the Query operation, or an equivalent SOAP fault in the case of the QueryEx operation.

If the HitHighlightedSummary property is not specified, the protocol server MUST return an empty value for the HitHighlightedProperties property, even if it is specified.<4>

If this element is used in the Query operation, the following additional restrictions apply:

- If the protocol client specifies at least one property, it MUST also specify the **Path** property.
- If it does not, the protocol server MUST return the status code "ERROR_BAD_QUERY".

For a discussion of properties, see the abstract data model in section 3.1.1.1. If the same property is listed more than once, the protocol server MUST return the status code "ERROR_BAD_QUERY"<5> in the case of the Query operation, or an equivalent SOAP fault in the case of the QueryEx operation.

If this element is used in the Query operation, the presence or absence of this element determines the output format that the protocol server uses in the Document element in the search results. See section 3.1.4.4.2.1 for details.

The protocol server MUST ignore this element for the **GetQuerySuggestions** (section 3.1.4.2) operation.

Query.Properties.Property: The property to return in the search results.

Query.Properties.Property.name: The property name.

Query.SortByProperties: The properties by which to sort the search results. It only applies to search queries where the **type** attribute of the **QueryText** element is "STRING" or "FQL". If the **type** attribute of the **QueryText** element is "MSSQLFT", this element MUST be ignored. If this element is not present, the protocol server MUST sort the search results in order of relevance to the search query, with the first search result being the most relevant. This element can contain more than one child element. In that case, the search results MUST be sorted as defined by the first **SortByProperty** element, with ties broken by the second **SortByProperty** element, and so on. If the same property is listed more than once, the protocol server MUST return the status code "ERROR_BAD_QUERY" in the case of the Query operation, or an equivalent SOAP fault in the case of the QueryEx operation.

The protocol server MUST ignore this element for the **GetQuerySuggestions** (section 3.1.4.2) operation.

Query.SortByProperties.SortByProperty: The property to sort the search results on, and how to sort on it. All actual information is contained in its attributes.

Query.SortByProperties.SortByProperty.name: The name of the property by which to sort the search results. If **ResultsProvider** is "FASTSearch", and the direction attribute is "FQLFormula", the value in this property MUST be according to a valid formula expression according to the following.

The evaluation MUST occur left-to-right and use standard mathematical-operator precedence. That is, functions and parenthetical groups MUST be evaluated first, multiplication and division operations MUST be performed next, and addition and subtraction operations MUST be performed last.

The expression element MUST NOT contain spaces.

The expression element supports the functions that are listed in the following table.

Function	Description
sqrt(*n***)**	The square root of *n*.
exp(*n***)**	The exponential function that is equivalent to **pow(2.71828182846,***n***)**.
log(*n***)**	The natural logarithm of *n*.
abs(*n***)**	The absolute value of *n*.
ceil(*n***)**	The ceiling of *n*. That is, if *n* is not a whole number, round up to the next whole number. If *n* is a whole number, use *n*.
floor(*n***)**	The floor of *n*. That is, if *n* is not a whole number, round down to the next whole number. If *n* is a whole number, use *n*.
round(*n***)**	The rounding of *n* to the nearest whole number.
sin(*n***)**	The sine of *n* radians.
cos(*n***)**	The cosine of *n* radians.

Function	Description
tan(*n*)	The tangent of *n* radians.
asin(*n*)	The arcsine, in radians, of *n*.
acos(*n*)	The arccosine, in radians, of *n*.
atan(*n*)	The arctangent, in radians, of *n*.
pow(*x*,*y*)	The value of *x* raised to the power of *y*.
atan2(*y*,*x*)	A two-argument arctangent—the angle in radians between the positive *x* axis and the specified Cartesian coordinate (*x*,*y*).
bucket(*b*,*n1*,…)	An arbitrary number of refinement bins for the **expression** element *b*. Values that follow *b* (that is, *n1*, *n2*, *n3*, and so forth) are numbers that specify refinement bin names and limits. The lowest bin value (*n1* if bins are specified in ascending order) MUST contain all the items for which *b* evaluates to a number that is less than *n1*. Subsequent refinement bins follow the same rule but MUST exclude the items that were included in previous bins. Values greater than the highest specified bin limit MUST be included in the highest bin.

Query.SortByProperties.SortByProperty.direction: The direction in which to sort the property specified in the **name** attribute. The options are "Ascending" and "Descending" (see section 2.2.5.1). If ResultProvider is FASTSearch, "FQLFormula" MUST also be a valid option. If value is "FQLFormula", the sorting MUST be according to formula specified in the name attribute, Query.SortByProperties.SortByProperty.name. If the **direction** attribute is not given, the sort MUST be performed as if "Ascending" had been specified.

Query.ImplicitAndBehavior: Specifies whether all the tokens in the query are required. If "true", the protocol server MUST find all tokens in the search query in the crawled item for it to be part of the search results. If this element is set to "false", the protocol server MUST simply find any of the tokens in the crawled item for it to be part of the search result. If this element is not present, the protocol server MUST behave the same way as if "true" had been specified. Note that the protocol makes no guarantees about which items MUST be in the search results; this is configured in the implementation of the protocol server. The protocol server MUST ignore this element for the **GetQuerySuggestions** (section 3.1.4.2) operation, or if **QueryText.type** is "FQL".

Query.RelevanceModel:<6> Specifies the unique identifier of the **ranking model** that is used for this search query. MUST be a GUID as specified in section 2.2.5.2. If this element is not present, the protocol server MUST consistently use the same ranking model for every invocation, denoted a "default ranking model". If **ResultProvider** is "FASTSearch", the protocol server MUST ignore the value. The protocol server MUST ignore this element for the **GetQuerySuggestions** (section 3.1.4.2) operation.

Query.EnableStemming: Specifies whether **inflectional forms** of the given tokens are used to locate crawled items or not. If this element is set to "true", the protocol server SHOULD use inflectional forms to locate crawled items. If this element is set to "false", the protocol server MUST NOT use inflectional forms to locate crawled items. For example, search queries with query text of

"car" will return crawled items containing the token "car" or the token "cars" if such items exist. If this element is not present, it depends on the language of the query, specified in the **Query.Contest.QueryText.language** attribute, whether the protocol server behaves as if this element had been set to "true" or to "false". Which language has which effect is an implementation detail. If the **type** attribute of the **QueryText** element is "STRING" or "FQL", this element applies to the whole query text. If the **type** attribute of the **QueryText** element is "MSSQLFT", this element applies to the FREETEXT predicate described in section 2.2.13.3.1.2. The protocol server MUST ignore this element for the **GetQuerySuggestions** (section 3.1.4.2) operation.

Query.EnableNicknames:<7> Specifies whether nickname forms of the given tokens are used to locate crawled items or not. If this element is set to "true", the protocol server SHOULD use nickname forms to locate crawled items. If this element is set to "false", the protocol server MUST NOT use nickname forms to locate crawled items. For example, search queries with query text of "steve" will return crawled items containing the token "steve" or the token "steven" if such items exist. These extra tokens MUST only be matched on **managed properties** that were created with the Nickname flag set. If this element is not present, the protocol server MUST behave as if "false" had been specified. If the **type** attribute of the **QueryText** element is "STRING", this element applies to the whole query text. If the **type** attribute of the **QueryText** element is "MSSQLFT", this element applies to the FREETEXT predicate described in section 2.2.13.3.1.2. The protocol server MUST ignore this element for the **GetQuerySuggestions** (section 3.1.4.2) operation, or if **ResultProvider** is "FASTSearch".

Query.EnablePhonetic:<8> Specifies whether phonetic representations of the given tokens are used to locate crawled items or not. If this element is set to "true", the protocol server SHOULD include crawled items that pronounce similar to the tokens entered. If this element is set to "false", the protocol server MUST NOT include crawled items that pronounce similar to the tokens entered. For example, search queries with query text of "steve" will return crawled items containing the token "steve", and also crawled items that contain tokens that pronounce similar to "steve", if such items exist. This phonetic matching MUST only be performed on managed properties that were created with the Pronunciation flag set. If this element is not present, the protocol server MUST behave as if "false" had been specified. If the **type** attribute of the **QueryText** element is "STRING", this element applies to the whole query text. If the **type** attribute of the **QueryText** element is "MSSQLFT", this element applies to the FREETEXT predicate described in section 2.2.13.3.1.2. The protocol server MUST ignore this element for the **GetQuerySuggestions** (section 3.1.4.2) operation, or if **ResultProvider** is "FASTSearch".

Query.TrimDuplicates: Specifies whether **duplicates** are removed before sorting, selecting, and sending the search results. If it is set to "true", the protocol server SHOULD perform **duplicate result removal**, if it is set to "false", the protocol server MUST NOT attempt to perform duplicate result removal. If this element is not present, the protocol server MUST behave as if "true" had been specified. If ResultProvider is "SharepointSearch", the protocol server MUST ignore any attributes when performing duplicate result removal. The protocol server MUST ignore this element for the **GetQuerySuggestions** (section 3.1.4.2) operation.

Query.TrimDuplicates.onproperty: Specifies the name of a managed property to use for duplicate result removal. The managed property MUST be of type integer. If this attribute is not present, the protocol server MUST behave as if "documentsignature" had been specified. The protocol server MUST list all available managed properties in the FASTSearchProperties table (section 3.1.4.3.2.2.2).

Query.TrimDuplicates.keepcount: Specifies the number of duplicates to return in the search results. If neither this attribute nor the TrimDuplicates.includeid attribute are present, the protocol server MUST behave as if a value of "1" had been specified for TrimDuplicates.keepcount.

Query.TrimDuplicates.includeid: Specifies a common value for a group of duplicates. This value corresponds to the value of the managed property fcoid that is returned in **query results**, and the protocol server MUST only return search results that have fcoid equal to this value. If both TrimDuplicates.keepcount and TrimDuplicates.includeid attributes are present, the protocol server MUST behave as if only the TrimDuplicates.includeid has been specified.

Query.IgnoreAllNoiseQuery: Specifies how to respond to queries where one, or more, of the full-text **predicates** contains only **noise words**. This element only applies to search queries where the **type** attribute of the **QueryText** element is "MSSQLFT". If this element is set to "true", those predicates MUST be ignored and the query MUST run as if those predicates evaluated to "true". If this element is set to "false", the search query MUST fail with a status code of "ERROR_ALL_NOISE" in the case of the Query operation, or an equivalent SOAP fault in the case of the QueryEx operation. This element MUST be ignored if the **type** attribute of the **QueryText** element is "STRING". If this element is absent, the protocol server MUST behave as if "true" had been specified. If **ResultProvider** is "FASTSearch", the protocol server MUST ignore the value. The protocol server MUST ignore this element for the **GetQuerySuggestions** (section 3.1.4.2) operation.

Query.IncludeRelevantResults: Specifies whether the RelevantResults Table is returned or not (see 3.1.4.5.2.2.1). If "true", the protocol server MUST include the RelevantResults Table in the response. Otherwise, the protocol server MUST NOT include the RelevantResults Table in the response. If this element is absent, the protocol server MUST behave as if "true" had been specified. This element is only valid for the QueryEx operation; it MUST be ignored for the Query operation. The **Query** operation MUST return the most relevant crawled items regardless of the presence and state of this element. The protocol server MUST ignore this element for the **GetQuerySuggestions** (section 3.1.4.2) operation.

Query.IncludeHighConfidenceResults: Specifies whether the HighConfidenceResults Table is returned or not (see 3.1.4.5.2.2.3). If "true", the protocol server SHOULD include the HighConfidenceResults Table in the response. If the value is set to "false", the protocol server MUST NOT include high confidence results in the response. If it is absent, the protocol server MUST behave as if the value is set to "false". This element is only valid for the QueryEx operation; It MUST be ignored for the Query operation. The Query operation MUST NOT return high confidence results regardless of the presence and state of this element. If **ResultProvider** is "FASTSearch", the protocol server MUST ignore the value. The protocol server MUST ignore this element for the **GetQuerySuggestions** (section 3.1.4.2) operation.

Query.IncludeSpecialTermResults: Specifies whether the SpecialTermResults Table is returned or not (see 3.1.4.5.2.2.2). If the value is set to "true", the response MUST include the SpecialTermResults Table. Otherwise, the response MUST NOT contain the SpecialTermResults Table. If the element is absent, the protocol server MUST behave as if "false" had been specified. This element is only valid for the QueryEx operation; It MUST be ignored for the Query operation. The **Query** operation MUST NOT return best bets regardless of the presence and state of this element. The protocol server MUST ignore this element for the **GetQuerySuggestions** (section 3.1.4.2) operation.

Query.PreQuerySuggestions: Specifies whether to get pre-query suggestions. If the value is "true", the protocol server MUST return pre-query suggestions; if the value is "false", the protocol server MUST return post-query suggestions. If this element is not present, the protocol server MUST behave as if "false" had been specified. This element is only valid for the **GetQuerySuggestions** (section 3.1.4.2) operation; it MUST be ignored for the Query and QueryEx operations.

Query.HighlightQuerySuggestions: Specifies whether to highlight emphasized parts of the suggestion. If the value is "true", the protocol server MUST enclose parts of the suggestions that it wants to emphasize in an open and closed () tag, such as word; if the value is "false" the protocol server MUST NOT emphasize suggestions with tags. If this element is not present, the protocol server MUST behave as if "false" had been specified. This element is only valid for the **GetQuerySuggestions** (section 3.1.4.2) operation; it MUST be ignored for the Query and QueryEx operations.

Query.CapitalizeFirstLetters: Specifies whether to capitalize a token in query suggestions when a token is present in the query text. If value is "true", the protocol server MUST capitalize the token; if the value is "false", the protocol server MUST NOT capitalize the token. The exact way of capitalizing tokens is an implementation detail. If this element is not present, the protocol server MUST behave as if "false" had been specified. This element is only valid for the **GetQuerySuggestions** (section 3.1.4.2) operation; it MUST be ignored for the Query and QueryEx operations.

Query.ResultProvider:<9> This element determines which **result provider** is used by the protocol server. Valid values are "SharepointSearch" or "FASTSearch"<10>. If the element is not present, or a different value is specified, the protocol server MUST behave as if "SharepointSearch" had been specified. This value MUST NOT be case-sensitive.

Query.EnableSpellcheck: Specifies how the protocol server suggests a different spelling of the search query. If the **EnableSpellcheck** element is present and the value is "Off", the protocol server MUST NOT suggest a different spelling of the search query. If the element is present and the value is "Suggest", the protocol server MUST suggest a different spelling if there is a good chance that the spelling suggestion will increase the quality of the search results. If there is a spelling suggestion, the protocol server MUST set it in the extended property **SpellingSuggestion** (see section 3.1.4.5.2.2). If the element is present and the value is "On", the protocol server MUST automatically modify the query terms before evaluating the query and returning results if there is a good chance that the modified query will increase the quality of the search results. If the query was modified, the protocol server MUST set the modified query in the extended property **QueryModification** (see section 3.1.4.5.2.2). If the element is present and the value is other than "Off", "Suggest", or "On", the protocol server MUST return a SOAP fault. If not present, the protocol server MUST behave as if the protocol client specified the value "Off". If **ResultProvider** is "SharepointSearch", the protocol server MUST ignore the value. This element is only valid for the QueryEx operation; it MUST be ignored for the Query and GetQuerySuggestions operations.

Query.ResubmitFlags: The parent element for one or more **ResubmitFlag** elements. This element and its sub-elements are only valid for the QueryEx operation; they MUST be ignored for the Query and GetQuerySuggestions operations.

Query.ResubmitFlags.ResubmitFlag: Specifies how the protocol server behaves if the protocol server returns no results from the original query. If not present or an attribute with **ResubmitFlag.value** is set to "NoResubmit", the protocol server MUST return with no results. If present and no element with **ResubmitFlag.value** is set to "NoResubmit", the protocol server MUST re-evaluate the query before returning with no search results to the client. If search query is re-evaluated, the protocol server MUST set the new search query in the extended property **QueryModification**. If **ResultProvider** is "SharepointSearch", the protocol server MUST ignore the value.

Query.ResubmitFlags.ResubmitFlag.value: MUST be one of the following values:

Value	Meaning
NoResubmit	The protocol server MUST return with zero results. Protocol server MUST ignore other **ResubmitFlag** elements.
EnableSpellcheckOnResubmit	The protocol server MUST behave as if **EnableSpellCheck** is set with a value of "On" when re-evaluating the query.
EnableSpellcheckSuggestOnResubmit	The protocol server MUST behave as if **EnableSpellCheck** is set with a value of "Suggest" when re-evaluating the query.
EnableStemmingOnResubmit	The protocol server MUST behave as if **EnableStemming** is set to "true" when re-evaluating the query.
AddSynonymsAutomatically	The protocol server MUST automatically add synonyms to query terms when re-evaluating the query.

Query.UserContext: The parent element for user context data. If **ResultProvider** is "SharepointSearch", this element and all sub-elements MUST be ignored by the protocol server.

Query.UserContext.includeuserprofile: Specifies that protocol server appends user specific context data to query before evaluating and returning search results. If present and value is "true", the protocol server MUST append user context to query. If this attribute is absent, the protocol server MUST behave as if "true" had been specified.

Query.UserContext.UserContextData: Specifies the **query context**. If present, and **best bets** and visual best bets are requested with **IncludeSpecialTermResults** and a **search setting context** is associated with best bets and visual best bets on the protocol server, the protocol server MUST only return best bets and visual best bets where all key-value pairs in the query context match all key-value pairs in the defined search setting context. How to match these key-value pairs is specific to the implementation of the protocol server. If present, and best bets and visual best bets are requested but have no search setting context associated with them, the value in query context MUST be ignored, and best bets and visual best bets with no associated search setting context MUST be returned.

The following is an **ABNF** description of the **UserContextData** syntax. ABNF is specified in [RFC5234].

```
usercontextdata = contextgroup *("|" contextgroup ) ["|"]

contextgroup    = key ":" [valuesplitter] ":" value

key             = escaped-string

valuesplitter   = escaped-string

value           = escaped-string *(valuesplitter escaped-string)

escape-sequence = "\" (":" / "\" / "|")

escaped-string  = *(escape-sequence / NONESCAPED)

NONESCAPED      = %x20-39 / %x3B-5B / %x5C-7B / %x7D-7E
```

"usercontextdata" MUST consist of one to more "contextgroup"s separated by a "|" character. Each "contextgroup" MUST consist of a "key", an optional "valuesplitter", and a "value", separated by a ":" character. "key" and "valuesplitter" MUST be "escaped-string" which is any string of **ASCII** characters with the limitation that the characters ":","|" and "\" MUST be escaped as follows: "\:", "\|" and "\\". The "key" MUST contain the name of a search setting context. The "value" can be a single "escaped-string", or multiple strings separated by "valuesplitter".

Example: UserContextData = SPS-Location::Redmond|SPS-Responsibility:;:value1;value2

In this example UserContextData value contains two context groups;

- Context group 1:
  - Key is "SPS-Location"
  - Single value is "Redmond"
- Context group 2:
  - Key is "SPS-Responsibility"
  - Value splitter is ";"
  - Multiple values are "value1" and "value2" which are separated by the value splitter string

If **ResultProvider** is "SharepointSearch", the protocol server MUST ignore the value.

Query.FindSimilar: The parent element for searching for results that are similar to already retrieved results. If present, the **FindSimilar.SimilarTo** element MUST be specified. If **ResultProvider** is "SharepointSearch", the protocol server MUST ignore the value.

Query.FindSimilar.SimilarTo: Specifies the **document vector** used for similarity comparison. The document vector indicates the most important terms in a search result, and the corresponding weight.

The weight is a float value between 0 and 1, where 1 indicates highest relevance. Format of the document vector is specified in **SimilarToType** (see 2.2.5.6), and the document vector of a search result is found in the property docvector. If **ResultProvider** is "SharepointSearch", the protocol server MUST ignore the value.

Query.FindSimilar.SimilarType: Specifies how the protocol server transforms the query when **SimilarTo** is set. The protocol server MUST append the terms in **SimilarTo** to the **QueryText** based on **SimilarType**. If value is "Find", the protocol server MUST add terms in **SimilarTo** using the OR operator. If value is "Refine", the protocol server MUST add terms in **SimilarTo** using the AND operator. If value is "Exclude", the protocol server MUST add terms in **SimilarTo** using the AND NOT operator. See section 2.2.11.2 for a specification of Keyword Query operators. If value is "None", the protocol server MUST NOT transform the query text. If the value is any other **string**, the protocol server MUST return a SOAP fault with the **Code** element set to "soap:Receiver" and the **Reason** element set to "System.ArgumentException", as specified in [SOAP1.1] section 4.4 or [SOAP1.2/1] section 5.4. If not set, but **SimilarTo** is set, the protocol server MUST behave as if "Find" was specified. If set, but **SimilarTo** is not set, the protocol server MUST ignore this value and MUST NOT transform the query text. If **ResultProvider** is "SharepointSearch", the protocol server MUST ignore the value.

Query.FindSimilar.SortSimilar: Specifies sorting of search results based on similarity of the results. If present and value is "true", the protocol server SHOULD sort according to similarity. How to sort according to similarity is specific to the implementation of the protocol server. If present, and value is "false" and **SortByProperties** is specified, sorting MUST be by **SortByProperties**. If **SortByProperties** is not specified, sorting MUST be by rank. If not present, but **SimilarTo** is specified, sorting MUST be by similarity. If **ResultProvider** is "SharepointSearch", the protocol server MUST ignore the value.

Query.IncludeRefinementResults: The parent element for refinement results. This element and its sub-elements are only valid for the QueryEx operation; they MUST be ignored for the Query and GetQuerySuggestions operations.

Query.IncludeRefinementResults.Refiners: The parent element for list of **refiners** the protocol server SHOULD return based on query results. If the **Refiners** element is absent or empty, the protocol server MUST NOT return refinement results. If **ResultProvider** is "SharepointSearch", the protocol server MUST ignore the value.

Query.IncludeRefinementResults.Refiners.Refiner: Specifies a refiner the protocol server returns based on query results. The following is an ABNF description of the **Refiner** syntax. ABNF is specified in [RFC5234].

```
refiner           = property-name ["(" refiner-settings ")"]

refiner-settings  = 1(refiner-setting) *("," refiner-setting)
refiner-setting   = discretize / sort / cutoff

discretize        = "discretize=manual" 2*["/" discretize-bucket]
discretize-bucket = date-date [date-time] / 1*digit

year              = 4digit
month             = 2digit
day               = 2digit
hour              = 2digit
minute            = 2digit
second            = 2digit / 2digit "Z"
date-date         = year "-" month "-" day
date-time         = hour ":" minute ":" second

sort              = "sort=" sort-algorithm "/" sort-direction
sort-algorithm    = "frequency" / "name" / "number"
sort-direction    = "descending" / "ascending"

cutoff            = "cutoff=" cutoff-frequency "/" cutoff-minbuckets "/" cutoff-maxbuckets
```

```
cutoff-frequency  = 1*digit
cutoff-minbuckets = 1*digit
cutoff-maxbuckets = 1*digit

property-name     = 1*alpha ; MUST be name of refinable property
```

Refiner MUST be specified by using the name of a managed property that has an associated refiner, and each refiner can be specified with additional refiner parameters.

The "discretize" parameter specifies the custom **buckets** the protocol server returns for a refinable property that is numeric or of type datetime. All buckets MUST be of same type, that is either datetime or numeric.

The "cutoff" parameter specifies that the protocol server limits the amount of refiner results it returns:

- "cutoff-frequency" specifies that the protocol server does not return a refinement value if the number of occurrences of the refinement value in a result set is less than or equal to this value.
- "cutoff-minbuckets" specifies the minimum value for frequency cutoff. If the number of unique refinement values for the query is less than this value, there is no frequency cutoff, and the protocol server MUST return all refinement values. When cutoff-frequency is used, this parameter specifies a minimum number of unique refinement values that the protocol server MUST return regardless of the number of occurrences.
- "cutoff-maxbuckets" specifies the maximum number of unique refinement values (buckets) that the protocol server returns for a navigator.

The "sort" parameter defines how the buckets within a refiner of type string are sorted;

- "sort-algorithm" specifies how the protocol server orders the refinement buckets.
  - "frequency" specifies that the order MUST be by occurrence within the refinement bucket.
  - "name" specifies that the order MUST be by label name.
  - "number" specifies that the protocol server treats the strings as numeric and use numeric sorting.
- "sort-direction" specifies the sorting direction to be descending or ascending.

If **ResultProvider** is "SharepointSearch", the protocol server MUST ignore the value.

Query.IncludeRefinementResults.MaxShallowRefinementHits: Specifies the number of results to be used to calculate refinement results. The protocol server MUST apply this only to refiners that are specified to have **shallow refinement**. If not set, a default value of 100 MUST be used by protocol server. If **ResultProvider** is "SharepointSearch", the protocol server MUST ignore the value.

Query.RefinementFilters: The parent element for refinement filters. If provided, **RefinementFilter** elements specify a drill down into a search result. If **ResultProvider** is "SharepointSearch", the protocol server MUST ignore this element and all sub-elements. This element and its sub-elements are only valid for the QueryEx operation; they MUST be ignored for the Query and GetQuerySuggestions operations.

Query.RefinementFilters.RefinementFilter: Specifies a **refinement token** representing a **query refinement**. Refinement tokens are returned as part of the **RefinementResults** table (see 3.1.4.5.2.2.4) for the previous query.

2.2.4 Complex Types

This specification does not define any common XML schema complex type definitions.

2.2.5 Simple Types

The following table summarizes the set of common XML schema simple type definitions defined by this specification. XML schema simple type definitions that are specific to a particular operation are described with the operation.

Simple type	Description
DirectionType	**Sort order** of search results
GUIDType	Defines a GUID.
QueryType	Type of a search query
StartAtType	Index of the first search result requested in the search query
StatusType	Result code of a search query or other types of SOAP requests the Search Protocol supports.
SimilarToType	Search for results with similar terms
String2048	String of up to 2048 characters
String255	String of up to 255 characters

2.2.5.1 DirectionType

Target namespace: urn:Microsoft.Search.Query

This type defines the sort order for a listing of search results.

```
<s:simpleType name="DirectionType">
  <s:restriction base="s:string">
    <s:enumeration value="Ascending"/>
    <s:enumeration value="Descending"/>
    <s:enumeration value="FQLFormula"/>
  </s:restriction>
</s:simpleType>
```

The following table defines the allowable values for **DirectionType**:

Value	Meaning
Ascending	Sort the search results in ascending order.
Descending	Sort the Search results in descending order.

Value	Meaning
FQLFormula<11>	Sort the search results by formula provided in **Query.SortByProperties.SortByProperty.name**, as specified in section 2.2.3.1.

2.2.5.2 GUIDType

Target namespace: urn:Microsoft.Search.Types

This type defines a GUID, with or without enclosing curly braces.

```
<s:simpleType name="GUIDType">
  <s:restriction base="s:string">
    <s:pattern value="\{[A-Fa-f0-9]{8}-([A-Fa-f0-9]{4}-){3}[A-Fa-f0-9]{12}\}"/>
    <s:pattern value="[A-Fa-f0-9]{8}-([A-Fa-f0-9]{4}-){3}[A-Fa-f0-9]{12}"/>
  </s:restriction>
</s:simpleType>
```

2.2.5.3 QueryType

Target namespace: urn:Microsoft.Search.Query

Defines the type of a search query.

```
<s:simpleType name="QueryType">
  <s:restriction base="s:string">
    <s:enumeration value="MSSQLFT"/>
    <s:enumeration value="STRING"/>
    <s:enumeration value="FQL"/>
  </s:restriction>
</s:simpleType>
```

The following table defines the allowable values for **QueryType**:

Value	Meaning
MSSQLFT<12>	The search query is specified as a SharePoint Search SQL query.
STRING	The search query is specified as a SharePoint Search keyword query.
FQL<13>	The search query is specified as a **FAST Query Language (FQL) query**, as described in [MS-FQL2].

2.2.5.4 StartAtType

Target namespace: urn:Microsoft.Search.Types

Defines the index of first search result requested in the search query. The lowest result index is '1'. For example, if the protocol client shows ten search results per page, then the search query to request the second page would set the value of **StartAtType** to 11.

```
<s:simpleType name="StartAtType">
  <s:restriction base="s:unsignedInt">
```

```
      <s:minInclusive value="1"/>
    </s:restriction>
  </s:simpleType>
```

2.2.5.5 StatusType

Target namespace: urn:Microsoft.Search.Response

This type contains the result code of a search query or other types of SOAP requests that the Search Protocol supports.

```
<s:simpleType name="StatusType">
  <s:restriction base="s:string">
    <s:enumeration value="SUCCESS"/>
    <s:enumeration value="ERROR_ALL_NOISE"/>
    <s:enumeration value="ERROR_NO_RESPONSE"/>
    <s:enumeration value="ERROR_BAD_PROPERTY"/>
    <s:enumeration value="ERROR_BAD_QUERY"/>
    <s:enumeration value="ERROR_BAD_SCOPE"/>
    <s:enumeration value="ERROR_BAD_REQUEST"/>
    <s:enumeration value="ERROR_NO_RESULTS_FOUND"/>
    <s:enumeration value="ERROR_NO_QUERY"/>
    <s:enumeration value="ERROR_NO_AUTHORIZATION"/>
    <s:enumeration value="ERROR_SERVER"/>
  </s:restriction>
</s:simpleType>
```

The following table defines the allowable values for **StatusType**:

Value	Meaning
SUCCESS	The protocol server finished the requested operation.
ERROR_ALL_NOISE	All tokens in query text were noise words.
ERROR_NO_RESPONSE	The protocol server failed to contact the **Office SharePoint Server Search service**.
ERROR_BAD_PROPERTY	A managed property that was referenced in the query does not exist.
ERROR_BAD_QUERY	The search query is malformed.
ERROR_BAD_SCOPE	The selected **search scope** is invalid.
ERROR_BAD_REQUEST	The SOAP Message was malformed.
ERROR_NO_RESULTS_FOUND	The request to the Office SharePoint Server Search service finished successfully, but no items matched the specified search query.
ERROR_NO_QUERY	No query text was specified.
ERROR_NO_AUTHORIZATION	The Office SharePoint Server Search service does not have permission to perform a required operation.
ERROR_SERVER	A generic error occurred.

2.2.5.6 SimilarToType

This type defines the format used when searching for similar results. The protocol client can use this type to indicate the most important terms for a search result.

```
<s:simpleType name="SimilarToType">
  <s:restriction base="s:string">
    <s:pattern value="(\[([\w ])+,(([0-9]+)?\.)?[0-9]+\])+"/>
  </s:restriction>
</s:simpleType>
```

2.2.5.7 String2048

Target namespace: urn:Microsoft.Search.Types

This type defines a string of up to 2048 characters.

```
<s:simpleType name="String2048">
  <s:restriction base="s:string">
    <s:minLength value="0"/>
    <s:maxLength value="2048"/>
  </s:restriction>
</s:simpleType>
```

2.2.5.8 String255

Target namespace: urn:Microsoft.Search.Types

This type defines a string of up to 255 characters.

```
<s:simpleType name="String255">
  <s:restriction base="s:string">
    <s:minLength value="0"/>
    <s:maxLength value="255"/>
  </s:restriction>
</s:simpleType>
```

2.2.6 Attributes

This specification does not define any common XML schema attribute definitions.

2.2.7 Groups

This specification does not define any common XML schema group definitions.

2.2.8 Attribute Groups

This specification does not define any common XML schema attribute group definitions.

2.2.9 Common Data Structures

This specification does not define any common XML schema data structures.

2.2.10 SharePoint Search Keyword Syntax v1

SharePoint Search Keyword Syntax v1<14> specifies a syntax for search queries that enables users to formulate search queries in a structure that resembles natural English. The syntax enables users to use some of the advanced features of the protocol server.

2.2.10.1 Keyword Search

In the case when none of the following sections "Phrasal Matching", "Keyword Exclusion", "Keyword Inclusion", or "Property Searches" apply, the protocol server MUST interpret the query text that the user typed as a sequence of tokens. The exact way the protocol server extracts the tokens from the query text is an implementation detail, but it SHOULD take into account the language in which the search query was formulated. By default, an item matches the search query if it contains all of the extracted tokens. This behavior can be modified such that an item matches the search query if it contains any of the extracted tokens.

There are some exceptions to this interpretation of SharePoint Search keyword queries. Those exceptions are detailed in the following sections.

2.2.10.2 Phrasal Matching

If a part of the query text is enclosed in double quotes, that part MUST be interpreted as a phrase. In detail, a phrase MUST match an item if the item contains all of the tokens in the phrase in the same order that they occur in the phrase.

Phrasal matching MUST take precedence over the other features. In other words, the other features of SharePoint Search keyword syntax MUST NOT be recognized within a phrase. For example, the search query ""mirror-universe"" matches the tokens in the phrase "mirror-universe" in order. The minus character MUST NOT be interpreted as the keyword exclusion feature described in the next section.

2.2.10.3 Keyword Exclusion

If a token is preceded by a minus character, items MUST match that part of the search query if they do not contain the token. For example, the search query "mirror –universe" matches items that contains the token "mirror" but not the token "universe".

This feature MUST NOT be combined with the phrasal matching feature. The results of the search query "trek –"mirror universe"" are undefined.

2.2.10.4 Keyword Inclusion

If a token is preceded by a plus character, the items MUST match that part of the search query if they contain the token. This syntax only makes sense for queries where the **ImplicitAndBehavior** element in the QueryPacket is set to false. It MUST be ignored otherwise.

This feature MUST NOT be combined with the phrasal matching feature. The results of the search query "trek +"mirror universe"" are undefined.

2.2.10.5 Property Searches

A sequence of the form "property:token" MUST be interpreted as a property search query. Items MUST match this expression if the specified property contains the specified token. If the property name contains whitespace characters, it MUST be enclosed in double quotes. The property specified in this way MUST be searchable with the full-text. If the specified property is not full-text searchable, the sequence MUST NOT be interpreted as a property search query but instead be interpreted simply as a sequence of tokens.

It is possible to search for a phrase in a single property as well. The format for this is "property: "phrase"". Items that contain the given phrase, as defined in section 2.2.10.2, in the given property MUST match this expression.

Finally, it is possible to negate a property search query by preceding the entire expression with a minus character.

2.2.11 SharePoint Search Keyword Syntax v2

SharePoint Search Keyword Syntax v2<15> specifies a syntax for search queries that enables users to formulate search queries in a structure that resembles natural language and at the same time allows users to specify Boolean matching rules on text and properties of the searched items. The syntax restricts the search query to the task of obtaining a list of search results from the protocol server.

The following is an ABNF description of the SharePoint Search Keyword Syntax, which is used in the Query and QueryEx operations. ABNF is specified in [RFC5234]. Following section 2.4 in [RFC5234], this description assumes **Unicode** as its external encoding. As such, the terminal values of this grammar represent **Unicode code points**.

2.2.11.1 Common Definitions

A white space character is a character that delimits other syntax literals.

```
ws           =  %x09 / %x20 / %x0A / %x0D; tab, space, line feed, carriage return
```

An implementation of the SharePoint Search Keyword syntax MUST support the following operators:

```
words-operator = %x57.4F.52.44.53 ;case-sensitive "WORDS"
near-operator = %x4E.45.41.52 ;case-sensitive "NEAR"
not-operator = %x4E.4F.54 ;case-sensitive "NOT"
and-operator = %x41.4E.44 ;case-sensitive "AND"
or-operator = %x4F.52 ;case-sensitive "OR"
all-operator = %x41.4C.4C ;case-sensitive "ALL"
any-operator = %x41.4E.59 ;case-sensitive "ANY"
none-operator = %x4E.4F.4E.45 ;case-sensitive "NONE"
```

Supported literals are listed as follows.

A non-terminal symbol property-quoted-string introduces string literals enclosed in double quotes. They can contain any Unicode character except double quotes. While string literals MUST NOT be limited in their length, they are limited by the length of the overall SharePoint Search Keyword query.

```
property-quoted-string = %x22 1*property-dquote-char %x22 ; %x22 is double quote
property-dquote-char   =   %x00-21 / %x23-%x10FFFF
```

A non-terminal symbol property-token introduces string literals not enclosed in double quotes. They can contain any Unicode character from the following Unicode General Categories: Lu, Ll, Lt, Lm, Lo, Nd, and Pc (these categories are specified in [UNICODE] section 4.5 General Category—Normative). General Category for each UNICODE code point is specified in [UNICODE3.1].

```
property-token = 1*property-char
property-char = %x30-39 / %x41-5A / %x5F / %x61-7A / %xAA / %xB5 / %xBA / %xC0-D6 / %xDF /
%xE0-FF ;defined as example for the codes lower than %x100
```

A language literal represents one of the languages, whether spoken, written, signed, or otherwise signaled. An implementation MUST support languages as specified in the [RFC4646] section 2.1.

```
property-language = langtag ; langtag is specified in [RFC4646] section 2.1
```

Boolean literals represent logical values and MUST be either "true" or "false":

```
property-boolean = "TRUE" / "FALSE"
```

Numeric literals represent numbers, including positive and negative integers and floating point values.

A non-terminal symbol property-decimal introduces integer literals. A protocol server can recognize a sequence of characters as property-decimal if it matches decimal non-terminal as specified in section 2.2.13.1. Alternatively a protocol server can take into account the language in which the search query was formulated and recognize the string representation of the integer specific to that language.

A non-terminal symbol property-float introduces floating point values. A protocol server can recognize a sequence of characters as property-float if it matches float non-terminal as specified in section 2.2.13.1. Alternatively a protocol server can take into account the language in which the search query was formulated and recognize the string representation of the floating point values specific to that language.

Date literals represent specific dates. The protocol server can allow time part to be present with the date. If time part is present, the protocol server MUST ignore it.

A non-terminal symbol property-date-no-ws introduces a date literal that MUST not contain any white space characters (specified by ws non-terminal symbol in this section). A protocol server can recognize a sequence of characters as property-date-no-ws if it matches date-date non-terminal as specified in section 2.2.13.1. Alternatively a protocol server can take into account the language in which the search query was formulated and recognize the string representation of dates specific to that language.

A non-terminal symbol property-date introduces a date literal that can contain white space characters (specified by ws non-terminal symbol in this section). A protocol server can recognize a sequence of characters as property-date if it matches date-value non-terminal as specified in section 2.2.13.1. Alternatively a protocol server can take into account the language in which the search query was formulated and recognize the string representation of dates specific to that language.

An implementation MUST support names that represent date intervals relative to the current date as listed:

Name of date interval	Description
Yesterday	The latest day that precedes the current date.
This week	The entire week that includes the current date.
This month	The entire month that includes the current date.
Last month	The latest entire month that precedes the current month.
This year	The entire year that includes the current date.
Last year	The latest entire year that precedes the current year.

```
property-date-named = "yesterday" / %x22 "yesterday" %x22 / %x22 "this week" %x22 / %x22
"this month" %x22 / %x22 "last month" %x22 / %x22 "this year" %x22 / %x22 "last year" %x22
```

A part of the query text enclosed in double quotes is a text-phrase. It MUST NOT itself contain any double quotes. A crawled item MUST match a phrase if it contains all tokens that appear between the quotes, in the same order that they appear in. As a special case, if the last character in the phrase is an asterisk character, all tokens in the phrase MUST be interpreted as prefixes. In other words, a crawled item MUST match the phrase if it contains tokens that share a common prefix with the tokens between the quotes, and these tokens appear in the same order.

```
text-phrase = %x22 0*text-dquote-char %x22 ; %x22 is double quote
text-dquote-char   =   %x00-21 / %x23-%x10FFFF ; %x22 is double quote
```

The protocol server MUST interpret the query text that is not a part of other syntax constructs as a sequence of tokens. The exact way the protocol server extracts the tokens from the query text is an implementation detail, but it SHOULD take into account the language in which the search query was formulated.

```
text-token = 1*( %x01-08 / %x0B-0C / %x0E-21 / %x23-27 / %x2A-10FFFF ); %x28 is "(", %x29 is
")"
```

2.2.11.2 Keyword Query

An implementation of the SharePoint Search Keyword Syntax MUST interpret query as a sequence of property expressions interchanging with text expressions. Property expression is used to specify matching rules on named properties of crawled items. Text expression is used to specify matching rules on the text of items. The protocol server MUST include the item in the list of search results when every property and text expression matches the item. The protocol server MUST NOT include an item in the list of results otherwise.

The query MUST NOT be longer than an implementation-defined number of Unicode characters. This length limit MUST be greater than 1024 characters.

This is the top-level element of the grammar:

```
keyword-query = [*ws property-expression [*ws and-operator]] *(*ws text-expression
[*ws and-operator] *ws property-expression [*ws and-operator]) *ws text-expression
*ws
keyword-query =/ [*ws text-expression [*ws and-operator]] *(*ws property-expression
[*ws and-operator] *ws text-expression [*ws and-operator]) *ws property-expression
*ws
```

2.2.11.3 Property Expression

The protocol server MUST interpret property expressions as a sequence of one or more property constraints joined implicitly (without AND or OR operators between them) or using operators AND or OR. If there are no operators between constraints, the protocol server MUST interpret an expression in the same way as if AND was present. The protocol server MUST allow a constraint to be negated by using the NOT operator.

The protocol server MUST use the following rules to match the property expression against the items:

- When an AND operator is used, the expression matches the item if both constraints on the left and right of the operator match the item;
- When an OR operator is used, the expression matches the item if any one or both constraints, on the left and right of the operator match the item;
- When a NOT operator is used, the expression matches the item if the constraint does not match the item;

The ordering of priority for AND, OR, and NOT operators is explicitly expressed by the following grammar rules. NOT has highest priority, then AND, then OR, and an implicit join has the lowest priority.

```
property-expression = property-or-expression
property-expression =/ property-expression *ws property-or-expression
property-or-expression = property-and-expression
property-or-expression =/ property-or-expression *ws or-operator *ws property-and-expression
property-and-expression = property-not-expression
property-and-expression =/ property-and-expression *ws and-operator *ws property-not-
expression
property-not-expression = property-constraint
property-not-expression =/ not-operator *ws property-constraint
```

2.2.11.4 Property Constraint

The protocol server MUST support two ways of specifying property constraint. The first is property restriction - qualified or not qualified. In this case, the constraint MUST match the item if the restriction matches the item. The second way is the property expression enclosed in parenthesis. In this case, the constraint MUST match the item if the enclosed expression matches the item.

```
property-constraint = property-qualified-restriction / property-restriction
property-constraint =/ "(" *ws property-expression *ws ")"
```

2.2.11.5 Property Restriction

The property restriction is a basic element in property expression. It specifies a Boolean predicate on one property of the item. The protocol server MUST recognize a sequence of characters as a property restriction if it starts with a property name, followed by one of the operators, followed by a value, without additional characters between name, operator, and value.

The property specified by its name MUST be searchable with the full-text. If the specified property is not full-text searchable, the sequence MUST NOT be interpreted as a property restriction but instead MUST be interpreted simply as a sequence of text tokens.

The type of the value MUST match the type of the property. The property restriction MUST match the item if the value provided in the query matches the value of the item's property according to the operator.

The operator MUST be one of the following:

Operator	Description
:	The property of the item contains the specified value.
=	The property of the item is equal to the specified value.

Operator	Description
<>	The property of the item is not equal to the specified value.
>	The property of the item is greater than the specified value.
>=	The property of the item is greater or equal to the specified value.
<	The property of the item is less than the specified value.
<=	The property of the item is less than or equal to the specified value.

```
property-restriction = property-name property-operator property-value
property-name = property-token / property-quoted-string
property-value = property-typed-value / property-token / property-quoted-string
property-operator = ":" / "=" / "<>" / ">" / ">=" / "<" / "<="
```

2.2.11.6 Property Typed Value

The protocol server MUST support the following types of values: language, Boolean, float, decimal and date. The protocol server MUST accept both quoted and unquoted forms of values.

For decimal and date value the protocol server MUST support ranges. The protocol server MUST interpret range A..B as set of values from A to B inclusive.

```
property-typed-value = property-language / %x22 *ws property-language *ws %x22
property-typed-value =/ property-boolean / %x22 *ws property-boolean *ws %x22
property-typed-value =/ property-float / %x22 *ws property-float *ws %x22
property-typed-value =/ property-decimal [".." property-decimal] / %x22 *ws property-decimal
[*ws ".." *ws property-decimal] *ws %x22
property-typed-value =/ property-date-named / property-date-no-ws [".." property-date-no-ws]
/ %x22 *ws property-date [*ws ".." *ws property-date] *ws %x22
```

2.2.11.7 Property Qualified Restriction

If a property restriction is preceded by a minus character, the protocol server MUST evaluate it the same way as if it was preceded by the NOT operator. For example, -size=100 is equivalent to NOT size=100, which is in turn equivalent to size<>100.

If a property restriction is preceded by a plus character, the protocol server MUST ignore the plus character. For example, size=100 is equivalent to +size=100.

```
property-qualified-restriction = "+" property-restriction
property-qualified-restriction =/ "-" property-restriction
```

2.2.11.8 Text Expression

The protocol server MUST interpret text expressions as a sequence of one or more text restrictions joined implicitly (without AND, OR, +, or - operators between them) or using operators AND and OR.

If there are no operators between restrictions, the protocol server MUST interpret an expression depending on the value of the **ImplicitAndBehavior** element in the QueryPacket. If the **ImplicitAndBehavior** element is set to "true", then the text expression MUST match the item if every restriction is matching the item. If the **ImplicitAndBehavior** element is set to "false" and the query string does not contain any of the operators AND, OR, NOT, NEAR, or WORDS, then the expression MUST match the item if the item contains any unqualified tokens and all other restrictions match the item. If any of the preceding operators are used in the query string then the protocol server MUST interpret the expression as if the **ImplicitAndBehavior** element was set to "true".

The protocol server MUST allow a restriction to be negated by using the NOT operator.

The protocol server MUST use the following rules to match the text expression against the items:

- When AND operator is used, the expression matches the item if both restrictions on the left and right of the operator match the item;
- When OR operator is used, the expression matches the item if any one or both restrictions on the left and right of the operator match the item;
- When NOT operator is used, the expression matches the item if the restriction does not match the item;

The ordering of priority is explicitly expressed by the following grammar rules. NOT has highest priority, then AND, then OR, and an implicit join has the lowest priority.

```
text-expression = text-or-expression
text-expression =/ text-expression *ws text-or-expression
text-or-expression = text-and-expressionCommon
text-or-expression =/ text-or-expression *ws or-operator *ws text-and-expression
text-and-expression = text-not-expression
text-and-expression =/ text-and-expression *ws and-operator *ws text-not-expression
text-not-expression = text-restriction
text-not-expression =/ not-operator *ws text-restriction
```

2.2.11.9 Text Restriction

The protocol server MUST interpret text restriction as one of the following:

- NEAR restriction, which also covers single token, prefix, or synonym. The text restriction MUST match the item if the NEAR restriction matches the item;
- Qualified or unqualified phrase or qualified token. The text restriction MUST match the item if the token or phrase matches the item;
- Text expression enclosed in parenthesis. The text restriction MUST match the item if the enclosed expression matches the item.
- ALL restriction. The text restriction MUST match the item if the ALL restriction matches the item.
- ANY restriction. The text restriction MUST match the item if the ANY restriction matches the item.
- NONE restriction. The text restriction MUST match the item if the NONE restriction matches the item.

```
text-restriction = text-near-restriction
text-restriction =/ "(" *ws text-expression *ws ")"
text-restriction =/ text-phrase / text-qualified-phrase / text-qualified-token
text-restriction =/ text-all-restriction
text-restriction =/ text-any-restriction
```

```
text-restriction =/ text-none-restriction
```

The NEAR restriction matches one or more tokens, prefixes, or synonyms that occur close to each other in the item. This restriction MUST match the item whenever all tokens, prefixes, or synonyms are found in the item with the preserved order and distance not greater than that implementation-defined value. That value MUST be greater than 1.

If there is only one token, prefix, or synonym is specified, then it MUST match the item if the item contains that single token, prefix, or synonym.

```
text-near-restriction = text-syn-term
text-near-restriction =/ text-near-restriction 1*ws near-operator 1*ws text-syn-term
text-syn-term = text-token / text-prefix / text-synonym
```

The ALL restriction MUST match the item if it contains all the tokens supplied between the parentheses.

```
text-all-restriction = all-operator "(" *ws 1*text-token *ws ")"
```

The ANY restriction MUST match the item if it contains one or more of the tokens supplied between the parentheses.<16>

```
text-any-restriction = any-operator "(" *ws 1*text-token *ws ")"
```

The NONE restriction MUST match the item if it contains none of the tokens supplied between the parentheses.

```
text-none-restriction = none-operator "(" *ws 1*text-token *ws ")"
```

2.2.11.10 Synonyms

The protocol server MUST support the definition of synonyms in a query string that uses the WORDS operator. The WORDS operator MUST match the item if the item contains one or more of the tokens or phrases that are specified inside the parenthesis.

```
text-synonym = words-operator "(" *ws text-syn-list *ws ")"
text-syn-list = text-term
text-syn-list =/ text-syn-list [*ws ","] *ws text-term
```

The protocol server MUST ignore the trailing asterisk character in a text-prefix. For example, WORDS(word1* word2) is equivalent to WORDS(word1 word2) which matches the items that contain token word1 or token word2 or both.

The protocol server MUST ignore preceding plus and minus characters in qualified tokens and qualified phrases. For example, WORDS(+word1 –"word2 word3") is equivalent to WORDS(word1 "word2 word3") which matches the items that contain token word1 or phrase "word2 word3" or both.

```
text-term = text-token / text-prefix / text-qualified-token / text-phrase / text-qualified-
phrase
```

2.2.11.11 Basic Text Blocks

If a token is preceded by a minus character, the items MUST match that part of the search query if they do not contain the token.

If a token is preceded by a plus character, the items MUST match that part of the search query if they do contain the token. This syntax only makes sense for queries where the **ImplicitAndBehavior** element in the QueryPacket is set to "false" and there are no operators (AND, OR, NOT, WORDS, or NEAR) in the query. A plus character MUST be ignored otherwise.

```
text-qualified-token =/ "+" text-token
text-qualified-token =/ "-" text-token
```

If a token is followed by an asterisk character, the items MUST match that part of the search query if they contain one or more tokens that have a specified prefix.

```
text-prefix = text-token "*"
```

If a phrase is preceded by a minus character, items MUST match that part of the search query if they do not contain the phrase.

If a phrase is preceded by a plus character, the items MUST match that part of the search query if they contain the phrase. This syntax only makes sense for queries where the **ImplicitAndBehavior** element in the QueryPacket is set to "false" and there are no operators (AND, OR, NOT, WORDS, or NEAR) in the query. A plus character MUST be ignored otherwise.

```
text-qualified-phrase =/ "+" text-phrase
text-qualified-phrase =/ "-" text-phrase
```

2.2.12 SharePoint Search SQL Syntax v1

SharePoint Search SQL Syntax v1<17> specifies a syntax for search queries that is similar to the language used by SQL server. The syntax restricts the search query to task of obtaining a list of search results from the protocol server.

The following is an ABNF description of the SharePoint Search SQL syntax, which is used in the Query and QueryEx operations. ABNF is specified in [RFC5234]. Following section 2.4 in [RFC5234], this description assumes Unicode as its external encoding. As such, the terminal values of this grammar represent Unicode code points.

2.2.12.1 Common Definitions

The first definitions are a set of basic definitions that are used throughout the rest of the description of the SharePoint Search SQL Syntax:

```
ws       =      %x09 / %x20 / %x0A / %x0D
 ; tab, space, line feed, carriage return
lowalpha  =      %x61-7A
upalpha   =      %x41-5A
alpha     =      lowalpha / upalpha
digit     =      %x30-39
alphanum  =      alpha / digit
```

Identifiers specify the names of properties. Identifiers MUST be between 1 and 128 characters in length. There are two types of identifiers, regular identifiers and delimited identifiers. Regular

identifiers MUST be formed from a limited set of characters and begin with a letter whereas delimited identifiers can contain any Unicode character. Regular identifiers MUST NOT be one of the keywords in SharePoint Search SQL Syntax: An identifier that is also a keyword in SharePoint Search SQL Syntax MUST be expressed using a delimited identifier. Delimited identifiers MUST be surrounded by double quotes. To use a double quote as part of a delimited identifier, it MUST be encoded as two double quotes in succession. Content is a special identifier. It MUST refer to a property that represents the body of the crawled items. Finally, aliases are regular identifiers that begin with a pound sign.

```
identifier            =      regular-identifier / delimited-identifier
regular-identifier    =      alpha *127identifier-char
identifier-char       =      alphanum / "_"
delimited-identifier  =      %x22 1*128quoted-dquote-char %x22      ; %x22 is double quote
quoted-dquote-char    =      %x00-21 / %x22.22 / %x23-%x10FFFF      ; %x22 is double quote
alias-name            =      "#" regular-identifier
```

An implementation of the SharePoint Search SQL Syntax MUST support various kinds of literals. One of these is the string literal. While string literals MUST NOT be limited in their length, they are limited by the length of the overall SharePoint Search SQL query. They can contain any Unicode character. They MUST be enclosed in single quotes. To use a single quote as part of a string, it MUST be encoded as two single quotes in succession.

```
string              =      "'" *quoted-quote-char "'"
quoted-quote-char   =      %x00-26 / "''" / %x28-%x10FFFF      ; single quote is %x27
```

Numeric literals represent numbers, including positive and negative integers expressed in decimal or hexadecimal, as well as floating point values including exponential notation using the character 'e'.

```
positive-decimal    =      1*digit
negative-decimal    =      "-" positive-decimal
decimal             =      positive-decimal / negative-decimal
hex-digit           =      digit / %x41-46 / %x61-66      ; digits and a-f
positive-hex        =      "0x" 1*hex-digit
negative-hex        =      "-" positive-hex
hex                 =      positive-hex / negative-hex
positive-float      =      1*digit ["." *digit] / "." 1*digit
negative-float      =      "-" positive-float
positive-float-exp  =      positive-float ["e" 1*digit]
negative-float-exp  =      "-" positive-float-exp
float               =      positive-float-exp / negative-float-exp
```

Boolean literals represent logical values and MUST be either "TRUE" or "FALSE":

```
boolean    =    "TRUE" / "FALSE"
```

Date literals represent specific dates or times. They MUST be enclosed in single quotation marks, and they MUST be in the form year/month/day or year-month-day. The year MUST be specified as a four-digit value. Time values MUST be in the form hours:minutes:seconds. The protocol server MUST understand concatenated date and time values as a timestamp value.

```
year      =      4digit
month     =      [digit] digit
day       =      [digit] digit
hour      =      [digit] digit
minute    =      [digit] digit
second    =      [digit] digit
date-date =      year "/" month "/" day
date-date = /      year "-" month "=" day
```

```
date-time =      hour ":" minute ":" second
date      =      "'" date-date [1*ws date-time] "'"
```

In some places, the protocol client MUST specify a language for a token or a phrase. In the SharePoint Search SQL Syntax, languages MUST be specified by an **LCID**.

```
lcid      =      decimal / hex
```

In several places, an implementation of the SharePoint Search SQL syntax MUST support the following comparison operators:

```
operator      =      "=" / "!=" / "<>" / ">" / ">=" / "<" / "<="
```

2.2.12.2 Query

An implementation of SharePoint Search SQL Syntax MUST support two statements:

- **SELECT**
- **SET**

Statements MUST be separated by semicolons. There MUST be exactly one SELECT statement in every SharePoint Search SQL query, and it MUST be the last statement. The query MUST NOT be longer than an implementation-defined number of Unicode characters. This length limit MUST be greater than 1024 characters.<18>

This is the top-level element of the grammar:

```
query      =      *ws *(set-statement *ws ";" *ws) select-statement *ws
```

2.2.12.3 SELECT Statement

The **SELECT** statement specifies which properties to return in the search results, and how to select the search results from the crawled items. The syntax is as follows:

```
SELECT properties FROM SCOPE() WHERE conditions ORDER BY sortproperties
```

Both the WHERE and the ORDER BY clauses are optional. In response to a SELECT statement, the protocol server MUST return search results according to the order specified in the ORDER BY clause, or an arbitrary group of results if no sort order is specified.

```
select-statement      =      *ws "SELECT" 1*ws properties 1*ws "FROM" 1*ws "SCOPE" *ws "("
*ws ")" *ws ["WHERE" 1*ws conditions 1*ws] ["ORDER BY" 1*ws sortproperties *ws]
```

The properties component of the SELECT statement specifies which properties to return in the search results. It is simply a comma-separated list of property names that are returned by the protocol server as part of the search results. It MUST have the same effect as specifying the properties in the Query.Properties element of the QueryPacket when the value of the type attribute of the QueryText

element is "STRING". All of the properties specified here MUST be retrievable. If any of the specified properties are not retrievable, the search query MUST fail and return an ERROR_SERVER status code.

```
properties      =      identifier *(*ws "," *ws properties)
```

2.2.12.3.1 Conditions in the SELECT Statement

The conditions component of the SELECT statement specify which items to include in the search results. In detail, it specifies a Boolean expression that is evaluated for every item. If the expression is true, the protocol server MUST include the item in the list of search results. If the expression is false, the protocol server MUST NOT include the item in the list of search results. The protocol server MUST correctly interpret Boolean expressions that consist of other Boolean expressions joined by the operators AND and OR. The protocol server MUST allow an expression to be negated by using the NOT operator. There is an ordering of priority in which those Boolean operators are evaluated. The NOT operator has the highest priority, followed by the AND, and then the OR operators. It is possible to override evaluation priority by enclosing an expression that needs to be evaluated first in parenthesis. Finally, the protocol server MUST interpret expressions that consist of one of the Boolean expressions detailed following.

The protocol server MUST interpret a set of group aliases specified before the Boolean expression. These group aliases give the protocol client the capability to refer to a group of properties by a single identifier in the Boolean expression. The details of how to do this are specified in section 2.2.12.3.2.

```
conditions      =      [group-aliases 1*ws] predicate
predicate       =      predicate 1*ws "AND" 1*ws predicate
predicate       =/     predicate 1*ws "OR" 1*ws predicate
predicate       =/     "NOT" 1*ws predicate
predicate       =/     "(" *ws predicate *ws ")"
```

An implementation of the SharePoint Search SQL Syntax MUST support the following two types of predicates:

- full-text predicates
- non full-text predicates

It MUST support the following two types of full-text predicates:

- CONTAINS predicate
- FREETEXT predicate

It MUST also support the following four non full-text predicates:

- LIKE predicate
- Multi-value predicate
- Literal predicate
- NULL predicate

```
predicate       =/     ft-predicate / non-ft-predicate
ft-predicate      =      contains-predicate / freetext-predicate
non-ft-predicate      =      like-predicate / literal-predicate / multi-value-predicate /
null-predicate
```

2.2.12.3.1.1 CONTAINS Predicate

The CONTAINS predicate, in its most basic form, MUST be true if a specified full-text searchable property contains a given token, and MUST be false otherwise. In particular, it MUST be false if the given property is not a full-text searchable property. It MUST support a number of special features, such as searching over all properties instead of only one, prefix matching, matching of inflectional forms of tokens, matching of a token in proximity to another token and matching of administrator-defined thesaurus terms. It MUST also support Boolean combinations of these features.

The basic format of the CONTAINS predicate is as follows:

```
…CONTAINS(contains_property, contains_condition, language)…
```

The contains_property component MUST be one of the following:

- The **property identifier**.
- "*".
- "ALL".

Both "*" and "ALL" represent all full-text searchable properties that are crawled for a given item. If the property specified is not a text property, the results of the search query are undefined. If the contains_property component is not specified, the protocol server MUST use a default of Content.

The optional "language" component specifies the language that the protocol server evaluates the contains_condition component in. If no language is specified, the protocol server MUST revert to a reasonable default, such as the default language of the underlying operating system.

```
contains-predicate      =      "CONTAINS" *ws "(" *ws [contains-property *ws "," *ws] "'"
contains-condition "'" *ws ["," *ws lcid *ws] ")"
contains-property      =      identifier / "*" / "ALL"
```

The contains_condition component MUST be either a single token, or an expression that specifies what to match in more detail.

```
contains-condition      =      contains-token / contains-expression
contains-token      =      *ws 1*alphanum *ws
```

2.2.12.3.1.1.1 Conditions in the CONTAINS Predicate

If the condition in the CONTAINS predicate is actually an expression, the usual rules of Boolean expressions apply. The protocol server MUST interpret expressions that consist of other expressions joined with an AND or an OR operator as well as expressions that are negated with the NOT operator. There is an order of evaluation of these operators, with the NOT operator having the highest priority, followed by the AND, and then OR operators. Enclosing an expression in parenthesis MUST override this ordering. In a deviation from standard Boolean logic, the NOT operator MUST only be used on the right side of an AND operator. Finally, the protocol server MUST interpret all of the following as expressions:

- A token or phrase.
- A prefix to be matched.
- A NEAR Expression.

- A FORMSOF Expression.
- An ISABOUT Expression.

The following list shows details about these expressions:

```
contains-expression      =      contains-expression 1*ws "OR" 1*ws contains-expression
contains-expression      =/      contains-expression 1*ws "AND" 1*ws ["NOT" 1*ws] contains-
expression
contains-expression      =/      "(" *ws contains-expression *ws ")"
contains-expression      =/      contains-phrase-expression / contains-prefix-expression /
contains-near-expression / contains-formsof-expression / contains-isabout-expression
```

2.2.12.3.1.1.1.1 Phrase Expression

The phrase expression of the CONTAINS predicate syntax matches the entire specified phrase to the specified property. The expression MUST only be true if the entire phrase is found in the property. The phrase is specified by enclosing the tokens that make up the phrase in double quotes. Quotes inside the phrase need to be encoded with two double quote characters. The only restriction on the phrase is that it MUST NOT end in an asterisk character.

```
contains-phrase-expression      =      %x22.22      ; %x22 is double quote
contains-phrase-expression      =/      %x22 *quoted-dquote-char (%x00-21 / %x22.22 / %x23-29
/ %x2B-) %x22
      ; %x22 is double quote, %x2A is asterisk
```

2.2.12.3.1.1.1.2 Prefix Expression

The prefix expression of the CONTAINS predicate syntax looks similar to the phrase expression described in section 2.2.12.3.1.1.1.1. It MUST be a list of tokens enclosed by double quotes, followed by an asterisk character. The expression MUST be true if the property specified in the CONTAINS predicate syntax contains words that begin with the same characters as the tokens given, in the same order, and immediately adjacent. For example, the expression "work hard *" matches occurrences of "working hard" in the property, but it does not match occurrences of "hardly working".

```
contains-prefix-expression      =      %x22 *quoted-dquote-char *ws "*" %x22
      ; %x22 is double quote
```

2.2.12.3.1.1.1.3 NEAR Expression

The NEAR expression of the CONTAINS predicate syntax matches two phrase expressions or prefix expressions that occur close to each other in the item. This expression MUST be true whenever both expressions are found in the property specified in the CONTAINS predicate syntax. The difference between the NEAR expression and specifying two expressions joined with an AND operator is in the way the value of the Rank property is calculated. If the two expressions are more than 50 tokens apart, the calculated value of the Rank property for this expression MUST be 0. If the two expressions are less than 50 tokens apart, the value of the Rank property for this expression MUST be greater than 0. The closer the expressions are, the higher the calculated value of the Rank property MUST be. It is up to the implementation to define exactly how the value of the Rank property is calculated beyond these limitations.

The syntax of the NEAR expression is as follows:

```
…<expression> NEAR <expression>…
```

or

```
<expression> ~ <expression>
contains-near-expression      =      (contains-phrase-expression / contains-prefix-
expression) 1*ws ("NEAR" / "~") 1*ws (contains-phrase-expression / contains-prefix-
expression)
```

2.2.12.3.1.1.1.4 FORMSOF Expression

The FORMSOF expression of the CONTAINS predicate syntax matches a token and its alternate forms. The implementation MUST support two different kinds of alternates. The INFLECTIONAL type matches a token and its inflectional forms. The THESAURUS type matches a token and the thesaurus terms that are associated with the token on the protocol server. The syntax is as follows:

```
…FORMSOF(<generation type>, <phrase>)…
```

The generation type MUST be one of the following:

- INFLECTIONAL
- THESAURUS

The expression MUST be true if the phrase, or an inflectional form thereof, is found in the property specified in the CONTAINS predicate syntax. For example, the following expression:

```
…FORMSOF(INFLECTIONAL, "hard work")…
matches occurrences of "hard working" in the item.
contains-formsof-expression      =      "FORMSOF" *ws "(" *ws contains-formsof-type *ws ","
*ws contains-phrase-expression *ws ")"
contains-formsof-type      =      "INFLECTIONAL" / "THESAURUS"
```

2.2.12.3.1.1.1.5 ISABOUT Expression

The ISABOUT expression of the CONTAINS predicate syntax matches properties against one or more phrase or prefix expressions. The syntax is as follows:

```
…ISABOUT(subexpression1, subexpression2, …) RANKMETHOD method…
```

The expression MUST be true if at least one of the sub-expressions of the ISABOUT expression is true. The difference between the ISABOUT expression and combining its sub-expressions with the OR operator is in the way the value of the Rank property for the items is calculated. The protocol server MUST allow the client to specify a method for calculating the value of the Rank property with the optional RANKMETHOD keyword. Possible methods for calculating value of the Rank property are:

- JACCARD COEFFICIENT
- DICE COEFFICIENT
- INNER PRODUCT
- MINIMUM
- MAXIMUM

If no method of calculating the value of the Rank property is specified, the protocol server MUST assume the JACCARD COEFFICIENT method. How these methods affect the value of the Rank property is an implementation detail. See [MSDN-RANKMETHOD] for more information.

The ISABOUT expression is deprecated and SHOULD NOT be used by the protocol client. However, for legacy support the protocol server MUST support it.

```
contains-isabout-expression      =      "ISABOUT" *ws "(" *ws contains-isabout-components *ws
")" ["RANKMETHOD" 1*ws rankmethod]
contains-isabout-components      =      contains-isabout-component [*ws "," *ws contains-
isabout-components]
contains-isabout-component      =      contains-phrase-expression / contains-prefix-
expression
rankmethod      =      "JACCARD" 1*ws "COEFFICIENT"
rankmethod      =/      "DICE" 1*ws "COEFFICIENT"
rankmethod      =/      "INNER" 1*ws "PRODUCT"
rankmethod      =/      "MINIMUM"
rankmethod      =/      "MAXIMUM"
```

2.2.12.3.1.2 FREETEXT Predicate

The FREETEXT predicate is the recommended predicate to find items that contain tokens and phrases and calculate the value of the Rank property of the matching items. Its basic syntax is as follows:

```
…FREETEXT(property, freetext_condition, language)
```

The predicate MUST be true if the given property matches the conditions specified in the freetext_condition component. The protocol server MUST allow the protocol client to specify a property to search over. The protocol server MUST allow the name of a property, or "*", or "ALL" for all full-text searchable properties, or "DEFAULTPROPERTIES". The client SHOULD set this to "DEFAULTPROPERTIES".

It is an implementation detail which properties the protocol server searches over if "DEFAULTPROPERTIES" is specified. If the property is not specified, the protocol server MUST behave as if "DEFAULTPROPERTIES" had been specified. If the specified property is not a full-text searchable property, the predicate MUST evaluate to false. The optional language component specifies the language that the protocol server evaluates the freetext_condition component in. If evaluation in the specified language is not possible the protocol server SHOULD use any closely matching language, the system default, or a language neutral evaluation. If no language is given, the protocol server SHOULD revert to a reasonable default, such as the default language of the underlying operating system.

```
freetext-predicate      =      "FREETEXT" *ws "(" *ws [freetext-property *ws "," *ws]
freetext-condition *ws ["," *ws lcid *ws] ")"
freetext-property      =      identifier / alias-name / "DEFAULTPROPERTIES" / "ALL" / "*"
freetext-condition      =      string
```

2.2.12.3.1.3 LIKE Predicate

The LIKE predicate MUST be true if the specified pattern matches in the property. The basic syntax is as follows:

```
…property LIKE pattern…
```

The pattern is a normal string, as defined in section 2.2.12.1. It MUST be interpreted like any other string, with the following exceptions:

The "%" character MUST match zero or more of any other character.

The "_" character MUST match any single character.

Characters enclosed by square brackets (for example "[abc]") MUST match a single occurrence of either of the characters within the square brackets. This matching MUST be case-sensitive. The square brackets can contain a range in the form "[a-z]". This syntax MUST match one of any character in the range. If the first character in the square brackets is a caret ("^"), the matching MUST reverse, that is, any character that would have matched without the caret MUST not match, and vice versa.

To specify a literal percent sign, the protocol client MUST enclose it in square brackets.

For example, the following predicate syntax example:

```
…title LIKE 'Com%'…
```

matches all items the title of which starts with the letters "Com".

The following predicate syntax example:

```
title LIKE 'Co[m-p]%'
```

matches items the title of which starts with "Com", "Con", "Coo" or "Cop".

It is possible that implementation details prevent the protocol server from performing this matching on a given property. In that case, this predicate MUST evaluate to false.

```
like-predicate      =      identifier 1*ws "LIKE" 1*ws like-wildcard-literal
like-wildcard-literal      =      string
```

2.2.12.3.1.4 Literal Predicate

The literal predicate MUST be true if a given property stands in a given relation to a given literal. The basic syntax is as follows:

```
…property operator literal…
```

The operator MUST be one of the following:

Operator	Description
=	equal to
!= or <>	not equal to
>	greater than
>=	greater or equal to
<	less than
<=	less than or equal to

The literal is simply a value. The type of the value MUST match the type of the property. As a special case, properties that store dates can be compared to a time calculated from the current time. The function that does this is the DATEADD function.

The basic syntax for the DATEADD function is as follows:

```
DATEADD(unit, offset, time)
```

The time in the DATEADD function MUST be another DATEADD function or the GETGMTDATE function, which returns the current date and time. It MUST NOT be a date literal. The offset MUST be a negative integer that specifies how much time to add to the specified time. Because this integer MUST be negative, the DATEADD function actually performs a subtraction. Finally, the first parameter specifies the unit that the second parameter is measured in. Possible values are as follows:

- YEAR
- QUARTER
- MONTH
- WEEK
- DAY
- HOUR
- MINUTE
- SECOND

For example, the following predicate example:

```
…createdDate > DATEADD(YEAR, -1, GETGMTDATE())…
```

matches all items having a createdDate property lies less than a year in the past.

```
literal-predicate      =      identifier *ws operator *ws value
value      =      literal / dateadd-function
literal      =      boolean / hex / decimal / float / string / date
dateadd-function      =      "DATEADD" *ws "(" *ws dateadd-unit *ws "," *ws dateadd-offset
*ws "," *ws dateadd-datetime *ws ")"
dateadd-unit      =      "YEAR" / "QUARTER" / "MONTH" / "WEEK" / "DAY" / "HOUR" / "MINUTE" /
"SECOND"
dateadd-offset      =      negative-decimal / negative-hex
dateadd-datetime      =      dateadd-function / "GETGMTDATE()"
```

2.2.12.3.1.5 Multi-Value Predicate

The multi-value predicate allows comparisons between literals and properties that can have multiple values at the same time. The syntax MUST be of the following form:

```
…property operator quantifier ARRAY[literal_1, literal_2, …]…
```

The property component of the syntax of the multi-value predicate MUST be a multivalue property that will be compared against the specified literals.

The operator MUST be one of the operators specified in the table in section 2.2.12.3.1.4.

The quantifier MUST either be absent, or be one of the following:

- ANY
- SOME

The array of literals specifies the array that the multivalue property MUST be compared against. The specified array MUST consist of literals that are all of the same type. The value of the predicate depends on the operator and the quantifier.

If no quantifier is specified, the protocol server MUST compare each element of the property to the corresponding element in the array of literals, according to the operator. The predicate MUST be true if all comparisons are true, and the number of elements in the array is equal to the number of elements in the property. Otherwise, the predicate MUST be false.

The SOME quantifier and the ANY quantifier MUST work the same way. Both MUST compare each element in the property with each element in the array according to the operator. If any of these comparisons is true, the predicate MUST be true. Otherwise, the predicate MUST be false. The number of elements in either the array or the property has no influence on the value of the predicate.

```
multi-value-predicate      =      identifier 1*ws operator 1*ws [("SOME" / "ANY") 1*ws] mv-
comparison-list
mv-comparison-list      =      "ARRAY" 1*ws "[" *ws literal-list *ws "]"
literal-list      =      literal [*ws "," *ws literal-list]
```

2.2.12.3.1.6 NULL Predicate

The NULL predicate MUST be true if the item doesn't have a value for the specified property. Its basic syntax is as follows:

```
…property IS NULL…
```

It has an alternate syntax that reverses its meaning. The alternate syntax is as follows:

```
…property IS NOT NULL…
null-predicate      =      identifier 1*ws "IS" 1*ws ["NOT" 1*ws] "NULL"
```

2.2.12.3.2 Group Aliases in the SELECT Statement

The SELECT statement MUST support grouping several properties under an alias. The protocol server MUST then allow the protocol client to use the alias instead of a property name in the FREETEXT and CONTAINS predicates in the rest of the SELECT statement. To do this, the SELECT statement MUST support the optional WITH clause, which MUST be of the following form:

```
…WITH (property_1, property_2, …) AS #name…
```

The property_n components MUST be identifiers for the properties that are to be grouped under the alias #name. All alias names MUST begin with a pound sign. For example, the following query example:

```
SELECT title, path FROM SCOPE() WHERE WITH (title, author) AS #x FREETEXT(#x, 'bill')
```

returns all items having **Title** or **Author** properties contain the word "bill".

```
group-aliases       =       "WITH" *ws "(" *ws property-list *ws ")" *ws "AS" 1*ws alias-name
property-list       =       identifier [*ws "," *ws property-list]
```

2.2.12.3.3 Specifying Sort Order in the SELECT Statement

The SELECT statement MUST support ordering the search result set by one or more properties. The sort order MUST be specified using an ORDER BY clause. The format of the ORDER BY clause is as follows:

```
…ORDER BY property_1 direction, property_2 direction, …
```

The protocol server MUST sort the search result set first by the first specified property, and then break ties using the next specified property, and so on. Optionally, a direction marker can specify which direction to sort in, that is, ascending or descending. The direction marker MUST be one of the following:

- ASC
- DESC

```
sortproperties      =       sortproperty [*ws "," *ws sortproperties]
sortproperty        =       identifier [1*ws sortdirection]
sortdirection       =       "ASC" / "DESC"
```

2.2.12.4 SET Statement

The SET statement is a legacy statement and SHOULD NOT be used by the protocol client. However, the protocol server MUST recognize it in two different syntaxes. One is as follows:

```
…SET RANKMETHOD method…
```

In this form, the default method for calculating the value of the Rank property for the ISABOUT expression MUST be set to the method given in the SET statement. The SET statement only sets the default; if the ISABOUT expression specifies a different method for calculating the value of the Rank property, it MUST take precedence over the one specified in the SET statement. If there are multiple SET statements, the rightmost one MUST take precedence. If there is no SET statement that specified a method, the JACCARD COEFFICIENT method MUST be used as a default.

```
set-statement       =       "SET" 1*ws "RANKMETHOD" 1*ws rankmethod
```

The form of the second syntax of the SET statement is as follows:

```
…SET PROPERTYNAME guid PROPID id AS name TYPE type…
```

It is a legacy statement that MUST have no effect on the processing of the rest of the search query, but it MUST be recognized and not be rejected by the protocol server as a malformed search query. The guid component of the syntax MUST be a GUID. The id component of the syntax MUST be a positive integer, either specified in decimal or in hexadecimal. The name component MUST be a valid identifier. The form of the optional TYPE clause is as follows:

```
…TYPE type…
```

where the type component is one of the following:

- DBTYPE_I2
- DBTYPE_I4
- DBTYPE_R4
- DBTYPE_R8
- DBTYPE_CY
- DBTYPE_DATE
- DBTYPE_BSTR
- DBTYPE_BOOL
- DBTYPE_STR
- DBTYPE_WSTR

```
set-statement      =/      "SET" 1*ws "PROPERTYNAME" 1*ws guid 1*"PROPID" 1*ws property-id
1*ws "AS" 1*ws identifier [1*ws type-clause]
guid      =      8hex-digit "-" 4hex-digit "-" 4hex-digit "-" 4hex-digit "-" 12hex-digit
property-id      =      positive-decimal / positive-hex
type-clause      =      "TYPE" 1*ws dbtype
dbtype      =      "DBTYPE_I2" / "DBTYPE_I4" / "DBTYPE_R4" / "DBTYPE_R8" / "DBTYPE_CY" /
"DBTYPE_DATE" / "DBTYPE_BSTR" / "DBTYPE_BOOL" / "DBTYPE_STR" / "DBTYPE_WSTR"
```

2.2.13 SharePoint Search SQL Syntax v2

SharePoint Search SQL Syntax v2<19> specifies a syntax for search queries that is similar to the language used by SQL server. The syntax restricts the search query to the task of obtaining a list of search results from the protocol server.

The following is an ABNF description of the SharePoint Search SQL syntax, which is used in the Query and QueryEx operations. ABNF is specified in [RFC5234]. Following section 2.4 in [RFC5234], this description assumes Unicode as its external encoding. As such, the terminal values of this grammar represent Unicode code points.

2.2.13.1 Common Definitions

The first definitions are a set of basic definitions that are used throughout the rest of the description of the SharePoint Search SQL syntax:

```
ws          =  %x09 / %x20 / %x0A / %x0D
                  ; tab, space, line feed, carriage return
lowalpha    =  %x61-7A
upalpha     =  %x41-5A
alpha       =  lowalpha / upalpha
digit       =  %x30-39
alphanum    =  alpha / digit
```

Identifiers specify the names of properties. Identifiers MUST be between 1 and 128 characters in length. There are two types of identifiers, regular identifiers and delimited identifiers. Regular

identifiers MUST be formed from a limited set of characters and begin with a letter whereas delimited identifiers can contain any Unicode character. Regular identifiers MUST NOT be one of the keywords in SharePoint Search SQL syntax. An identifier that is also a keyword in SharePoint Search SQL syntax MUST be expressed using a delimited identifier. Delimited identifiers MUST be surrounded by double quotes. To use a double quote as part of a delimited identifier, it MUST be encoded as two double quotes in succession. **Content** is a special identifier. It MUST refer to a property that represents the body of the crawled items. Finally, aliases are regular identifiers that begin with a pound sign.

```
identifier             =   regular-identifier / delimited-identifier
regular-identifier     =   alpha *127identifier-char
identifier-char        =   alphanum / "_"
delimited-identifier   =   %x22 1*128quoted-dquote-char %x22
                                    ; %x22 is double quote
quoted-dquote-char     =   %x00-21 / %x22.22 / %x23-%x10FFFF
                               ; %x22 is double quote
alias-name             =   "#" regular-identifier
```

An implementation of the SharePoint Search SQL syntax MUST support various kinds of literals. One of these is the string literal. While string literals MUST NOT be limited in their length, they are limited by the length of the overall SharePoint Search SQL query. They can contain any Unicode character. They MUST be enclosed in single quotes. To use a single quote as part of a string, it MUST be encoded as two single quotes in a succession.

```
string= "'" *quoted-quote-char "'"
quoted-quote-char=%x00-26 / "''" / %x28-%x10FFFF
    ; single quote is %x27
```

Numeric literals represent numbers, including positive and negative integers expressed in decimal or hexadecimal, as well as floating point values, including exponential notation using the character 'e'.

```
positive-decimal        =   1*digit
negative-decimal        =   "-" positive-decimal
decimal                 =   positive-decimal / negative-decimal
hex-digit               =   digit / %x41-46 / %x61-66; digits and a-f
positive-hex            =   "0x" 1*hex-digit
negative-hex            =   "-" positive-hex
hex                     =   positive-hex / negative-hex
positive-float          =   1*digit ["." *digit] / "." 1*digit
negative-float          =   "-" positive-float
positive-float-exp      =   positive-float ["e" 1*digit]
negative-float-exp      =   "-" positive-float-exp
float                   =   positive-float-exp / negative-float-exp
```

Boolean literals represent logical values and MUST be either "TRUE" or "FALSE":

```
boolean="TRUE" / "FALSE"
```

Date literals represent specific dates or times. They MUST be enclosed in single quotation marks, and they MUST be in the form year/month/day or year-month-day. The year MUST be specified as a four-digit value. Time values MUST be in the form hours:minutes:seconds. The protocol server MUST interpret concatenated date and time values as a timestamp value.

```
year                    =   4digit
month                   =   [digit] digit
day                     =   [digit] digit
hour                    =   [digit] digit
minute                  =   [digit] digit
```

```
second                  =   [digit] digit
date-date               =   year "/" month "/" day
date-date               =/  year "-" month "-" day
date-time               =   hour ":" minute ":" second
date-value              =   date-date [1*ws date-time]
date                    =   "'" date-value "'"
```

CONTAINS Predicate and FREETEXT Predicate allow specifying a language for a token or a phrase. In the SharePoint Search SQL syntax, languages MUST be specified by an LCID.

```
lcid                    =   decimal / hex
```

For Literal Predicate, an implementation of the SharePoint Search SQL syntax MUST support the following comparison operators:

```
operator                =   "=" / "!=" / "<>" / ">" / ">=" / "<" / "<="
```

2.2.13.2 Query

An implementation of the SharePoint Search SQL syntax MUST support two statements:

- SELECT Statement
- SET Statement

Statements MUST be separated by semicolons. There MUST be exactly one **SELECT** statement in every SharePoint Search SQL query, and it MUST be the last statement. The query MUST NOT be longer than an implementation-defined number of Unicode characters. This length limit MUST be greater than 1024 characters.<20>

This is the top-level element of the grammar:

```
query                   =   *ws *(set-statement *ws ";" *ws) select-statement *ws
```

2.2.13.3 SELECT Statement

The **SELECT** statement specifies which properties to return in the search results, and how to select the search results from the crawled items. The syntax is as follows:

```
SELECT properties FROM SCOPE() WHERE conditions ORDER BY sortproperties
```

Both the **WHERE** and the **ORDER BY** clauses are optional. In response to a **SELECT** statement, the protocol server MUST return search results according to the order specified in the **ORDER BY** clause, or an arbitrary group of results if no sort order is specified.

```
select-statement = *ws "SELECT" 1*ws properties 1*ws "FROM" 1*ws "SCOPE" *ws "(" *ws ")" *ws
["WHERE" 1*ws conditions 1*ws] ["ORDER BY" 1*ws sortproperties *ws]
```

The *properties* component of the **SELECT** statement specifies which properties to return in the search results. It is simply a comma-separated list of property names that are returned by the protocol server as part of the search results if values for these properties are available. It MUST have the same effect as specifying the properties in the **Query.Properties** element of the **QueryPacket** when the value of

the **type** attribute of the **QueryText** element is "STRING". All of the properties specified here MUST be retrievable properties. If any of the specified properties are not retrievable, the search query MUST fail and return an ERROR_SERVER status code.

```
properties = identifier *(*ws "," *ws properties)
```

2.2.13.3.1 Conditions in the SELECT Statement

The *conditions* component of the **SELECT** statement specify which items to include in the search results. In detail, it specifies a Boolean expression that is evaluated for every item. If the expression is true, the protocol server MUST include the item in the list of search results. If the expression is false, the protocol server MUST NOT include the item in the list of search results. The protocol server MUST correctly interpret Boolean expressions that consist of other Boolean expressions joined by the operators **AND** and **OR**. The protocol server MUST allow an expression to be negated by using the **NOT** operator. There is an ordering of priority in which those Boolean operators are evaluated. The **NOT** operator has the highest priority, followed by the **AND,** and then the **OR** operators. It is possible to override evaluation priority by enclosing an expression that needs to be evaluated first in parenthesis. Finally, the protocol server MUST interpret expressions that consist of one of the following Boolean expressions.

```
conditions              =   predicate
predicate               =   predicate 1*ws "AND" 1*ws predicate
predicate               =/  predicate 1*ws "OR" 1*ws predicate
predicate               =/  "NOT" 1*ws predicate
predicate               =/  "(" *ws predicate *ws ")"
```

An implementation of the SharePoint Search SQL syntax MUST support the following two types of predicates:

- full-text predicates
- non full-text predicates.

It MUST support the following two types of full-text predicates:

- CONTAINS predicate
- FREETEXT predicate

It MUST also support the following three non full-text predicates:

- LIKE predicate
- Literal predicate
- NULL predicate.

```
predicate               =/  ft-predicate / non-ft-predicate
ft-predicate            =   contains-predicate / freetext-predicate
non-ft-predicate        = like-predicate / literal-predicate / null-predicate
```

2.2.13.3.1.1 CONTAINS Predicate

The **CONTAINS** predicate, in its most basic form, MUST be true if a specified full-text searchable property contains a given token, and MUST be false otherwise. In particular, it MUST be false if the given property is not a full-text searchable property. It MUST support a number of special features, such as searching over all properties instead of only one, prefix matching, matching of inflectional

forms of tokens, matching of a token in proximity to another token, and matching of administrator-defined **thesaurus** terms. It MUST also support Boolean combinations of these features.

The basic format of the **CONTAINS** predicate is as follows:

```
…CONTAINS(contains_property, contains_condition, language)…
```

The *contains_property* component MUST be one of the following:

- The property identifier.
- "*".
- "ALL."

Both "*" and "ALL" represent all full-text searchable properties that are crawled for a given item. If the property specified is not a text property, the results of the search query are undefined. If the *contains_property* component is not specified, the protocol server MUST use a default of **Contents**.

The optional "language" component specifies the language in which the protocol server evaluates the *contains_condition* component. If no language is specified, the protocol server MUST revert to a reasonable default, such as the default language of the underlying operating system.

```
contains-predicate     =   "CONTAINS" *ws "(" *ws [contains-property *ws "," *ws] "'"
contains-condition "'" *ws ["," *ws lcid *ws] ")"
contains-property      =   identifier / "*" / "ALL"
```

The *contains_condition* component MUST be either a single token, or an expression that specifies what to match in more detail.

```
contains-condition     =   contains-token / contains-expression
contains-token         =   *ws 1*alphanum *ws
```

2.2.13.3.1.1.1 Conditions in the CONTAINS Predicate

If the condition in the **CONTAINS** predicate is actually an expression, the usual rules of Boolean expressions apply. The protocol server MUST interpret expressions that consist of other expressions joined with an **AND** or an **OR** operator as well as expressions that are negated with the **NOT** operator. There is an order of evaluation of these operators, with the **NOT** operator having the highest priority, followed by the **AND,** and then **OR** operators. Enclosing an expression in parenthesis MUST override this ordering. In a deviation from standard Boolean logic, the **NOT** operator MUST only be used on the right side of an **AND** operator. Finally, the protocol server MUST interpret all of the following as expressions:

- A token or phrase.
- A prefix to be matched.
- A NEAR Expression.
- A FORMSOF Expression.

The following list shows details about these expressions:

```
contains-expression    =   contains-expression 1*ws "OR" 1*ws contains-expression
contains-expression    =/  contains-expression 1*ws "AND" 1*ws ["NOT" 1*ws] contains-
expression
```

```
contains-expression    =/  "(" *ws contains-expression *ws ")"
contains-expression    =/  contains-token-expression / contains-phrase-expression / contains-
prefix-expression / contains-near-expression / contains-formsof-expression
```

2.2.13.3.1.1.1.1 Token Expression

The token expression of the **CONTAINS** predicate syntax matches a single token to the specified property. The expression MUST only be true if the token is found in the property. The token is specified as a simple string.

```
contains-token-expression =  *quoted-dquote-char
```

2.2.13.3.1.1.1.2 Phrase Expression

The phrase expression of the **CONTAINS** predicate syntax matches the entire specified phrase to the specified property. The expression MUST only be true if the entire phrase is found in the property. The phrase is specified by enclosing the tokens that make up the phrase in double quotes. Quotes inside the phrase need to be encoded with two double quote characters. The only restriction on the phrase is that it MUST NOT end in an asterisk character.

```
contains-phrase-expression =   %x22.22    ; %x22 is double quote
contains-phrase-expression =/  %x22 *quoted-dquote-char (%x00-21 / %x22.22 / %x23-29 / %x2B-)
%x22
                               ; %x22 is double quote, %x2A is asterisk
```

2.2.13.3.1.1.1.3 Prefix Expression

The prefix expression of the **CONTAINS** predicate syntax looks similar to the phrase expression described in the section 2.2.13.3.1.1.1.2. It MUST be a list of tokens enclosed by double quotes, followed by an asterisk character. The expression MUST be true if the property specified in the **CONTAINS** predicate syntax contains words that begin with the same characters as the tokens given, in the same order, and immediately adjacent. For example, the expression "work hard *" matches occurrences of "working hard" in the property, but it does not match occurrences of "hardly working".

```
contains-prefix-expression =   %x22 *quoted-dquote-char *ws "*" %x22
                               ; %x22 is double quote
```

2.2.13.3.1.1.1.4 NEAR Expression

The **NEAR** expression of the **CONTAINS** predicate syntax matches two phrase expressions or prefix expressions that occur close to each other in the item. This expression MUST be true whenever both expressions are found in the property specified in the **CONTAINS** predicate syntax.

The syntax of the **NEAR** expression is as follows:

```
…<expression> NEAR <expression>…
```

or

```
…<expression> ~ <expression>…

contains-near-expression    =  (contains-phrase-expression / contains-prefix-expression) 1*ws
("NEAR" / "~") 1*ws (contains-phrase-expression / contains-prefix-expression)
```

2.2.13.3.1.1.1.5 FORMSOF Expression

The **FORMSOF** expression of the **CONTAINS** predicate syntax matches a token and its alternate forms. The implementation MUST support two different kinds of alternates. The *INFLECTIONAL* type matches a token and its inflectional forms. The *THESAURUS* type matches a token and the thesaurus terms that are associated with the token on the protocol server. The syntax is as follows:

```
…FORMSOF(<generation type>, <phrase>)…
```

The generation type MUST be one of the following:

- INFLECTIONAL
- THESAURUS

The expression MUST be true if the phrase, or an inflectional form thereof, is found in the property specified in the **CONTAINS** predicate syntax. For example, the following expression:

```
…FORMSOF(INFLECTIONAL, "hard work")…
matches occurrences of "hard working" in the item.
contains-formsof-expression      =  "FORMSOF" *ws "(" *ws contains-formsof-type *ws "," *ws
contains-phrase-expression *ws ")"
contains-formsof-type               =  "INFLECTIONAL" / "THESAURUS"
```

2.2.13.3.1.2 FREETEXT Predicate

The **FREETEXT** predicate is the recommended predicate to find items that contain tokens and phrases and calculate the value of the **Rank** property of the matching items. Its basic syntax is as follows:

```
…FREETEXT(property, freetext_condition, language)
```

The predicate MUST be true if the given property matches the conditions specified in the *freetext_condition* component. The protocol server MUST allow the protocol client to specify a property to search over. The protocol server MUST allow the name of a property, or "*", or "ALL" for all full-text searchable properties, or "DEFAULTPROPERTIES". The client SHOULD set this to "DEFAULTPROPERTIES", but it can set this to any of the other values the protocol server allows.

It is an implementation detail which properties the protocol server searches over if "DEFAULTPROPERTIES" is specified. If the property is not specified, the protocol server MUST behave as if "DEFAULTPROPERTIES" had been specified. If the specified property is not a full-text searchable property, the predicate MUST evaluate to false. The optional *language* component specifies the language that the protocol server SHOULD evaluate the *freetext_condition* component in. If evaluation in the specified language is not possible the protocol server SHOULD use any closely matching language, the system default, or a language neutral evaluation. If no language is given, the protocol server SHOULD revert to a reasonable default, such as the default language of the underlying operating system.

```
freetext-predicate              =   "FREETEXT" *ws "(" *ws [freetext-property *ws "," *ws]
freetext-condition *ws ["," *ws lcid *ws] ")"
freetext-property               =   identifier / alias-name / "DEFAULTPROPERTIES" / "ALL" /
"*"
freetext-condition              =   text-expression
```

2.2.13.3.1.3 LIKE Predicate

The **LIKE** predicate MUST be true if the specified pattern matches in the property, and the property is a retrievable property. Otherwise, the predicate MUST be false. The basic syntax is as follows:

```
…property LIKE pattern…
```

The pattern is a normal string, as defined in section 2.2.13.1. It MUST be interpreted as any other string, with the following exceptions:

The "%" character MUST match zero or more of any other character.

The "_" character MUST match any single character.

Characters enclosed by square brackets (for example "[abc]") MUST match a single occurrence of either of the characters within the square brackets. This matching MUST be case-insensitive. The square brackets can contain a range in the form "[a-z]". The result of this syntax MUST match one of any character in the range. If the first character in the square brackets is a caret ("^"), the matching MUST reverse, that is, any character that would have matched without the caret MUST NOT match, and vice versa.

To specify a literal percent sign, the protocol client MUST enclose it in square brackets.

For example, the following predicate syntax example:

```
…title LIKE 'com%'…
```

matches all items whose title starts with the letters "com".

The following predicate syntax example:

```
title LIKE 'co[m-p]%'
```

matches items whose title starts with "com", "con", "coo" or "cop".

```
like-predicate                =   identifier 1*ws "LIKE" 1*ws like-wildcard-literal
like-wildcard-literal         =   string
```

2.2.13.3.1.4 Literal Predicate

The literal predicate MUST be true if a given property stands in a given relation to a given literal. The basic syntax is as follows:

```
…property operator literal…
```

The operator MUST be one of the following:

Operator	Description
=	equal to
!= or <>	not equal to
>	greater than
>=	greater or equal to

Operator	Description
<	less than
<=	less than or equal to

The literal is simply a value. The type of the value MUST match the type of the property. As a special case, properties that store dates can be compared to a time calculated from the current time. The function that does this is the **DATEADD** function.

The basic syntax for the **DATEADD** function is as follows:

```
DATEADD(unit, offset, time)
```

The time in the **DATEADD** function MUST be another **DATEADD** function or the **GETGMTDATE** function, which returns the current date and time. It MUST NOT be a date literal. The offset MUST be a negative integer that specifies how much time to add to the specified time. Because this integer MUST be negative, the **DATEADD** function actually performs a subtraction. Finally, the first parameter specifies the unit in which the second parameter is measured. Possible values are as follows:

- YEAR
- QUARTER
- MONTH
- WEEK
- DAY
- HOUR
- MINUTE
- SECOND

For example, the following predicate example:

```
…createdDate > DATEADD(YEAR, -1, GETGMTDATE())…
```

matches all items for which the createdDate property lies less than a year in the past.

```
literal-predicate        =   identifier *ws operator *ws value
value                    =   literal / dateadd-function
literal                  =   boolean / hex / decimal / float / string / date
dateadd-function         =   "DATEADD" *ws "(" *ws dateadd-unit *ws "," *ws dateadd-offset
*ws "," *ws dateadd-datetime *ws ")"
dateadd-unit             =   "YEAR" / "QUARTER" / "MONTH" / "WEEK" / "DAY" / "HOUR" /
"MINUTE" / "SECOND"
dateadd-offset           =   negative-decimal / negative-hex
dateadd-datetime         =   dateadd-function / "GETGMTDATE()"
```

2.2.13.3.1.5 NULL Predicate

The **NULL** predicate MUST be true if the item doesn't have a value for the specified property. Its basic syntax is as follows:

```
…property IS NULL…
```

It has an alternate syntax that reverses its meaning. The alternate syntax is as follows:

```
…property IS NOT NULL…
```

If this predicate is used with the properties Path, Rank, ContentClass or SiteName, the query MUST return with the ERROR_SERVER error code.

If this predicate is used with the WorkId property, the query MUST return with the ERROR_BAD_QUERY error code.

If this predicate is used with the properties HitHighlightedSummary or HitHighlightedProperties, it MUST be true regardless of the value of the specified property.

If this predicate is used with the PictureThumbnailURL property, it MUST be false regardless of the value of the specified property.

```
null-predicate          =   identifier 1*ws "IS" 1*ws ["NOT" 1*ws] "NULL"
```

2.2.13.3.2 Specifying Sort Order in the SELECT Statement

The Section NULL Predicate MUST support ordering the search result set by one or more properties. The sort order MUST be specified using an **ORDER BY** clause. The format of the **ORDER BY** clause is as follows:

```
…ORDER BY property_1 direction, property_2 direction, …
```

The protocol server MUST sort the search result set first by the first specified property, and then break ties using the next specified property, and so on. Optionally, a direction marker can specify which direction to sort in, that is, ascending or descending. The direction marker MUST be one of the following:

- ASC
- DESC

```
Sortproperties = sortproperty [*ws "," *ws sortproperties]
Sortproperty = identifier [1*ws sortdirection]
Sortdirection = "ASC" / "DESC"
```

2.2.13.4 SET Statement

The form of the SET statement is as follows:

```
…SET PROPERTYNAME guid PROPID id AS name TYPE type…
```

It is a legacy statement that MUST have no effect on the processing of the rest of the search query, but it MUST be recognized and not be rejected by the protocol server as a malformed search query. The *guid* component of the syntax MUST be a GUID. The *id* component of the syntax MUST be a positive integer, either specified in decimal or in hexadecimal. The *name* component MUST be a valid identifier. The optional **TYPE** clause MUST be in the following form:

…TYPE *type*…

where the *type* component is one of the following:

- DBTYPE_I2
- DBTYPE_I4
- DBTYPE_R4
- DBTYPE_R8
- DBTYPE_CY
- DBTYPE_DATE
- DBTYPE_BSTR
- DBTYPE_BOOL
- DBTYPE_STR
- DBTYPE_WSTR

```
set-statement =/ "SET" 1*ws "PROPERTYNAME" 1*ws guid 1*"PROPID" 1*ws property-id 1*ws "AS"
1*ws identifier [1*ws type-clause]
guid = 8hex-digit "-" 4hex-digit "-" 4hex-digit "-" 4hex-digit "-" 12hex-digit
property-id = positive-decimal / positive-hex
type-clause = "TYPE" 1*ws dbtype
dbtype = "DBTYPE_I2" / "DBTYPE_I4" / "DBTYPE_R4" / "DBTYPE_R8" / "DBTYPE_CY" / "DBTYPE_DATE"
/ "DBTYPE_BSTR" / "DBTYPE_BOOL" / "DBTYPE_STR" / "DBTYPE_WSTR"
```

3 Protocol Details

In the following sections, the schema definition might differ from the processing rules imposed by the protocol. The WSDL in this specification matches the WSDL that shipped with the product and provides a base description of the schema. The text that introduces the WSDL might specify differences that reflect actual Microsoft product 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**, **not null**, and **present**.

The client side of this protocol is simply a pass-through. That is, no additional timers or other state is required on the client side of this protocol. Calls made by the higher-layer protocol or application are passed directly to the transport, and the results returned by the transport are passed directly back to the higher-layer protocol or application.

Except where specified, protocol clients SHOULD interpret HTTP Status Codes returned by the protocol server as specified in [RFC2616], Section 10.

This protocol allows protocol servers to notify protocol clients of application-level faults using SOAP faults. Except where specified, these SOAP faults are not significant for interoperability, and protocol clients can interpret them in an implementation-specific manner.

3.1 Server Details

3.1.1 Abstract Data Model

This section describes a conceptual model of possible data organization that an implementation maintains to participate in this protocol. The described organization is provided to facilitate the explanation of how the protocol behaves. This document does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this document.

3.1.1.1 Crawled Items and Properties

The protocol server maintains a list of crawled items. The protocol server also associates a list of properties with each crawled item. In addition to that, the protocol server computes some property values at query time. This is typically done for properties for which the values depend on the search query, such as the **Rank** or **HitHighlightedSummary** properties.

Properties are uniquely identified by their names. A property name is a unique, case-insensitive string. Every property has an associated type that defines the possible values it can take. For a list of possible types, see section 3.1.4.4.4.1. Properties come in different varieties. They can be retrievable property, full-text searchable property, or both.

The following table contains a set of default properties that every crawled item has:

Name	Type of the property value	Variety of the property	Description
CollapsingStatus	Int64	Retrievable, computed at query time	1 if duplicate result removal was performed as part of the request to the protocol server, and this search result had duplicates that were removed; 0 otherwise.

Name	Type of the property value	Variety of the property	Description
Contents	String	Full-text searchable, stored with the item	Contents of the item.
HitHighlightedProperties	String	Retrievable, computed at query time	String in the form: <HHTitle>*sample_title*</HHTitle><HHUrl>*sample_url*</HHUrl> *sample_title* is replaced by the actual title of the crawled item if title specified in the query request. *sample_url* is replaced by the actual URL of the crawled item. If a token in the URL or title is also present in the query text, this token can be enclosed by <c0> and </c0>.<21>
HitHighlightedSummary	String	Retrievable, computed at query time	A set of excerpts from the crawled item that are relevant to the search query. These excerpts are meant to be shown to the user to help the user decide whether the item is relevant to the user's search query or not. Excerpts are separated by a <ddd/> tag. The total length of all excerpts equates to a small paragraph of text. The exact length will vary depending on how the protocol client shows the **hit highlighted summary** in its user interface. If a token in the URL or title is also present in the query text, this token can be enclosed by <c0> and </c0>.<22>
Path	String	Retrievable, Full-text searchable, stored with the item	URL of the crawled item.
Rank	Int64	Retrievable, computed at query time	A value from 0 to 100000000, representing the **rank** of the crawled item among all search results, where the higher value means the higher relevance.<23>
Scope	String	Neither, stored with the item	The search scope in which the item exists. This property doesn't exist if **Query.ResultProvider** is "FASTSearch".<24>
Size	Int64	Retrievable, Full-text searchable, stored with the item	The approximate size of the crawled item in bytes.
Title	String	Retrievable, Full-text searchable, stored with the item	The title for the crawled item.

Name	Type of the property value	Variety of the property	Description
WorkId	Int64	Retrievable, Full-text searchable, stored with the item	The unique identifier for the crawled item among all the crawled items that the protocol server knows about. If **Query.ResultProvider** is "FASTSearch", the property MUST be empty.

The protocol also defines a set of properties that some, but not all crawled items in the search results have:

Name	Type of the property value	Variety of the property	Description
Author	String	Retrievable, Full-text searchable, stored with the item	The name of the author of the crawled item.
ContentClass	String	Retrievable, stored with the item	This property is present only when the crawled item is a **securable object**. It is one of the following strings: ▪ "STS_Site" ▪ "STS_Web" ▪ "STS_List" ▪ "STS_ListItem" ▪ "STS_Document" ▪ "STS_ListItem_DocumentLibrary" ▪ "STS_List_PictureLibrary" Respectively, they mean that the item is a site, a **subsite**, a **list**, a **list item**, an item stored in a site, a list item stored in a document library, and a list that stores pictures as its items.
Description	String	Retrievable, stored with the item	A short description of the crawled item.
IsDocument	Int64	Retrievable, stored with the item	If the crawled item is logically a container of another item, such as a **folder**, or if the crawled item is a list item, this property is 0. Otherwise, it is 1.

Name	Type of the property value	Variety of the property	Description
PictureThumbnailURL	String	Retrievable, stored with the item	URL of an image file that the protocol client can use to represent the item in a list of search results. The image file is a common image file type that popular web browsers can display, such as **JPEG**, **GIF** or **PNG**.
SiteName	String	Retrievable, stored with the item	URL of the site (2) that contains the crawled item. If the crawled item is not contained in a site, this property is absent.
Write	DateTime	Retrievable, stored with the item	Time and date that the crawled item was last changed.
Url	String	Retrievable, Full-text searchable, stored with the item	URL of the crawled item. This property doesn't exist if **Query.ResultProvider** is "SharepointSearch".
ServerRedirectedUrl	String	Retrievable, stored with the item	URL of preview of crawled item. This property doesn't exist if **Query.ResultProvider** is "SharepointSearch".
FileExtension	String	Retrievable, Full-text searchable, stored with the item	File extension of the crawled item. This property doesn't exist if **Query.ResultProvider** is "SharepointSearch".
SpSiteUrl	String	Retrievable, Full-text searchable, stored with the item	Root URL of the site. This property doesn't exist if **Query.ResultProvider** is "SharepointSearch".
docvector	String	Retrievable, stored with the item	Document vector of the crawled item. This property doesn't exist if **Query.ResultProvider** is "SharepointSearch".
fcocount	String	Retrievable, computed at query time	Number of duplicates for crawled item in the search results. This property doesn't exist **Query.ResultProvider** is "SharepointSearch".
fcoid	String	Retrievable, computed at query time	Value of field used for duplicate result removal. This property doesn't exist if **Query.ResultProvider** is "SharepointSearch".
PictureWidth	String	Retrievable, Full-text searchable, stored with the item	Width of an image file that the protocol client can use to represent the item in a list of search results. This property doesn't exist if **Query.ResultProvider** is "SharepointSearch".

Name	Type of the property value	Variety of the property	Description
PictureHeight	String	Retrievable, Full-text searchable, stored with the item	Height of an image file that the protocol client can use to represent the item in a list of search results. This property doesn't exist if **Query.ResultProvider** is "SharepointSearch".

3.1.1.2 High Confidence

The protocol server maintains a list of high confidence properties.

If the query text matches the value of one of the high confidence properties exactly, the associated item is considered a high confidence result for the search query.

3.1.1.3 Best Bets

The following diagram describes the abstract data model for best bets. In the diagram, each table specifies a type of entity in the model, and each arrow specifies that one type of entity always contains a reference to another.

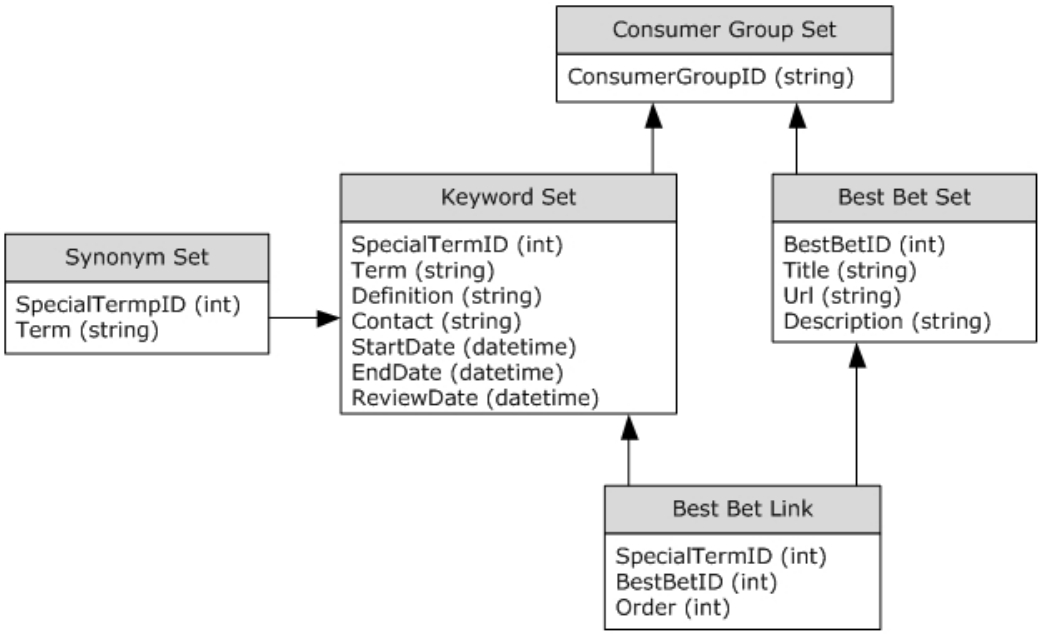

Figure 2: Best Bets and Keywords Abstract Data Model

The protocol server stores all implementation-specific information about best bets and keywords in the following sets of data:

Consumer Group Set: A collection of entries corresponding to keyword consumers. Each entry MUST be uniquely identified by its ConsumerGroupID and it MUST include the following elements:

- **ConsumerGroupID:** The unique identifier of the keyword consumer.

Keyword Set: A collection of entries representing keywords defined within a site collection. There is a many-to-many relationship between keywords and best bets. A keyword can have more than one best bet associated with it, and a best bet can be associated with multiple keywords. Each entry MUST be uniquely identified by its **SpecialTermId**, and it MUST include the following elements:

- **SpecialTermId**: The unique identifier of the keyword.
- **Term**: The term for the keyword.
- **Definition**: The definition of the keyword.
- **StartDate:** The date and time when the keyword begins to appear in search results.
- **Contact:** The contact name for the keyword.
- **EndDate:** The date and time when the keyword stops appearing in search result.
- **ReviewDate:** The date and time when the keyword is expected to be reviewed.

Best Bet Set: A collection of entries representing best bets defined within a site collection. Each entry MUST be uniquely identified by its **BestBetID**, and it MUST include the following elements:

- **BestBetID**: The unique identifier of the best bet.
- **Title**: The title of the best bet.
- **Url**: The URL for the best bet.
- **Description**: The description of the best bet.

Synonym Set: A collection of entries representing keyword synonyms associated with keywords. Each entry MUST include the following elements:

- **SpecialTermId**: The unique identifier of the keyword associated with the keyword synonym.
- **Term**: The term of the keyword synonym**.**

3.1.1.4 Visual Best Bets

The protocol server maintains visual best bets if the query role is FASTSearch.

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Message Processing Events and Sequencing Rules

The following table summarizes the list of **WSDL operations** defined by this specification.

Operation	Description
GetPortalSearchInfo<25>	Retrieves information about the protocol server's search scopes. This has been deprecated in favor of **GetSearchMetadata**.

Operation	Description
GetQuerySuggestions<26>	Retrieves a list of query suggestions.
GetSearchMetadata<27>	Retrieves information about the protocol server's properties and search scopes.
Query	Performs a search query.
QueryEx<28>	Performs a search query. It differs from the **Query** operation in the way it presents the output.
RecordClick<29>	MUST NOT be used.
Registration	Requests registration information from the protocol server.
Status	Requests the status of the protocol server.

3.1.4.1 GetPortalSearchInfo

This operation is used by the protocol client to retrieve information about the protocol server's search scopes. This operation has been deprecated in favor of GetSearchMetadata**.**

```
<wsdl:operation name="GetPortalSearchInfo">
  <wsdl:input message="tns:GetPortalSearchInfoSoapIn"/>
  <wsdl:output message="tns:GetPortalSearchInfoSoapOut"/>
</wsdl:operation>
```

The protocol client sends a GetPortalSearchInfoSoapIn request message and the protocol server responds with a GetPortalSearchInfoSoapOut response message, as follows:

- The protocol server returns a list of search scopes that have been defined by the **site collection administrator**.

3.1.4.1.1 Messages

The following table summarizes the set of WSDL message definitions that are specific to this operation.

Message	Description
GetPortalSearchInfoSoapIn	A request to initiate a **GetPortalSearchInfo** operation on the protocol server.
GetPortalSearchInfoSoapOut	A response from the protocol server at completion of the **GetPortalSearchInfo** operation.

3.1.4.1.1.1 GetPortalSearchInfoSoapIn

This message is sent by the protocol client to request information about the protocol server's search scopes.

The **SOAP action** value of the message is defined as:

```
http://microsoft.com/webservices/OfficeServer/QueryService/GetPortalSearchInfo
```

The **SOAP body** contains a **GetPortalSearchInfo** element.

3.1.4.1.1.2 GetPortalSearchInfoSoapOut

This message is used by the protocol server to respond to a GetPortalSearchInfoSoapIn message.

The SOAP body contains a GetPortalSearchInfoResponse element.

3.1.4.1.2 Elements

The following table summarizes the XML schema element definitions that are specific to this operation.

Element	Description
GetPortalSearchInfo	Body of the **GetPortalSearchInfoSoapIn** message.
GetPortalSearchInfoResponse	Body of the **GetPortalSearchInfoSoapOut** message.
SiteConfigInfo	Actual content of the protocol client's **GetPortalSearchInfoSoapOut** message.

3.1.4.1.2.1 GetPortalSearchInfo

This element is the body of the GetPortalSearchInfoSoapIn SOAP message.

```
<s:element name="GetPortalSearchInfo">
  <s:complexType/>
</s:element>
```

Because the **GetPortalSearchInfo** operation takes no arguments, the element is empty.

3.1.4.1.2.2 GetPortalSearchInfoResponse

This element is the body of the GetPortalSearchInfoSoapOut SOAP message.

```
<s:element name="GetPortalSearchInfoResponse">
  <s:complexType>
    <s:sequence>
      <s:element name="GetPortalSearchInfoResult" type="s:string" minOccurs="0"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

GetPortalSearchInfoResult: A **binary large object (BLOB)** of well-formed XML containing a **SiteConfigInfo** element as its **root element**, encoded as an XML string (see [XML10]).

3.1.4.1.2.3 SiteConfigInfo

This element is the actual content of the protocol client's GetPortalSearchInfoSoapOut SOAP message.

```
<s:element name="SiteConfigInfo">
```

```
  <s:complexType>
    <s:sequence>
      <s:element name="Name" type="s:string"/>
       <s:element name="Id" type="t:GUIDType"/>
      <s:element name="Scopes">
        <s:complexType>
          <s:sequence>
            <s:element name="Scope" type="s:string" minOccurs="0" maxOccurs="unbounded">
              <s:complexType>
                <s:sequence>
                  <s:element name="Name" type="s:string"/>
                </s:sequence>
              </s:complexType>
            </s:element>
          </s:sequence>
        </s:complexType>
      </s:element>
    </s:sequence>
  </s:complexType>
</s:element>
```

Name: Human-readable name of the protocol server.

Id: GUID of the protocol server.

Scopes: List of search scopes for the protocol server. This element MUST contain in its child elements a list of all the search scopes that the protocol server supports.

Scopes.Scope: Information about an individual search scope. It MUST occur once for each search scope.

Scopes.Scope.Name: Name of a search scope**.**

3.1.4.1.3 Complex Types

None.

3.1.4.1.4 Simple Types

None.

3.1.4.1.5 Attributes

None.

3.1.4.1.6 Groups

None.

3.1.4.1.7 Attribute Groups

None.

3.1.4.2 GetQuerySuggestions

This operation is used to retrieve a list of query suggestions to match the search terms typed in the search box. There are two types of query suggestions: pre-query suggestions and post-query suggestions.

```
<wsdl:operation name="GetQuerySuggestions">
```

```
    <wsdl:input message="tns:GetQuerySuggestionsSoapIn" />
    <wsdl:output message="tns:GetQuerySuggestionsSoapOut" />
</wsdl:operation>
```

The protocol client sends a GetQuerySuggestionsSoapIn request message and the protocol server responds with a GetQuerySuggestionsSoapOut response message, as follows:

- The protocol client formulates a search query for which **query** suggestions are requested.
- On receipt, the protocol server interprets the search query and assembles a list of query suggestions.
- On success, the protocol server returns a list of query suggestions to the client.
- On error, the protocol server returns a SOAP fault.

3.1.4.2.1 Messages

The following table summarizes the set of WSDL message definitions that are specific to this operation.

Message	Description
GetQuerySuggestionsSoapIn	A request to initiate a **GetQuerySuggestions** operation on the protocol server.
GetQuerySuggestionsSoapOut	A response from the protocol server at completion of the **GetQuerySuggestions** operation.

3.1.4.2.1.1 GetQuerySuggestionsSoapIn

This message is sent by the protocol client to request a list of query suggestions.

The SOAP action value of the message is defined as:

```
http://microsoft.com/webservices/OfficeServer/QueryService/GetQuerySuggestions
```

The SOAP body contains a GetQuerySuggestions element.

3.1.4.2.1.2 GetQuerySuggestionsSoapOut

This message is used by the protocol server to respond to a GetQuerySuggestionsSoapIn message.

The SOAP body contains a GetQuerySuggestionsResponse element.

3.1.4.2.2 Elements

The following table summarizes the XML schema element definitions that are specific to this operation.

Element	Description
GetQuerySuggestions	Body of the **GetQuerySuggestionsSoapIn** message.
GetQuerySuggestionsResponse	Body of the **GetQuerySuggestionsSoapOut** message.

3.1.4.2.2.1 GetQuerySuggestions

This element is the body of the GetQuerySuggestionsSoapIn SOAP message.

```
<s:element name="GetQuerySuggestions">
    <s:complexType>
         <s:sequence>
             <s:element name="queryXml" type="s:string" />
          </s:sequence>
    </s:complexType>
</s:element>
```

queryXml: A BLOB containing a QueryPacket element. This element MUST contain a BLOB of well-formed XML containing a **QueryPacket** element as its root element, encoded as an XML string (see [XML10]).

3.1.4.2.2.2 GetQuerySuggestionsResponse

This element is the body of the GetQuerySuggestionsSoapOut SOAP message.

```
<s:element name="GetQuerySuggestionsResponse">
    <s:complexType>
        <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="GetQuerySuggestionsResult"
type="tns:ArrayOfString" />
        </s:sequence>
    </s:complexType>
</s:element>
```

GetQuerySuggestionsResult: A list of query suggestions. The value MUST be of type **ArrayOfString**.

3.1.4.2.3 Complex Types

The following table summarizes the XML schema element definitions that are specific to this operation.

Complex type	Description
ArrayOfString	Defines a list of query suggestions.

3.1.4.2.3.1 ArrayOfString

This type defines a list of query suggestions.

```
<s:complexType name="ArrayOfString">
    <s:sequence>
        <s:element minOccurs="0" maxOccurs="unbounded" name="string" nillable="true"
type="s:string" />
    </s:sequence>
</s:complexType>
```

string: An individual query suggestion.

3.1.4.2.4 Simple Types

None.

3.1.4.2.5 Attributes

None.

3.1.4.2.6 Groups

None.

3.1.4.2.7 Attribute Groups

None.

3.1.4.3 GetSearchMetadata

This operation is used by the protocol client to retrieve information about properties and search scopes.

```
<wsdl:operation name="GetSearchMetadata">
    <wsdl:input message="tns:GetSearchMetadataSoapIn"/>
    <wsdl:output message="tns:GetSearchMetadataSoapOut"/>
</wsdl:operation>
```

The protocol client sends a GetSearchMetadataSoapIn request message and the protocol server responds with a GetSearchMetadataSoapOut response message, as follows:

The protocol server returns a list of search scopes and a list of properties.

3.1.4.3.1 Messages

The following table summarizes the set of WSDL message definitions that are specific to this operation.

Message	Description
GetSearchMetadataSoapIn	A request to initiate a **GetSearchMetadata** operation on the protocol server.
GetSearchMetadataSoapOut	A response from the protocol server at completion of the **GetSearchMetadata** operation.

3.1.4.3.1.1 GetSearchMetadataSoapIn

This message is sent by the protocol client to request information about the properties and search scopes available on protocol server.

The SOAP action value of the message is defined as:

```
http://microsoft.com/webservices/OfficeServer/QueryServiceGetSearchMetadata
```

The SOAP body contains a GetSearchMetadata element.

3.1.4.3.1.2 GetSearchMetadataSoapOut

This message is used by the protocol server to respond to a GetSearchMetadataSoapIn message.

The SOAP body contains a GetSearchMetadataResponse element.

3.1.4.3.2 Elements

The following table summarizes the XML schema element definitions that are specific to this operation.

Element	Description
GetSearchMetadata	Body of the **GetSearchMetadataSoapIn** message.
GetSearchMetadataResponse	Body of the **GetSearchMetadataSoapOut** message.

3.1.4.3.2.1 GetSearchMetadata

This element is the body of the GetSearchMetadataSoapIn SOAP message.

```
<s:element name="GetSearchMetadata">
  <s:complexType/>
</s:element>
```

Because the **GetSearchMetadata** operation takes no arguments, the element is empty.

3.1.4.3.2.2 GetSearchMetadataResponse

This element is the body of the GetSearchMetadataSoapOut SOAP message.

```
<s:element name="GetSearchMetadataResponse">
  <s:complexType>
    <s:sequence>
      <s:element name="GetSearchMetadataResult">
        <s:complexType>
          <s:sequence>
            <s:element ref="s:schema"/>
            <s:any/>
          </s:sequence>
        </s:complexType>
      </s:element>
    </s:sequence>
  </s:complexType>
</s:element>
```

GetSearchMetadataResult: Specifies information about the properties and search scopes that the protocol server supports. The information MUST be encoded in an ADO.NET **DataSet** object, which MUST BE encoded in the DiffGram format. This format is specified in [MS-DSDIFFGRAM]. The name of the **DataSet** object MUST be "SearchMetadata". The **DataSet** object MUST contain the Properties Table and the Scopes Table.<30>

3.1.4.3.2.2.1 The Properties Table

The **Properties** table MUST contain information about all properties that are available on the protocol server. It has the following columns:

Name	Type	Description
Name	String	Name of the property.
Description	String	Description of the property.
Type	String	Type of the property. Values MUST be one of the following: System.String System.Int64 System.Double System.DateTime System.Boolean System.Byte[]
Retrievable	Boolean	True if the property is a retrievable property; otherwise false.
FullTextQueryable	Boolean	True if the property is a full-text searchable property; otherwise false.

The **Properties** table MUST have one row per property available on the protocol server.

3.1.4.3.2.2.2 The FASTSearchProperties Table

The **FASTSearchProperties** table MUST contain information about all managed properties that are available on the protocol server.<31> It has the following columns:

Name	Type	Description
Name	String	Name of the property.
Description	String	Description of the property.
Type	String	Type of the property. Values MUST be one of the following: System.String System.Int64 System.Double System.DateTime System.Boolean System.Byte[]
Retrievable	Boolean	True if the property is a retrievable property; otherwise false.
FullTextQueryable	Boolean	True if the property is a full-text searchable property; otherwise false.
Sortable	Boolean	True if property can be used to sort search results ascending or descending; otherwise false.
Refinable	Boolean	True if property has a refiner; otherwise false.

The **FASTSearchProperties** table MUST have one row per property available on the protocol server.

3.1.4.3.2.2.3 The Scopes Table

The **Scopes** table MUST contain information about all search scopes that are available on the protocol server. It has the following columns:

Name	Type	Description
Name	String	Name of the search scope.
Description	String	Description of the search scope.

The **Scopes** table MUST have one row per search scope available on the protocol server.

3.1.4.3.3 Complex Types

None.

3.1.4.3.4 Simple Types

None.

3.1.4.3.5 Attributes

None.

3.1.4.3.6 Groups

None.

3.1.4.3.7 Attribute Groups

None.

3.1.4.4 Query

This operation is used to run a search query and retrieve a list of crawled items that are relevant to the search query, together with associated information.

```
<wsdl:operation name="Query">
  <wsdl:input message="tns:QuerySoapIn"/>
  <wsdl:output message="tns:QuerySoapOut"/>
</wsdl:operation>
```

The protocol client sends a **QuerySoapIn** message and the protocol server responds with a **QuerySoapOut** (section 3.1.4.4.1.2) message, as follows:

The protocol client formulates a search query that specifies which crawled items to retrieve and which properties to retrieve for each crawled item (see section 3.1.1.1).

On receipt, the protocol server interprets the search query and assembles a list of crawled items that are relevant to the search query. Then it retrieves the properties that were specified in the search query for each result, and includes them in the response to the search query.

Alternatively, the request can fail. In that case the protocol server returns a response containing an error code as defined in section 2.2.5.5.

3.1.4.4.1 Messages

The following table summarizes the set of WSDL message definitions that are specific to this operation.

Message	Description
QuerySoapIn	A request to initiate a **Query** operation on the protocol server.
QuerySoapOut	A response from the protocol server at completion of the **Query** operation.

3.1.4.4.1.1 QuerySoapIn

This message is used by the protocol client to request search results from the protocol server.

The SOAP action value of the message is defined as:

```
urn:Microsoft.Search/Query
```

The SOAP body contains a Query element.

3.1.4.4.1.2 QuerySoapOut

This message is used by the protocol server to respond to a QuerySoapIn message.

The SOAP body contains a QueryResponse element.

3.1.4.4.2 Elements

The following table summarizes the XML schema element definitions that are specific to this operation.

Element	Description
Document	Describes a crawled item.
Query	Body of the **QuerySoapIn** message.
QueryResponse	Body of the **QuerySoapOut** message.
ResponsePacket	Actual content of the protocol server's **QuerySoapOut** message.

3.1.4.4.2.1 Document

This element describes a crawled item as part of the search results.

```
<s:element name="Document">
  <s:complexType>
    <s:sequence>
      <s:element name="Title" type="s:string" minOccurs="0"/>
      <s:element name="Action">
        <s:complexType>
          <s:sequence>
            <s:element name="LinkUrl">
```

```
              <s:complexType>
                <s:simpleContent>
                  <s:extension base="s:string">
                    <s:attribute name="size" type="s:unsignedInt" use="optional"/>
                    <s:attribute name="fileExt" type="s:string" use="optional"/>
                  </s:extension>
                </s:simpleContent>
              </s:complexType>
            </s:element>
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:choice>
        <s:sequence>
          <s:element name="Description" type="s:string"/>
          <s:element name="Date" type="s:dateTime"/>
        </s:sequence>
        <s:element name="Properties">
          <s:complexType>
            <s:sequence>
              <s:element name="Property" minOccurs="0" maxOccurs="unbounded">
                <s:complexType>
                  <s:sequence>
                    <s:element name="Name" type="s:string"/>
                    <s:element name="Type" type="d:PropertyType"/>
                    <s:element name="Value" type="s:string"/>
                  </s:sequence>
                </s:complexType>
              </s:element>
            </s:sequence>
          </s:complexType>
        </s:element>
      </s:choice>
    </s:sequence>
    <s:attribute name="relevance" type="s:double"/>
  </s:complexType>
</s:element>
```

relevance: Relevance of the crawled item. A higher value means a higher relevance.

Title: Title of the crawled item. This element MUST only be present if the **QueryPacket** element of the request did not have a **Query.Properties** child element.

Action: Action to take when the crawled item is selected by the user from the list of search results.

Action.LinkUrl: URL of the crawled item.

Action.LinkUrl.size: Size of the crawled item. If present, it MUST contain the size of the crawled item in bytes. If the **QueryPacket** element of the request contains a **Query.Properties** child element, this attribute MUST be absent; otherwise this attribute MUST be present.

Action.LinkUrl.fileExt: The **file extension** of the crawled item. This attribute MUST be present if the protocol server is able to determine a meaningful file extension for the crawled item; the attribute MUST NOT be present otherwise.

Description: The description of the crawled item. Some crawled items don't have a description available. In those cases, the content of this element can be empty. This element MUST only be present if the **QueryPacket** element of the request does not have a **Query.Properties** child element.

Date: The date the crawled item was last modified. This element MUST only be present if the **QueryPacket** element of the request does not have a **Query.Properties** child element.

Properties: The property values for the crawled item. This element MUST only be present if the request's **QueryPacket** element contained a **Query.Properties** child element. It MUST only contain

the property values for the properties which were specified in the request's **QueryPacket.Query.Properties** element. If **QueryPacket.Query.Properties** has no child elements, the protocol server will return the default properties as specified in section 2.2.3.1.

Properties.Property: The information for a property of the crawled item. Note that not all crawled items have non-empty values for every property. If an item has an empty value in a requested property for an item in the search results, the protocol server SHOULD NOT generate a corresponding **Properties.Property** element.

Properties.Property.Name: The name of the property.

Properties.Property.Type: The type of the property. See section 3.1.4.4.4.1 for possible values<32>.

Properties.Property.Value: The property value for this crawled item. The values MUST be encoded according to their type. Refer to section 3.1.4.4.4.1 for details about the encoding.

3.1.4.4.2.2 Query

This element is the body of the QuerySoapIn SOAP message.

```
<s:element name="Query">
  <s:complexType>
    <s:sequence>
      <s:element name="queryXml" type="s:string"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

queryXml: A BLOB containing a **QueryPacket** (section 2.2.3.1) element. This element MUST contain a BLOB of well-formed XML containing a **QueryPacket** element as its root element, encoded as an XML string, (see [XML10]).

3.1.4.4.2.3 QueryResponse

This element is the body of the **QuerySoapOut** (section 3.1.4.4.1.2) SOAP message.

```
<s:element name="QueryResponse">
  <s:complexType>
    <s:sequence>
      <s:element name="QueryResult" type="s:string"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

QueryResult: BLOB of well-formed XML containing a **ResponsePacket** element as its root element, encoded as an XML string (see [XML10]).

3.1.4.4.2.4 ResponsePacket

This element is the actual content of the protocol server's **QuerySoapOut** (section 3.1.4.4.1.2) SOAP message. It contains the search results that the protocol server created in response to the search query issued by the **QuerySoapIn** SOAP message.

```
<s:element name="ResponsePacket">
  <s:complexType>
    <s:sequence>
      <s:element name="Response">
        <s:complexType>
```

```
            <s:sequence>
              <s:element name="QueryId" type="t:GUIDType" minOccurs="0"/>
              <s:element name="Range" minOccurs="0">
                <s:complexType>
                  <s:sequence>
                    <s:element name="StartAt" type="t:StartAtType"/>
                    <s:element name="Count" type="s:unsignedInt"/>
                    <s:element name="TotalAvailable" type="s:unsignedInt"/>
                    <s:element name="Results" minOccurs="0">
                      <s:complexType>
                        <s:sequence>
                          <s:element ref="d:Document" maxOccurs="unbounded"/>
                        </s:sequence>
                      </s:complexType>
                    </s:element>
                  </s:sequence>
                </s:complexType>
              </s:element>
              <s:element name="Status" type="r:StatusType"/>
              <s:element name="DebugErrorMessage" type="t:String2048" minOccurs="0"/>
            </s:sequence>
            <s:attribute name="domain" type="t:String255" use="optional"/>
          </s:complexType>
        </s:element>
      </s:sequence>
    </s:complexType>
  </s:element>
```

Response: This element contains the response to the query.

Response.QueryId: The unique identifier of the search query (see section 2.2.5.2). It MUST only be present if the request's **QueryPacket.Query** element contained a **QueryId** child element. In that case, this element MUST be set to the same value that was set in the request's **QueryPacket.Query.QueryId** element**.**

Response.Range: Information about the range of search results returned, in addition to search results. For each query request, the protocol server MUST behave as if it generated a large list of all results, ordered as specified in the request's **QueryPacket.Query.SortByProperties** element. The results that are returned in the response to a search query MUST behave as if they were a subset from that list as specified in the request's **QueryPacket.Query.Range** element. If the requested count of search results exceeds the number of items in the search results, the protocol server MUST return all search results. In that case, this element MUST contain the information about the range of search results that were actually returned (as opposed to the range that was requested). This element MUST be present if the status code in the **Response.Status** element is "SUCCESS". This element MUST NOT be present otherwise.

Response.Range.StartAt: Where to begin the returned results (see section 2.2.5.4). This element MUST specify which element in the big list of all results corresponds to the first element returned by the protocol server for the query request. For example, if this is set to 10, the first result returned is the tenth result in the list of all results.

Response.Range.Count: Specifies how many items are being returned. It MUST be set to the number of results that follow.

Response.Range.TotalAvailable: Approximate number of results for this query. This element SHOULD be set to the approximate number of results that are available for the given search query.

Response.Range.Results: Result items for this query. This element MUST contain all the actual results in its child elements. The child elements MUST be of type Document.

Response.Status: Status of the query. This element MUST contain a status code that indicates the success of the request. See section 2.2.5.5 for possible values.

Response.DebugErrorMessage: If the value of the **Response.Status** element is anything but "SUCCESS", this element MUST either be empty or contain an error message (see section 2.2.5.7). This error message is intended to be read by an administrator. If the value of the **Response.Status** element is "SUCCESS", this element MUST NOT be present.

Response.domain: If the request's **QueryPacket.Query** element had a **domain** attribute, this attribute MUST be set to the same value that was present in the **QueryPacket.Query.domain** attribute of the request (see section 2.2.5.8). If the request's **QueryPacket.Query** element did not have a **domain** attribute, the presence of this attribute depends on whether the status code in the **Response.Status** element is "SUCCESS" or not. If the status code in the **Response.Status** element is "SUCCESS", and the request's **QueryPacket.Query** element did not have a **domain** attribute, this attribute MUST be absent. If the status code in the **Response.Status** element is anything other than "SUCCESS", and the request's **QueryPacket.Query** element did not have a **domain** attribute, this attribute MUST be present and set to an empty string.

3.1.4.4.3 Complex Types

None.

3.1.4.4.4 Simple Types

The following table summarizes the XML schema simple type definitions that are specific to this operation.

Simple type	Description
PropertyType	Defines the type of property.

3.1.4.4.4.1 PropertyType

This type defines the type of a property. The type of the property imposes restrictions upon the possible values of the property as outlined here:

```
<s:simpleType name="PropertyType">
  <s:restriction base="s:string">
    <s:enumeration value="Boolean"/>
    <s:enumeration value="Byte"/>
    <s:enumeration value="Char"/>
    <s:enumeration value="DateTime"/>
    <s:enumeration value="Double"/>
    <s:enumeration value="Int16"/>
    <s:enumeration value="Int32"/>
    <s:enumeration value="Int64"/>
    <s:enumeration value="Single"/>
    <s:enumeration value="String"/>
    <s:enumeration value="UInt16"/>
    <s:enumeration value="UInt32"/>
    <s:enumeration value="UInt64"/>
  </s:restriction>
</s:simpleType>
```

The following table defines the allowable values for **PropertyType**:

Value	Meaning
Boolean	The value MUST be either "true" or "false".

Value	Meaning
Byte	The value MUST be a positive integer in the range of 0 to 255.
Char	The value MUST be a single Unicode character.
DateTime	The value MUST be a valid description of a time in the format described in [RFC3339].
Double	The value MUST be a valid double-precision floating point number as defined in [IEEE754]. The number MUST be expressed in the format ".X", "X.X", ".XEX" or "X.XEX", where each X can stand for 1+ instances of 0-9. The values of NaN or Infinity are not supported.
Int16	The value MUST be an integer in the range from -32768 to 32767.
Int32	The value MUST be an integer in the range from -2147483648 to 2147483647.
Int64	The value MUST be an integer in the range from -9223372036854775808 to 9223372036854775807.
Single	The value MUST be a valid single-precision floating point number as defined in [IEEE754]. The number MUST be expressed in the format ".X", "X.X", ".XEX" or "X.XEX", where each X can stand for 1+ instances of 0-9. The values of NaN or Infinity are not supported.
String	The value MUST be any valid Unicode string. If the implementation of the protocol server imposes a limit on the length of a string property, the limit MUST be at least 2048 Unicode characters.
UInt16	The value MUST be a positive integer in the range of 0 to 65535.
UInt32	The value MUST be a positive integer in the range of 0 to 4294967295.
UInt64	The value MUST be a positive integer in the range of 0 to 18446744073709551615.

3.1.4.4.5 Attributes

None.

3.1.4.4.6 Groups

None.

3.1.4.4.7 Attribute Groups

None.

3.1.4.5 QueryEx

This operation is used to run a search query and retrieve a list of links to crawled items that are relevant to the search query, together with associated information.

```
<wsdl:operation name="QueryEx">
  <wsdl:input message="tns:QueryExSoapIn"/>
  <wsdl:output message="tns:QueryExSoapOut"/>
</wsdl:operation>
```

The protocol client sends a QueryExSoapIn request message and the protocol server responds with a QueryExSoapOut response message, as follows:

The client formulates a search query that specifies which crawled items to retrieve and which properties to retrieve for each crawled item (see section 3.1.1.1).

On receipt, the protocol server interprets the search query and assembles a list of crawled items that are relevant to the search query. Then, it retrieves the properties that were specified in the search query for each result, and includes them in the response to the search query.

Alternatively, the request can fail. In that case the protocol server returns a response containing an error code as defined in section 2.2.5.5.

3.1.4.5.1 Messages

The following table summarizes the set of WSDL message definitions that are specific to this operation.

Message	Description
QueryExSoapIn	A request to initiate a **QueryEx** operation on the protocol server.
QueryExSoapOut	A response from the protocol server at completion of the **QueryEx** operation.

3.1.4.5.1.1 QueryExSoapIn

This message is used by the protocol client to request search results from the protocol server.

The SOAP action value of the message is defined as:

```
http://microsoft.com/webservices/OfficeServer/QueryService/QueryEx
```

The SOAP body contains a QueryEx element.

3.1.4.5.1.2 QueryExSoapOut

This message is used by the protocol server to respond to a QueryExSoapIn message.

The SOAP body contains a QueryExResponse element.

3.1.4.5.2 Elements

The following table summarizes the XML schema element definitions that are specific to this operation.

Element	Description
QueryEx	Body of the **QueryExSoapIn** message.
QueryExResponse	Body of the **QueryExSoapOut** message.

3.1.4.5.2.1 QueryEx

This element is the body of the QueryExSoapIn SOAP message.

```
<s:element name="QueryEx">
  <s:complexType>
    <s:sequence>
      <s:element name="queryXml" type="s:string"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

queryXml: A **BLOB** containing a QueryPacket element. This element MUST contain a **BLOB** of well-formed XML containing a **QueryPacket** element as its root element, encoded as an XML string (see [XML10]).

3.1.4.5.2.2 QueryExResponse

This element is the body of the QueryExSoapOut SOAP message.

```
<s:element name="QueryExResponse">
  <s:complexType>
    <s:sequence>
      <s:element name="QueryExResult" minOccurs="0">
        <s:complexType>
          <s:sequence>
            <s:element ref="s:schema"/>
            <s:any/>
          </s:sequence>
        </s:complexType>
      </s:element>
    </s:sequence>
  </s:complexType>
</s:element>
```

QueryExResult: Contains the search results for the search query. The information MUST be encoded in an ADO.NET **DataSet** object, which MUST be encoded in the DiffGram format. This format is specified in [MS-DSDIFFGRAM]. The name of the **DataSet** object MUST be "Results".

The **DataSet** object MUST contain the following extended properties:

Name	Description
QueryTerms	The extracted tokens from the search query, separated by semicolons, in the order that they occur, including tokens generated by the protocol server in the case the request's **Query.EnableStemming** element, or **Query.EnableNicknames** element, or **Query.EnablePhonetic** element were set to "true" as specified in section 2.2.3.1. For a search query of type "MSSQLFT", this includes all tokens that are used in the CONTAINS and FREETEXT predicates. For a search query of type "STRING", this includes all tokens used in the query text. If **Query.ResultProvider** is "FASTSearch", the protocol server MUST return the original query text without semicolons between terms.
IgnoredNoiseWords	The noise words that were found in the search query, separated by semicolons, in any order. For a search query of type "MSSQLFT", this includes all noise words that are used in the CONTAINS and FREETEXT predicates. For a search query of type "STRING", this includes all noise words used in the query text. If **Query.ResultProvider** is "FASTSearch", the protocol server MUST return an empty value.

Name	Description
SpellingSuggestion	The spelling suggestion for the search query. The protocol server suggests a different spelling of the search query if it detects a good chance that the spelling suggestion will increase the quality of the search results. How this chance is calculated is an implementation detail. If there is no good chance that a spelling suggestion would increase the quality of the search results, the protocol server MUST leave this extended property empty.
ElapsedTime	The time, in milliseconds, it took to run the search query.
Keyword	The token that is defined in the **Definition** extended property. The protocol server can store a list of definitions for certain tokens. If such a token is found in the query text, this extended property is set to the token that a definition was found for. If no definition could be found for any of the tokens in the query text, this extended property MUST NOT be set.
Definition	The definition of the token in the **Keyword** extended property. If the **Keyword** extended property is empty, the value of this extended property MUST be empty as well.
QueryModification	The query modification for the search query. If original query returns zero results, the protocol server modifies the query according to **Query.ResubmitFlags**, if set on the request, and re-evaluates the query. If query was modified and re-evaluated before returning search results, the protocol server MUST return the modified query in this property. If **ResultProvider** is "SharepointSearch", the protocol server MUST not return this property.

For a description of extended properties in general, see [MS-DSDIFFGRAM] section 2.3.1.

Depending on the formulation of the search query, there can be up to three tables in the **DataSet** object, one for the relevant results, one for best bets and one for high confidence results. All of these tables MUST contain two extended properties, **TotalRows** and **IsTotalRowsExact,** as specified in [MS-QSSWS] section 3.1.4.1.3.6. The former MUST contain the total number of crawled items, best bets or high confidence results that match the conditions given in the search query, or an approximation thereof. The latter MUST be "True" if the number given in **TotalRows** is an exact number, rather than an approximation, and "False" otherwise.

3.1.4.5.2.2.1 The RelevantResults Table

The **RelevantResults** table contains the actual search results. It MUST only be present if the **IncludeRelevantResults** element in the QueryPacket element of the QueryEx message was either absent or set to true. If this table is present, it MUST be the first table in the DataSet object. The table MUST have one column per requested property. It MUST have one row per search result. In any given row, the values in the columns MUST represent the value that the corresponding property has for that item.

3.1.4.5.2.2.2 The SpecialTermResults Table

The **SpecialTermResults** table contains best bets that apply to the search query. It MUST only be present if the **IncludeSpecialTermResults** element in the QueryPacket element of the QueryEx operation was present and set to true. It MUST have one row per best bet, and it MUST have the columns *Title*, *Url* and *Description*. The columns MUST contain, respectively, the title of the best bet, the URL that the best bet links to, and a human-readable description of the best bet.

3.1.4.5.2.2.3 The HighConfidenceResults Table

The **HighConfidenceResults** table contains high confidence results that apply to the search query. It MUST only be present if the **IncludeHighConfidenceResults** element in the QueryPacket element of

the QueryEx message was present and set to true. It MUST have one row per high confidence result, and it MUST have the following columns:

- Title
- Url
- Description
- HighConfidenceImageURL
- HighConfidenceDisplayProperty1
- HighConfidenceDisplayProperty2
- HighConfidenceDisplayProperty3
- HighConfidenceDisplayProperty4
- HighConfidenceDisplayProperty5
- HighConfidenceDisplayProperty6
- HighConfidenceDisplayProperty7
- HighConfidenceDisplayProperty8
- HighConfidenceDisplayProperty9
- HighConfidenceDisplayProperty10
- HighConfidenceDisplayProperty11
- HighConfidenceDisplayProperty12
- HighConfidenceDisplayProperty13
- HighConfidenceDisplayProperty14
- HighConfidenceDisplayProperty15
- HighConfidenceType

All of these columns are of **string** type. The Title column MUST contain the title of the high confidence result. This column MUST NOT be empty. The URL column MUST contain a valid URL that points to the high confidence result. This column MUST NOT be empty. The Description column SHOULD contain a description of the high confidence result. The HighConfidenceImageUrl column SHOULD contain the URL of a picture that represents the high confidence result. The HighConfidenceType column MUST contain the type of the high confidence result. It MUST NOT be empty. The possible value for this column is defined by the search administrator. The use of all other columns is up to the search administrator.

3.1.4.5.2.2.4 The RefinementResults Table

The **RefinementResults** table contains the refinement results. It MUST only be present if the **IncludeRefinementResults** element in the QueryPacket element of the QueryEx message contain one or more refiners specified in **Refiners** elements. The table MUST only contain data for the refiners specified in the **Refiners** element. It MUST have one row per refinement bucket, and it MUST have the columns RefinerName, RefinementName, RefinementValue, RefinementCountand RefinementToken. The columns MUST contain, respectively, the name of the refinable managed property, the display name of the refinement, the value of the refinement bucket, the result count for

this refinement, and the token used to apply the refinement through a new query. All of these columns are of string type, except RefinementCount which MUST be of type **integer**.

3.1.4.5.2.2.5 The VisualBestBetsResults Table

The **VisualBestBetsResults** table contains **visual best bets** that apply to the search query. It MUST only be present if the **IncludeSpecialTermResults** element in the QueryPacket element of the QueryEx operation was present and set to true. It MUST have one row per visual best bet, and it MUST have the columns Name, Uri, Description, Keyword, Teaser, TeaserContentType. The columns MUST contain, respectively, the name of the visual best bet, the URL that the visual best bet links to, a human-readable description of the visual best bet, a short summary to be displayed, and the content type of the summary. All of these columns MUST be of string type.

3.1.4.5.3 Complex Types

None.

3.1.4.5.4 Simple Types

None.

3.1.4.5.5 Attributes

None.

3.1.4.5.6 Groups

None.

3.1.4.5.7 Attribute Groups

None.

3.1.4.6 RecordClick

MUST NOT be used.

```
<wsdl:operation name="RecordClick">
  <wsdl:input message="tns:RecordClickSoapIn"/>
  <wsdl:output message="tns:RecordClickSoapOut"/>
</wsdl:operation>
```

3.1.4.6.1 Messages

The following table summarizes the set of WSDL message definitions that are specific to this operation.

Message	Description
RecordClickSoapIn	A request to initiate a **RecordClick** operation on the protocol server.
RecordClickSoapOut	A response from the protocol server at completion of the **RecordClick** operation.

3.1.4.6.1.1 RecordClickSoapIn

MUST NOT be used.

The SOAP action value of the message is defined as:

```
urn:Microsoft.Search/RecordClick
```

The SOAP body contains a **RecordClick** element.

3.1.4.6.1.2 RecordClickSoapOut

MUST NOT be used.

The SOAP body contains a RecordClickResponse element.

3.1.4.6.2 Elements

The following table summarizes the XML schema element definitions that are specific to this operation.

Element	Description
RecordClick	Body of the **RecordClickSoapIn** message.
RecordClickResponse	Body of the **RecordClickSoapOut** message.

3.1.4.6.2.1 RecordClick

MUST NOT be used.

```
<s:element name="RecordClick">
  <s:complexType>
    <s:sequence>
      <s:element name="clickInfoXml" type="s:string"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

3.1.4.6.2.2 RecordClickResponse

MUST NOT be used.

```
<s:element name="RecordClickResponse">
  <s:complexType/>
</s:element>
```

3.1.4.6.3 Complex Types

None.

3.1.4.6.4 Simple Types

None.

3.1.4.6.5 Attributes

None.

3.1.4.6.6 Groups

None.

3.1.4.6.7 Attribute Groups

None.

3.1.4.7 Registration

This operation is used by the protocol client to gather registration information from the protocol server. The protocol client does this mainly to gather information from the protocol server that can be shown in a user interface.

```
<wsdl:operation name="Registration">
  <wsdl:input message="tns:RegistrationSoapIn"/>
  <wsdl:output message="tns:RegistrationSoapOut"/>
</wsdl:operation>
```

The protocol client sends a RegistrationSoapIn request message and the protocol server responds with a RegistrationSoapOut response message, as follows:

The protocol server returns assorted information about itself.

3.1.4.7.1 Messages

The following table summarizes the set of WSDL message definitions that are specific to this operation.

Message	Description
RegistrationSoapIn	A request to initiate a **Registration** operation on the protocol server.
RegistrationSoapOut	A response from the protocol server at completion of the **Registration** operation.

3.1.4.7.1.1 RegistrationSoapIn

This message is used by the protocol client to request registration information from the protocol server.

The SOAP action value of the message is defined as:

```
urn:Microsoft.Search/Registration
```

The SOAP body contains a section Registration element.

3.1.4.7.1.2 RegistrationSoapOut

This message is used by the protocol server to respond to a RegistrationSoapIn message.

The SOAP body contains a RegistrationResponse element.

3.1.4.7.2 Elements

The following table summarizes the XML schema element definitions that are specific to this operation.

Element	Description
ProviderUpdate	The actual content of the protocol server's **RegistrationSoapOut** message.
Registration	Body of the **RegistrationSoapIn** message.
RegistrationRequest	The actual content of the protocol client's **RegistrationSoapIn** message.
RegistrationResponse	Body of the **RegistrationSoapOut** message.

3.1.4.7.2.1 ProviderUpdate

This element is the actual content of the protocol server's RegistrationSoapOut SOAP message. It contains all registration information about a protocol server.

The structure of this element assumes that one protocol server can have multiple **search applications**, and each search application can have multiple Office SharePoint Server Search services. However, this version of the Search Protocol only supports exactly one search application per protocol server, and exactly one Office SharePoint Server Search service per search application.

```
<s:element name="ProviderUpdate">
  <s:complexType>
    <s:sequence>
      <s:element name="Status" type="r:StatusType"/>
      <s:element name="DebugErrorMessage" type="s:string" minOccurs="0"/>
      <s:element name="Providers" minOccurs="0">
        <s:complexType>
          <s:sequence>
            <s:element name="Provider">
              <s:complexType>
                <s:sequence>
                  <s:element name="Id" type="t:GUIDType"/>
                  <s:element name="Name" type="t:String255"/>
                  <s:element name="QueryPath" type="s:anyURI"/>
                  <s:element name="Type" type="rrs:ProviderType"/>
                  <s:element name="Services" minOccurs="0">
                    <s:complexType>
                      <s:sequence>
                        <s:element name="Service">
                          <s:complexType>
                            <s:sequence>
                              <s:element name="Id" type="t:GUIDType"/>
                              <s:element name="Name" type="t:String255"/>
                              <s:element name="Category" type="t:CategoryType"/>
                              <s:element name="Description" type="t:String2048"
minOccurs="0"/>
                              <s:element name="Copyright" type="t:String2048" minOccurs="0"/>
                              <s:element name="Display" type="rrs:DisplayType"
minOccurs="0"/>
                            </s:sequence>
                          </s:complexType>
                        </s:element>
                      </s:sequence>
                    </s:complexType>
```

```
                    </s:element>
                  </s:sequence>
                </s:complexType>
              </s:element>
            </s:sequence>
          </s:complexType>
        </s:element>
      </s:sequence>
    </s:complexType>
  </s:element>
```

Status: Status of the request. For possible values, see section 2.2.5.5.

DebugErrorMessage: Contains an error message if an error occurred. If an error occurred, this element MUST contain an empty string or an error message. If no error occurred, this element MUST NOT be present.

Providers: Contains detailed information about the search application that this protocol server offers. Although the name of this element is plural, there MUST be exactly one search application per protocol server. This element MUST only be present if the **Status** element contains the value "SUCCESS". It MUST NOT be present in any other case.

Providers.Provider: Contains detailed information about a search application.

Providers.Provider.Id: GUID of the search application. This GUID MUST uniquely identify the search application.

Providers.Provider.Name: Name for the search application. This name is intended to be shown in the user interface of the protocol client. The length of this name, as specified in section 2.2.5.8, SHOULD be less than 80 characters to enable easy viewing on the protocol client application.

Providers.Provider.QueryPath: URL to which all query requests to send.

Providers.Provider.Type: Type of the search application, as specified in section 3.1.4.7.4.3, MUST always be set to "SOAP".

Providers.Provider.Services: Contains information about the Office SharePoint Server Search service that this search application offers. Although the name of this element is plural, there MUST be exactly one Office SharePoint Server Search service per search application.

Providers.Provider.Services.Service: Contains information about an Office SharePoint Server Search service.

Providers.Provider.Services.Service.Id: GUID of the Office SharePoint Server Search service. This GUID uniquely identifies the Office SharePoint Server Search service.

Providers.Provider.Services.Service.Name: A name in **String255** Simple Type format (as specified in section 2.2.5.8) for the Office SharePoint Server Search service.

Providers.Provider.Services.Service.Category: Category of the Office SharePoint Server Search service, as specified in section 3.1.4.7.4.1. The category given is intended to be shown in the user interface of the protocol client. MUST be set to "INTRANET_GENERAL".

Providers.Provider.Services.Service.Description: Description of the Office SharePoint Server Search service, as specified in section 2.2.5.7.

Providers.Provider.Services.Service.Copyright: Copyright information about the Office SharePoint Server Search service, as specified in section 2.2.5.7. This information is intended to be shown to an administrator only.

Providers.Provider.Services.Service.Display: Display state of the Office SharePoint Server Search service, as specified in section 3.1.4.7.4.2, MUST always be set to "On".

3.1.4.7.2.2 Registration

This element is the body of the RegistrationSoapIn **SOAP Message**.

```
<s:element name="Registration">
  <s:complexType>
    <s:sequence>
      <s:element name="registrationXml" type="s:string"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

registrationXml: MUST contain a BLOB of well-formed XML containing a RegistrationRequest element as its root element, encoded as an XML string (see [XML10]).

3.1.4.7.2.3 RegistrationRequest

This element is the actual content of the protocol client's RegistrationSoapIn SOAP message. All of its attributes and child elements MUST be ignored by the protocol server.

```
<s:element name="RegistrationRequest">
  <s:complexType>
    <s:sequence>
      <s:any minOccurs="0" maxOccurs="unbounded" processContents="skip"/>
    </s:sequence>
    <s:anyAttribute processContents="skip"/>
  </s:complexType>
</s:element>
```

3.1.4.7.2.4 RegistrationResponse

This element is the body of the RegistrationSoapOut SOAP message.

```
<s:element name="RegistrationResponse">
  <s:complexType>
    <s:sequence>
      <s:element name="RegistrationResult" type="s:string"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

RegistrationResult: MUST contain a BLOB of well-formed XML containing a ProviderUpdate element as its root element, encoded as an XML string (see [XML10]).

3.1.4.7.3 Complex Types

None.

3.1.4.7.4 Simple Types

The following table summarizes the XML schema simple type definitions that are specific to this operation.

Simple type	Description

Simple type	Description
CategoryType	Defines the type of category shown in a user interface.
DisplayType	Defines the visibility of an Office SharePoint Server Search service in a user interface.
ProviderType	Defines the type of a search application.

3.1.4.7.4.1 CategoryType

Target namespace: urn:Microsoft.Search.Registration.Response

This type defines the category under which an Office SharePoint Server Search service can be shown in a user interface. There is only one possible value, INTRANET_GENERAL, which is used for every field that has this type.

```
<s:simpleType name="CategoryType">
  <s:restriction base="s:string">
    <s:enumeration value="INTRANET_GENERAL"/>
  </s:restriction>
</s:simpleType>
```

The following table defines the allowable values for **CategoryType**:

Value	Meaning
INTRANET_GENERAL	The Office SharePoint Server Search service can be shown under a general intranet search category.

3.1.4.7.4.2 DisplayType

Target namespace: urn:Microsoft.Search.Registration.Response

This type defines the visibility of an Office SharePoint Server Search service in a user interface. There is only one possible value, "On", which is used for every field that has this type.

```
<s:simpleType name="DisplayType">
  <s:restriction base="s:string">
    <s:enumeration value="On"/>
  </s:restriction>
</s:simpleType>
```

The following table defines the allowable values for **DisplayType**:

Value	Meaning
On	The Office SharePoint Server Search service appears as on and visible in the protocol client's user interface.

3.1.4.7.4.3 ProviderType

Target namespace: urn:Microsoft.Search.Registration.Response

This type defines the type of a search application. There is only one possible value, "SOAP", which MUST be used for all fields of this type.

```
<s:simpleType name="ProviderType">
  <s:restriction base="s:string">
    <s:enumeration value="SOAP"/>
  </s:restriction>
</s:simpleType>
```

The following table defines the allowable values for **ProviderType**:

Value	Meaning
SOAP	The search application is of the SOAP type.

3.1.4.7.5 Attributes

None.

3.1.4.7.6 Groups

None.

3.1.4.7.7 Attribute Groups

None.

3.1.4.8 Status

This operation is used by the protocol client to retrieve the status of the protocol server.

```
<wsdl:operation name="Status">
  <wsdl:input message="tns:StatusSoapIn"/>
  <wsdl:output message="tns:StatusSoapOut"/>
</wsdl:operation>
```

The protocol client sends a StatusSoapIn request message and the protocol server responds with a StatusSoapOut response message, as follows:

The protocol server returns the string "ONLINE".

3.1.4.8.1 Messages

The following table summarizes the set of WSDL message definitions that are specific to this operation.

Message	Description
StatusSoapIn	A request to initiate a **Status** operation on the protocol server.

Message	Description
StatusSoapOut	A response from the protocol server at completion of the **Status** operation.

3.1.4.8.1.1 StatusSoapIn

This message is sent by the protocol client to request status information from the protocol server.

The SOAP action value of the message is defined as:

```
urn:Microsoft.Search/Status
```

The SOAP body contains a Status element.

3.1.4.8.1.2 StatusSoapOut

This message is used by the protocol server to respond to a StatusSoapIn message.

The SOAP body contains a StatusResponse element.

3.1.4.8.2 Elements

The following table summarizes the XML schema element definitions that are specific to this operation.

Element	Description
Status	Body of the **StatusSoapIn** message.
StatusRequest	Body of the **StatusSoapOut** message.

3.1.4.8.2.1 Status

This element is the body of the StatusSoapIn SOAP message.

```
<s:element name="Status">
  <s:complexType/>
</s:element>
```

Because the **Status** operation takes no parameters, this element MUST be empty.

3.1.4.8.2.2 StatusResponse

This element is the body of the StatusSoapOut SOAP message.

```
<s:element name="StatusResponse">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="StatusResult" type="s:string"/>
    </s:sequence>
  </s:complexType>
```

```
</s:element>
```

The following section defines the elements of the **StatusResponse** message:

StatusResult: The status of the server. This is a string that MUST always be set to "ONLINE".

3.1.4.8.3 Complex Types

None.

3.1.4.8.4 Simple Types

None.

3.1.4.8.5 Attributes

None.

3.1.4.8.6 Groups

None.

3.1.4.8.7 Attribute Groups

None.

3.1.5 Timer Events

None.

3.1.6 Other Local Events

None.

4 Protocol Examples

4.1 Obtain Information about Server Search Scopes

The protocol client might request information about the protocol server's search scopes by sending a request such as this one:

```
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap12="http://www.w3.org/2003/05/soap-
envelope">
  <soap12:Body>
    <GetPortalSearchInfo xmlns="http://microsoft.com/webservices/OfficeServer/QueryService"
/>
  </soap12:Body>
</soap12:Envelope>
```

The protocol server might respond with a message such as this one:

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
    <GetPortalSearchInfoResponse
xmlns="http://microsoft.com/webservices/OfficeServer/QueryService">
      <GetPortalSearchInfoResult>&lt;SiteConfigInfo
xmlns="urn:Microsoft.MSSearch.Response.Config"&gt;&lt;Name&gt;MCtest&lt;/Name&gt;&lt;Id&gt;{5
56f4ae3-17eb-44dc-8aa4-
9e7a7e512e69}&lt;/Id&gt;&lt;Scopes&gt;&lt;Scope&gt;&lt;Name&gt;People&lt;/Name&gt;&lt;/Scope&
gt;&lt;Scope&gt;&lt;Name&gt;All
Sites&lt;/Name&gt;&lt;/Scope&gt;&lt;Scope&gt;&lt;Name&gt;Global Query
Exclusion&lt;/Name&gt;&lt;/Scope&gt;&lt;Scope&gt;&lt;Name&gt;Rank Demoted
Sites&lt;/Name&gt;&lt;/Scope&gt;&lt;/Scopes&gt;&lt;/SiteConfigInfo&gt;</GetPortalSearchInfoRe
sult>
    </GetPortalSearchInfoResponse>
  </soap:Body>
</soap:Envelope>
```

The request information in the preceding message is doubly encoded XML. The following shows the decoded value of the **GetPortalSearchInfoResult** string:

```
<SiteConfigInfo xmlns="urn:Microsoft.MSSearch.Response.Config">
  <Name>MCtest</Name>
  <Id>{556f4ae3-17eb-44dc-8aa4-9e7a7e512e69}</Id>
  <Scopes>
    <Scope>
      <Name>People</Name>
    </Scope>
    <Scope>
      <Name>All Sites</Name>
    </Scope>
    <Scope>
      <Name>Global Query Exclusion</Name>
    </Scope>
    <Scope>
      <Name>Rank Demoted Sites</Name>
    </Scope>
  </Scopes>
</SiteConfigInfo>
```

4.2 Obtain Registration Information

The protocol client might request registration information by sending a request such as this one:

```
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <Registration xmlns="urn:Microsoft.Search">
      <registrationXml>&lt;RegistrationRequest revision=&#39;2&#39;
build=&#39;(12.0.6017)&#39;
xmlns=&#39;urn:Microsoft.Search.Registration.Request&#39;&gt;&lt;OriginatorId&gt;{F6FF7BE0-
F39C-4ddc-A7D0-
09A4C6C647A5}&lt;/OriginatorId&gt;&lt;SystemInformation&gt;&lt;SkuLanguage&gt;en-
us&lt;/SkuLanguage&gt;&lt;LanguagePack&gt;en-
us&lt;/LanguagePack&gt;&lt;InterfaceLanguage&gt;en-
us&lt;/InterfaceLanguage&gt;&lt;Location&gt;US&lt;/Location&gt;&lt;/SystemInformation&gt;&lt;
/RegistrationRequest&gt;</registrationXml>
    </Registration>
  </soap12:Body>
</soap12:Envelope>
```

The request information in the preceding message is doubly encoded XML. The following shows the decoded value of the **registrationXml** string:

```
<RegistrationRequest revision='2' build='(12.0.6017)'
xmlns='urn:Microsoft.Search.Registration.Request'>
  <OriginatorId>{F6FF7BE0-F39C-4ddc-A7D0-09A4C6C647A5}</OriginatorId>
  <SystemInformation>
    <SkuLanguage>en-us</SkuLanguage>
    <LanguagePack>en-us</LanguagePack>
    <InterfaceLanguage>en-us</InterfaceLanguage>
    <Location>US</Location>
  </SystemInformation>
</RegistrationRequest>
```

Note that the protocol server does not interpret this information; it only verifies that the XML is valid.

This might be protocol server's response to this message:

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope
  xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
    <RegistrationResponse xmlns="urn:Microsoft.Search">
      <RegistrationResult>&lt;ProviderUpdate
xmlns="urn:Microsoft.Search.Registration.Response"&gt;&lt;Status&gt;SUCCESS&lt;/Status&gt;&lt
;Providers&gt;&lt;Provider&gt;&lt;Id&gt;{09bd9f60-9072-4786-a2dc-
9863dc3a0c3a}&lt;/Id&gt;&lt;Name&gt;Microsoft Search Server
Fourteen&lt;/Name&gt;&lt;QueryPath&gt;http://example.com/_vti_bin/search.asmx&lt;/QueryPath&g
t;&lt;Type&gt;SOAP&lt;/Type&gt;&lt;Services&gt;&lt;Service&gt;&lt;Id&gt;{09bd9f60-9072-4786-
a2dc-9863dc3a0c3a}&lt;/Id&gt;&lt;Name&gt;Example
Site&lt;/Name&gt;&lt;Category&gt;INTRANET GENERAL&lt;/Category&gt;&lt;Description&gt;This
service allows you to search the site : Example
Site&lt;/Description&gt;&lt;Copyright&gt;MicrosoftÂ® Search Server
Fourteen&lt;/Copyright&gt;&lt;Display&gt;On&lt;/Display&gt;&lt;/Service&gt;&lt;/Services&gt;&
lt;/Provider&gt;&lt;/Providers&gt;&lt;/ProviderUpdate&gt;</RegistrationResult>
    </RegistrationResponse>
  </soap:Body>
```

```
</soap:Envelope>
```

The request information in the preceding message is doubly encoded XML. The following shows the decoded value of the **RegistrationResult** string:

```
<ProviderUpdate xmlns="urn:Microsoft.Search.Registration.Response">
  <Status>SUCCESS</Status>
  <Providers>
    <Provider>
      <Id>{09bd9f60-9072-4786-a2dc-9863dc3a0c3a}</Id>
      <Name>Microsoft Search Server Fourteen</Name>
      <QueryPath>http://example.com/_vti_bin/search.asmx</QueryPath>
      <Type>SOAP</Type>
      <Services>
        <Service>
          <Id>{09bd9f60-9072-4786-a2dc-9863dc3a0c3a}</Id>
          <Name>Example Site</Name>
          <Category>INTRANET_GENERAL</Category>
          <Description>This service allows you to search the site : Example
Site</Description>
          <Copyright>MicrosoftÂ® Search Server Fourteen</Copyright>
          <Display>On</Display>
        </Service>
      </Services>
    </Provider>
  </Providers>
</ProviderUpdate>
```

4.3 Perform a Query

The protocol client might query for documents which satisfies the following criteria:

- The document contains the text "get started"
- The document does not contain the word "vehicle"
- The title of the document contains the text "example site"
- The author of the document is either "domainname\username1" or "username2"

The issued query would be:

```
"get started" AND NOT vehicle title:"example site" author:domainname\username1 OR
author:username2
```

For this scenario, the protocol client sends the following request message:

```
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <Query xmlns="urn:Microsoft.Search">
      <queryXml xsi:type='xsd:string'>
        &lt;QueryPacket xmlns='urn:Microsoft.Search.Query'&gt;
        &lt;Query&gt;
        &lt;Context&gt;
        &lt;QueryText type='STRING' language='en-us' &gt;&quot;get started&quot; AND NOT
vehicle title:"example site" author:domainname\username1 OR
author:&quot;username2&quot;&lt;/QueryText&gt;
```

```
          &lt;LanguagePreference&gt;en-us&lt;/LanguagePreference&gt;
          &lt;/Context&gt;
          &lt;Properties&gt;
          &lt;Property name='path'/&gt;
          &lt;Property name='rank'/&gt;
          &lt;Property name='title'/&gt;
          &lt;Property name='author'/&gt;
          &lt;/Properties&gt;
          &lt;EnableStemming&gt;
          true
          &lt;/EnableStemming&gt;
          &lt;/Query&gt;
          &lt;/QueryPacket&gt;
        </queryXml>
      </Query>
    </soap12:Body>
  </soap12:Envelope>
```

The query information in the preceding message is doubly encoded XML. The following shows the decoded value of the Query string:

```
<QueryPacket xmlns='urn:Microsoft.Search.Query'>
  <Query>
    <Context>
      <QueryText type='STRING' language='en-us' >
        "get started" AND NOT vehicle title:"example site" author:domainname\username1 OR
author:"username2"
      </QueryText>
      <LanguagePreference>en-us</LanguagePreference>
    </Context>
    <Properties>
      <Property name='path'/>
      <Property name='rank'/>
      <Property name='title'/>
      <Property name='author'/>
    </Properties>
    <EnableStemming>
      true
    </EnableStemming>
  </Query>
</QueryPacket>
```

This might be a protocol server's response to this message:

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope
  xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
    <QueryResponse xmlns="urn:Microsoft.Search">
      <QueryResult>&lt;ResponsePacket
xmlns="urn:Microsoft.Search.Response"&gt;&lt;Response&gt;&lt;Range&gt;&lt;StartAt&gt;1&lt;/St
artAt&gt;&lt;Count&gt;2&lt;/Count&gt;&lt;TotalAvailable&gt;2&lt;/TotalAvailable&gt;&lt;Result
s&gt;&lt;Document
xmlns="urn:Microsoft.Search.Response.Document"&gt;&lt;Action&gt;&lt;LinkUrl&gt;http://example
.com&lt;/LinkUrl&gt;&lt;/Action&gt;&lt;Properties
xmlns="urn:Microsoft.Search.Response.Document.Document"&gt;&lt;Property&gt;&lt;Name&gt;path&l
t;/Name&gt;&lt;Type&gt;String&lt;/Type&gt;&lt;Value&gt;http://example.com&lt;/Value&gt;&lt;/P
roperty&gt;&lt;Property&gt;&lt;Name&gt;rank&lt;/Name&gt;&lt;Type&gt;Int64&lt;/Type&gt;&lt;Val
ue&gt;78543522&lt;/Value&gt;&lt;/Property&gt;&lt;Property&gt;&lt;Name&gt;title&lt;/Name&gt;&l
t;Type&gt;String&lt;/Type&gt;&lt;Value&gt;Example
Site&lt;/Value&gt;&lt;/Property&gt;&lt;Property&gt;&lt;Name&gt;author&lt;/Name&gt;&lt;Type&gt
;String&lt;/Type&gt;&lt;Value&gt;DOMAINNAME\USERNAME1&lt;/Value&gt;&lt;/Property&gt;&lt;/Prop
```

```
erties&gt;&lt;/Document&gt;&lt;Document
xmlns="urn:Microsoft.Search.Response.Document"&gt;&lt;Action&gt;&lt;LinkUrl
fileExt="aspx"&gt;http://example.com/Lists/Announcements/AllItems.aspx&lt;/LinkUrl&gt;&lt;/Ac
tion&gt;&lt;Properties
xmlns="urn:Microsoft.Search.Response.Document.Document"&gt;&lt;Property&gt;&lt;Name&gt;path&l
t;/Name&gt;&lt;Type&gt;String&lt;/Type&gt;&lt;Value&gt;http://example.com/Lists/Announcements
/AllItems.aspx&lt;/Value&gt;&lt;/Property&gt;&lt;Property&gt;&lt;Name&gt;rank&lt;/Name&gt;&lt
;Type&gt;Int64&lt;/Type&gt;&lt;Value&gt;75711418&lt;/Value&gt;&lt;/Property&gt;&lt;Property&g
t;&lt;Name&gt;title&lt;/Name&gt;&lt;Type&gt;String&lt;/Type&gt;&lt;Value&gt;Example Site -
Announcements&lt;/Value&gt;&lt;/Property&gt;&lt;Property&gt;&lt;Name&gt;author&lt;/Name&gt;&l
t;Type&gt;String&lt;/Type&gt;&lt;Value&gt;Username2&lt;/Value&gt;&lt;/Property&gt;&lt;/Proper
ties&gt;&lt;/Document&gt;&lt;/Results&gt;&lt;/Range&gt;&lt;Status&gt;SUCCESS&lt;/Status&gt;&l
t;/Response&gt;&lt;/ResponsePacket&gt;</QueryResult>
    </QueryResponse>
  </soap:Body>
</soap:Envelope>
```

The query information in the preceding message is doubly encoded XML. The following shows the decoded value of the QueryResponse string:

```
<ResponsePacket xmlns="urn:Microsoft.Search.Response">
  <Response>
    <Range>
      <StartAt>1</StartAt>
      <Count>2</Count>
      <TotalAvailable>2</TotalAvailable>
      <Results>
        <Document xmlns="urn:Microsoft.Search.Response.Document">
          <Action>
            <LinkUrl>http://example.com</LinkUrl>
          </Action>
          <Properties xmlns="urn:Microsoft.Search.Response.Document.Document">
            <Property>
              <Name>path</Name>
              <Type>String</Type>
              <Value>http://example.com</Value>
            </Property>
            <Property>
              <Name>rank</Name>
              <Type>Int64</Type>
              <Value>78543522</Value>
            </Property>
            <Property>
              <Name>title</Name>
              <Type>String</Type>
              <Value>Example Site</Value>
            </Property>
            <Property>
              <Name>author</Name>
              <Type>String</Type>
              <Value>DOMAINNAME\USERNAME1</Value>
            </Property>
          </Properties>
        </Document>
        <Document xmlns="urn:Microsoft.Search.Response.Document">
          <Action>
            <LinkUrl
fileExt="aspx">http://example.com/Lists/Announcements/AllItems.aspx</LinkUrl>
          </Action>
          <Properties xmlns="urn:Microsoft.Search.Response.Document.Document">
            <Property>
              <Name>path</Name>
              <Type>String</Type>
              <Value>http://example.com/Lists/Announcements/AllItems.aspx</Value>
            </Property>
            <Property>
              <Name>rank</Name>
```

```
              <Type>Int64</Type>
              <Value>75711418</Value>
            </Property>
            <Property>
              <Name>title</Name>
              <Type>String</Type>
              <Value>Example Site - Announcements</Value>
            </Property>
            <Property>
              <Name>author</Name>
              <Type>String</Type>
              <Value>Username2</Value>
            </Property>
          </Properties>
        </Document>
      </Results>
    </Range>
    <Status>SUCCESS</Status>
  </Response>
</ResponsePacket>
```

4.4 Obtain Status Information from the Server

The protocol client might request status information from protocol server by sending a request such as this one:

```
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap12="http://www.w3.org/2003/05/soap-
envelope">
  <soap12:Body>
    <Status xmlns="urn:Microsoft.Search" />
  </soap12:Body>
</soap12:Envelope>
```

The protocol server might respond with a message such as this one:

```
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap12="http://www.w3.org/2003/05/soap-
envelope">
  <soap12:Body>
    <StatusResponse xmlns="urn:Microsoft.Search">
      <StatusResult>ONLINE</StatusResult>
    </StatusResponse>
  </soap12:Body>
</soap12:Envelope>
```

4.5 GetSuggestedQueries

To obtain a list of suggested queries which contains the word "example", the protocol client sends a request message as follows:

```
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <GetQuerySuggestions
      xmlns="http://microsoft.com/webservices/OfficeServer/QueryService">
```

```
      <queryXml>&lt;?xml version=&quot;1.0&quot; encoding=&quot;utf-8&quot;
?&gt;&lt;QueryPacket xmlns=&quot;urn:Microsoft.Search.Query&quot; Revision=&quot;1&quot;&gt;
&lt;Query domain=&quot;QDomain&quot;&gt;    &lt;Context&gt;      &lt;QueryText
language=&quot;en-us&quot; type=&quot;STRING&quot;&gt;example&lt;/QueryText&gt;
&lt;/Context&gt;    &lt;Range&gt;      &lt;StartAt&gt;1&lt;/StartAt&gt;
&lt;Count&gt;8&lt;/Count&gt;    &lt;/Range&gt;
&lt;PreQuerySuggestions&gt;true&lt;/PreQuerySuggestions&gt;
&lt;HighlightQuerySuggestions&gt;false&lt;/HighlightQuerySuggestions&gt;
&lt;/Query&gt;&lt;/QueryPacket&gt;</queryXml>
    </GetQuerySuggestions>
  </soap12:Body>
</soap12:Envelope>
```

The request information in the preceding message is doubly encoded XML. The following shows the decoded value of the **queryXml** string:

```
<?xml version="1.0" encoding="utf-8" ?>
<QueryPacket xmlns="urn:Microsoft.Search.Query" Revision="1">
  <Query domain="QDomain">
    <Context>
      <QueryText language="en-us" type="STRING">example</QueryText>
    </Context>
    <Range>
      <StartAt>1</StartAt>
      <Count>8</Count>
    </Range>
    <PreQuerySuggestions>true</PreQuerySuggestions>
    <HighlightQuerySuggestions>false</HighlightQuerySuggestions>
  </Query>
</QueryPacket>
```

The protocol server responds with a message such as this one:

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope
  xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
    <GetQuerySuggestionsResponse
      xmlns="http://microsoft.com/webservices/OfficeServer/QueryService">
      <GetQuerySuggestionsResult>
        <string>example site</string>
        <string>client example</string>
        <string>server example</string>
        <string>example documents</string>
        <string>code example</string>
      </GetQuerySuggestionsResult>
    </GetQuerySuggestionsResponse>
  </soap:Body>
</soap:Envelope>
```

4.6 QueryEx

The protocol client might query for documents which contains word "SharePoint".

The issued query would be: SharePoint

For this scenario, the protocol client sends the following request message:

```
<QueryPacket xmlns="urn:Microsoft.Search.Query">
  <Query>
    <SupportedFormats>
      <Format>urn:Microsoft.Search.Response.Document:Document</Format>

    </SupportedFormats>
    <Context>
      <QueryText type="STRING" language="en-us">sharepoint</QueryText>
    </Context>
    <Range>
      <StartAt>1</StartAt>
      <Count>2</Count>
    </Range>
    <Properties>
      <Property name="path"/>
      <Property name="title"/>
    </Properties>
    <EnableStemming>true</EnableStemming>
    <TrimDuplicates>true</TrimDuplicates>
    <IgnoreAllNoiseQuery>true</IgnoreAllNoiseQuery>
    <ImplicitAndBehavior>true</ImplicitAndBehavior>
  </Query>

</QueryPacket>
```

This might be a protocol server's response to this message:

```
<?xml version="1.0" encoding="utf-8" ?>
 <DataSet xmlns="http://microsoft.com/webservices/OfficeServer/QueryService">
   <xs:schema id="Results" xmlns="" xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:msdata="urn:schemas-microsoft-com:xml-msdata" xmlns:msprop="urn:schemas-microsoft-
com:xml-msprop">
     <xs:element name="Results" msdata:IsDataSet="true" msdata:UseCurrentLocale="true"
msprop:QueryTerms="sharepoint;sharepoint;" msprop:IgnoredNoiseWords="" msprop:Keyword=""
msprop:QueryModification="" msprop:ElapsedTime="3646" msprop:Definition=""
msprop:SpellingSuggestion="">
       <xs:complexType>
         <xs:choice minOccurs="0" maxOccurs="unbounded">
           <xs:element name="RelevantResults" msprop:TotalRows="30"
msprop:IsTotalRowsExact="False">
             <xs:complexType>
               <xs:sequence>
                <xs:element name="path" type="xs:string" minOccurs="0" />
                <xs:element name="title" type="xs:string" minOccurs="0" />
              </xs:sequence>
            </xs:complexType>
          </xs:element>
        </xs:choice>
      </xs:complexType>
    </xs:element>
  </xs:schema>
   <diffgr:diffgram xmlns:msdata="urn:schemas-microsoft-com:xml-msdata"
xmlns:diffgr="urn:schemas-microsoft-com:xml-diffgram-v1">
     <Results xmlns="">
       <RelevantResults diffgr:id="RelevantResults1" msdata:rowOrder="0">
        <path>file://back/scratch/ddoorn/backup/giving it control v2.docx</path>
        <title>Giving IT Control</title>
      </RelevantResults>
      <RelevantResults diffgr:id="RelevantResults2" msdata:rowOrder="1">
        <path>file://back/scratch/ddoorn/backup/l0 dfd.vsd</path>
        <title>Sharepoint Search DFD</title>
      </RelevantResults>
```

```
    </Results>
  </diffgr:diffgram>
</DataSet>
```

4.7 GetSearchMetadata

The protocol client might request information about the protocol server's properties and search scopes.

This might be a protocol server's response to this message:

```
<?xml version="1.0" encoding="utf-8" ?>
 <DataSet xmlns="http://microsoft.com/webservices/OfficeServer/QueryService">
   <xs:schema id="SearchMetadata" xmlns="" xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:msdata="urn:schemas-microsoft-com:xml-msdata" xmlns:msprop="urn:schemas-microsoft-
com:xml-msprop">
     <xs:element name="SearchMetadata" msdata:IsDataSet="true" msdata:UseCurrentLocale="true"
msprop:SiteName="Search Center">
       <xs:complexType>
         <xs:choice minOccurs="0" maxOccurs="unbounded">
           <xs:element name="Properties">
             <xs:complexType>
               <xs:sequence>
                <xs:element name="Name" type="xs:string" minOccurs="0" />
                <xs:element name="Description" type="xs:string" minOccurs="0" />
                <xs:element name="Type" type="xs:string" minOccurs="0" />
                <xs:element name="Retrievable" type="xs:boolean" minOccurs="0" />
                <xs:element name="FullTextQueriable" type="xs:boolean" minOccurs="0" />
               </xs:sequence>
             </xs:complexType>
           </xs:element>
           <xs:element name="Scopes">
             <xs:complexType>
               <xs:sequence>
                <xs:element name="Name" type="xs:string" minOccurs="0" />
                <xs:element name="Description" type="xs:string" minOccurs="0" />
               </xs:sequence>
             </xs:complexType>
           </xs:element>
         </xs:choice>
       </xs:complexType>
     </xs:element>
   </xs:schema>
   <diffgr:diffgram xmlns:msdata="urn:schemas-microsoft-com:xml-msdata"
xmlns:diffgr="urn:schemas-microsoft-com:xml-diffgram-v1">
     <SearchMetadata xmlns="">
       <Properties diffgr:id="Properties1" msdata:rowOrder="0" diffgr:hasChanges="inserted">
        <Name>AboutMe</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
       </Properties>
       <Properties diffgr:id="Properties2" msdata:rowOrder="1" diffgr:hasChanges="inserted">
        <Name>Account</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>false</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
       </Properties>
       <Properties diffgr:id="Properties3" msdata:rowOrder="2" diffgr:hasChanges="inserted">
        <Name>AccountName</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
```

```
      </Properties>
       <Properties diffgr:id="Properties4" msdata:rowOrder="3" diffgr:hasChanges="inserted">
        <Name>AssignedTo</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties5" msdata:rowOrder="4" diffgr:hasChanges="inserted">
        <Name>Author</Name>
        <Description>author</Description>
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties6" msdata:rowOrder="5" diffgr:hasChanges="inserted">
        <Name>BaseOfficeLocation</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties7" msdata:rowOrder="6" diffgr:hasChanges="inserted">
        <Name>BestBetKeywords</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>false</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties8" msdata:rowOrder="7" diffgr:hasChanges="inserted">
        <Name>CategoryNavigationUrl</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>false</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties9" msdata:rowOrder="8" diffgr:hasChanges="inserted">
        <Name>CollapsingStatus</Name>
        <Description />
        <Type>System.Int64</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties10" msdata:rowOrder="9" diffgr:hasChanges="inserted">
        <Name>Colleagues</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>false</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties11" msdata:rowOrder="10"
diffgr:hasChanges="inserted">
        <Name>contact</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties12" msdata:rowOrder="11"
diffgr:hasChanges="inserted">
        <Name>contentclass</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties13" msdata:rowOrder="12"
diffgr:hasChanges="inserted">
        <Name>ContentsHidden</Name>
```

```
        <Description />
        <Type>System.String</Type>
        <Retrievable>false</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties14" msdata:rowOrder="13"
diffgr:hasChanges="inserted">
        <Name>ContentSource</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties15" msdata:rowOrder="14"
diffgr:hasChanges="inserted">
        <Name>ContentType</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>false</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties16" msdata:rowOrder="15"
diffgr:hasChanges="inserted">
        <Name>CreatedBy</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties17" msdata:rowOrder="16"
diffgr:hasChanges="inserted">
        <Name>Department</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties18" msdata:rowOrder="17"
diffgr:hasChanges="inserted">
        <Name>Description</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties19" msdata:rowOrder="18"
diffgr:hasChanges="inserted">
        <Name>DisplayDate</Name>
        <Description />
        <Type>System.DateTime</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties20" msdata:rowOrder="19"
diffgr:hasChanges="inserted">
        <Name>DocComments</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>false</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties21" msdata:rowOrder="20"
diffgr:hasChanges="inserted">
        <Name>DocId</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
```

```
       <Properties diffgr:id="Properties22" msdata:rowOrder="21"
diffgr:hasChanges="inserted">
        <Name>DocKeywords</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>false</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties23" msdata:rowOrder="22"
diffgr:hasChanges="inserted">
        <Name>DocSignature</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>false</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties24" msdata:rowOrder="23"
diffgr:hasChanges="inserted">
        <Name>DocSubject</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>false</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties25" msdata:rowOrder="24"
diffgr:hasChanges="inserted">
        <Name>DuplicateHash</Name>
        <Description />
        <Type>System.Int64</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties26" msdata:rowOrder="25"
diffgr:hasChanges="inserted">
        <Name>EMail</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties27" msdata:rowOrder="26"
diffgr:hasChanges="inserted">
        <Name>EndDate</Name>
        <Description />
        <Type>System.DateTime</Type>
        <Retrievable>false</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties28" msdata:rowOrder="27"
diffgr:hasChanges="inserted">
        <Name>ExcludeFromSummary</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>false</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties29" msdata:rowOrder="28"
diffgr:hasChanges="inserted">
        <Name>ExpirationTime</Name>
        <Description />
        <Type>System.DateTime</Type>
        <Retrievable>false</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties30" msdata:rowOrder="29"
diffgr:hasChanges="inserted">
        <Name>FileExtension</Name>
        <Description>file extension</Description>
        <Type>System.String</Type>
```

```
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties31" msdata:rowOrder="30"
diffgr:hasChanges="inserted">
        <Name>Filename</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties32" msdata:rowOrder="31"
diffgr:hasChanges="inserted">
        <Name>FirstName</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties33" msdata:rowOrder="32"
diffgr:hasChanges="inserted">
        <Name>FollowAllAnchor</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>false</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties34" msdata:rowOrder="33"
diffgr:hasChanges="inserted">
        <Name>HierarchyUrl</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties35" msdata:rowOrder="34"
diffgr:hasChanges="inserted">
        <Name>HighConfidenceDisplayProperty1</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties36" msdata:rowOrder="35"
diffgr:hasChanges="inserted">
        <Name>HighConfidenceDisplayProperty10</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties37" msdata:rowOrder="36"
diffgr:hasChanges="inserted">
        <Name>HighConfidenceDisplayProperty11</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties38" msdata:rowOrder="37"
diffgr:hasChanges="inserted">
        <Name>HighConfidenceDisplayProperty12</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties39" msdata:rowOrder="38"
diffgr:hasChanges="inserted">
```

```
        <Name>HighConfidenceDisplayProperty13</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties40" msdata:rowOrder="39"
diffgr:hasChanges="inserted">
        <Name>HighConfidenceDisplayProperty14</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties41" msdata:rowOrder="40"
diffgr:hasChanges="inserted">
        <Name>HighConfidenceDisplayProperty15</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties42" msdata:rowOrder="41"
diffgr:hasChanges="inserted">
        <Name>HighConfidenceDisplayProperty2</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties43" msdata:rowOrder="42"
diffgr:hasChanges="inserted">
        <Name>HighConfidenceDisplayProperty3</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties44" msdata:rowOrder="43"
diffgr:hasChanges="inserted">
        <Name>HighConfidenceDisplayProperty4</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties45" msdata:rowOrder="44"
diffgr:hasChanges="inserted">
        <Name>HighConfidenceDisplayProperty5</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties46" msdata:rowOrder="45"
diffgr:hasChanges="inserted">
        <Name>HighConfidenceDisplayProperty6</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties47" msdata:rowOrder="46"
diffgr:hasChanges="inserted">
        <Name>HighConfidenceDisplayProperty7</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
```

```
      </Properties>
       <Properties diffgr:id="Properties48" msdata:rowOrder="47"
diffgr:hasChanges="inserted">
        <Name>HighConfidenceDisplayProperty8</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties49" msdata:rowOrder="48"
diffgr:hasChanges="inserted">
        <Name>HighConfidenceDisplayProperty9</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties50" msdata:rowOrder="49"
diffgr:hasChanges="inserted">
        <Name>HighConfidenceImageURL</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties51" msdata:rowOrder="50"
diffgr:hasChanges="inserted">
        <Name>HighConfidenceMatching</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties52" msdata:rowOrder="51"
diffgr:hasChanges="inserted">
        <Name>HighConfidenceResultType</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties53" msdata:rowOrder="52"
diffgr:hasChanges="inserted">
        <Name>HitHighlightedProperties</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties54" msdata:rowOrder="53"
diffgr:hasChanges="inserted">
        <Name>HitHighlightedSummary</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties55" msdata:rowOrder="54"
diffgr:hasChanges="inserted">
        <Name>HostingPartition</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>false</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties56" msdata:rowOrder="55"
diffgr:hasChanges="inserted">
        <Name>ImageDateCreated</Name>
        <Description />
```

```
        <Type>System.DateTime</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties57" msdata:rowOrder="56"
diffgr:hasChanges="inserted">
        <Name>Interests</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties58" msdata:rowOrder="57"
diffgr:hasChanges="inserted">
        <Name>inttest</Name>
        <Description />
        <Type>System.Int64</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties59" msdata:rowOrder="58"
diffgr:hasChanges="inserted">
        <Name>IsDocument</Name>
        <Description />
        <Type>System.Boolean</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties60" msdata:rowOrder="59"
diffgr:hasChanges="inserted">
        <Name>JobTitle</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties61" msdata:rowOrder="60"
diffgr:hasChanges="inserted">
        <Name>Keywords</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties62" msdata:rowOrder="61"
diffgr:hasChanges="inserted">
        <Name>LastModifiedTime</Name>
        <Description />
        <Type>System.DateTime</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties63" msdata:rowOrder="62"
diffgr:hasChanges="inserted">
        <Name>LastName</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties64" msdata:rowOrder="63"
diffgr:hasChanges="inserted">
        <Name>Location</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
```

```
       <Properties diffgr:id="Properties65" msdata:rowOrder="64"
diffgr:hasChanges="inserted">
        <Name>Memberships</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties66" msdata:rowOrder="65"
diffgr:hasChanges="inserted">
        <Name>MetadataAuthor</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties67" msdata:rowOrder="66"
diffgr:hasChanges="inserted">
        <Name>MobilePhone</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties68" msdata:rowOrder="67"
diffgr:hasChanges="inserted">
        <Name>ModifiedBy</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>false</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties69" msdata:rowOrder="68"
diffgr:hasChanges="inserted">
        <Name>NLCodePage</Name>
        <Description />
        <Type>System.Int64</Type>
        <Retrievable>false</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties70" msdata:rowOrder="69"
diffgr:hasChanges="inserted">
        <Name>Notes</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>false</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties71" msdata:rowOrder="70"
diffgr:hasChanges="inserted">
        <Name>OfficeNumber</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties72" msdata:rowOrder="71"
diffgr:hasChanges="inserted">
        <Name>OrgNames</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties73" msdata:rowOrder="72"
diffgr:hasChanges="inserted">
        <Name>OrgParentNames</Name>
        <Description />
        <Type>System.String</Type>
```

```
        <Retrievable>true</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties74" msdata:rowOrder="73"
diffgr:hasChanges="inserted">
        <Name>OrgParentUrls</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties75" msdata:rowOrder="74"
diffgr:hasChanges="inserted">
        <Name>OrgUrls</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties76" msdata:rowOrder="75"
diffgr:hasChanges="inserted">
        <Name>OWS_URL</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>false</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties77" msdata:rowOrder="76"
diffgr:hasChanges="inserted">
        <Name>parentLink</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties78" msdata:rowOrder="77"
diffgr:hasChanges="inserted">
        <Name>PastProjects</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties79" msdata:rowOrder="78"
diffgr:hasChanges="inserted">
        <Name>Path</Name>
        <Description>path, DAV:href, VPath, DocAddress or Item</Description>
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties80" msdata:rowOrder="79"
diffgr:hasChanges="inserted">
        <Name>PictureHeight</Name>
        <Description />
        <Type>System.Int64</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties81" msdata:rowOrder="80"
diffgr:hasChanges="inserted">
        <Name>PictureThumbnailURL</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties82" msdata:rowOrder="81"
diffgr:hasChanges="inserted">
```

```
        <Name>PictureURL</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties83" msdata:rowOrder="82"
diffgr:hasChanges="inserted">
        <Name>PictureWidth</Name>
        <Description />
        <Type>System.Int64</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties84" msdata:rowOrder="83"
diffgr:hasChanges="inserted">
        <Name>PreferredName</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties85" msdata:rowOrder="84"
diffgr:hasChanges="inserted">
        <Name>Priority</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties86" msdata:rowOrder="85"
diffgr:hasChanges="inserted">
        <Name>PrivateColleagues</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>false</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties87" msdata:rowOrder="86"
diffgr:hasChanges="inserted">
        <Name>Pronunciations</Name>
        <Description>Pronuciations of people names</Description>
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties88" msdata:rowOrder="87"
diffgr:hasChanges="inserted">
        <Name>Purpose</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>false</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties89" msdata:rowOrder="88"
diffgr:hasChanges="inserted">
        <Name>Rank</Name>
        <Description />
        <Type>System.Int64</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties90" msdata:rowOrder="89"
diffgr:hasChanges="inserted">
        <Name>RankingWeightHigh</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>false</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
```

```
      </Properties>
       <Properties diffgr:id="Properties91" msdata:rowOrder="90"
diffgr:hasChanges="inserted">
        <Name>RankingWeightLow</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>false</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties92" msdata:rowOrder="91"
diffgr:hasChanges="inserted">
        <Name>RankingWeightName</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>false</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties93" msdata:rowOrder="92"
diffgr:hasChanges="inserted">
        <Name>Responsibilities</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties94" msdata:rowOrder="93"
diffgr:hasChanges="inserted">
        <Name>Schools</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties95" msdata:rowOrder="94"
diffgr:hasChanges="inserted">
        <Name>SecondaryFileExtension</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties96" msdata:rowOrder="95"
diffgr:hasChanges="inserted">
        <Name>ServerRedirectedURL</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties97" msdata:rowOrder="96"
diffgr:hasChanges="inserted">
        <Name>ServiceApplicationID</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties98" msdata:rowOrder="97"
diffgr:hasChanges="inserted">
        <Name>SipAddress</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties99" msdata:rowOrder="98"
diffgr:hasChanges="inserted">
        <Name>Site</Name>
        <Description />
```

```
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties100" msdata:rowOrder="99"
diffgr:hasChanges="inserted">
        <Name>SiteID</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>false</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties101" msdata:rowOrder="100"
diffgr:hasChanges="inserted">
        <Name>SiteTitle</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties102" msdata:rowOrder="101"
diffgr:hasChanges="inserted">
        <Name>Size</Name>
        <Description />
        <Type>System.Int64</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties103" msdata:rowOrder="102"
diffgr:hasChanges="inserted">
        <Name>Skills</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties104" msdata:rowOrder="103"
diffgr:hasChanges="inserted">
        <Name>SocialTagTextUrl</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>false</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties105" msdata:rowOrder="104"
diffgr:hasChanges="inserted">
        <Name>SPSiteURL</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>false</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties106" msdata:rowOrder="105"
diffgr:hasChanges="inserted">
        <Name>StartDate</Name>
        <Description />
        <Type>System.DateTime</Type>
        <Retrievable>false</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties107" msdata:rowOrder="106"
diffgr:hasChanges="inserted">
        <Name>Status</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
```

```
       <Properties diffgr:id="Properties108" msdata:rowOrder="107"
diffgr:hasChanges="inserted">
        <Name>Title</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties109" msdata:rowOrder="108"
diffgr:hasChanges="inserted">
        <Name>UrlDepth</Name>
        <Description />
        <Type>System.Int64</Type>
        <Retrievable>false</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties110" msdata:rowOrder="109"
diffgr:hasChanges="inserted">
        <Name>urn:schemas.microsoft.com:fulltextqueryinfo:cataloggroup</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties111" msdata:rowOrder="110"
diffgr:hasChanges="inserted">
        <Name>urn:schemas.microsoft.com:fulltextqueryinfo:sourcegroup</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties112" msdata:rowOrder="111"
diffgr:hasChanges="inserted">
        <Name>urn:schemas-microsoft-com:office:office#Description</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties113" msdata:rowOrder="112"
diffgr:hasChanges="inserted">
        <Name>urn:schemas-microsoft-com:office:office#ows_CrawlType</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties114" msdata:rowOrder="113"
diffgr:hasChanges="inserted">
        <Name>urn:schemas-microsoft-com:office:office#ows_ListTemplate</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties115" msdata:rowOrder="114"
diffgr:hasChanges="inserted">
        <Name>urn:schemas-microsoft-com:office:office#Subject</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties116" msdata:rowOrder="115"
diffgr:hasChanges="inserted">
        <Name>urn:schemas-microsoft-com:office:office#Title</Name>
        <Description />
        <Type>System.String</Type>
```

```
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties117" msdata:rowOrder="116"
diffgr:hasChanges="inserted">
        <Name>urn:schemas-microsoft-com:publishing:Category</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties118" msdata:rowOrder="117"
diffgr:hasChanges="inserted">
        <Name>urn:schemas-microsoft-com:publishing:CategoryTitle</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties119" msdata:rowOrder="118"
diffgr:hasChanges="inserted">
        <Name>urn:schemas-microsoft-com:sharepoint:portal:area:CategoryUrlNavi</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties120" msdata:rowOrder="119"
diffgr:hasChanges="inserted">
        <Name>urn:schemas-microsoft-com:sharepoint:portal:area:Path</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties121" msdata:rowOrder="120"
diffgr:hasChanges="inserted">
        <Name>UserName</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties122" msdata:rowOrder="121"
diffgr:hasChanges="inserted">
        <Name>UserProfile GUID</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties123" msdata:rowOrder="122"
diffgr:hasChanges="inserted">
        <Name>WebId</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties124" msdata:rowOrder="123"
diffgr:hasChanges="inserted">
        <Name>WikiCategory</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>false</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties125" msdata:rowOrder="124"
diffgr:hasChanges="inserted">
```

```
        <Name>WorkEmail</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>true</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties126" msdata:rowOrder="125"
diffgr:hasChanges="inserted">
        <Name>WorkPhone</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Properties diffgr:id="Properties127" msdata:rowOrder="126"
diffgr:hasChanges="inserted">
        <Name>YomiDisplayName</Name>
        <Description />
        <Type>System.String</Type>
        <Retrievable>true</Retrievable>
        <FullTextQueriable>false</FullTextQueriable>
      </Properties>
       <Scopes diffgr:id="Scopes1" msdata:rowOrder="0" diffgr:hasChanges="inserted">
        <Name>People</Name>
        <Description>Search for people.</Description>
      </Scopes>
       <Scopes diffgr:id="Scopes2" msdata:rowOrder="1" diffgr:hasChanges="inserted">
        <Name>All Sites</Name>
        <Description>Search for everything available for searching.</Description>
      </Scopes>
       <Scopes diffgr:id="Scopes3" msdata:rowOrder="2" diffgr:hasChanges="inserted">
        <Name>Global Query Exclusion</Name>
        <Description>Everything that should be omitted from all searches by
default.</Description>
      </Scopes>
       <Scopes diffgr:id="Scopes4" msdata:rowOrder="3" diffgr:hasChanges="inserted">
        <Name>Rank Demoted Sites</Name>
        <Description>Sites whose ranks will be demoted in click-distance
calculation.</Description>
      </Scopes>
       <Scopes diffgr:id="Scopes5" msdata:rowOrder="4" diffgr:hasChanges="inserted">
        <Name>test</Name>
        <Description />
      </Scopes>
       <Scopes diffgr:id="Scopes6" msdata:rowOrder="5" diffgr:hasChanges="inserted">
        <Name>ddorn</Name>
        <Description />
      </Scopes>
    </SearchMetadata>
  </diffgr:diffgram>
</DataSet>
```

5 Security

5.1 Security Considerations for Implementers

None.

5.2 Index of Security Parameters

None.

6 Appendix A: Full WSDL

For ease of implementation, the full WSDL and schema are provided in this appendix.

```
<?xml version="1.0"?>
<wsdl:definitions
  xmlns:s0="urn:Microsoft.Search"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns:tns="http://microsoft.com/webservices/OfficeServer/QueryService"
  xmlns:s="http://www.w3.org/2001/XMLSchema"
  xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
  xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
  xmlns:t="urn:Microsoft.Search.Types"
  xmlns:rrq="urn:Microsoft.Search.Registration.Request"
  xmlns:rrs="urn:Microsoft.Search.Registration.Response"
  xmlns:q="urn:Microsoft.Search.Query"
  xmlns:d="urn:Microsoft.Search.Response.Document"
  xmlns:r="urn:Microsoft.Search.Response"
  xmlns:diffgr="urn:schemas-microsoft-com:xml-diffgram-v1"
  targetNamespace="http://microsoft.com/webservices/OfficeServer/QueryService">
  <wsdl:types>
    <s:schema targetNamespace="urn:schemas-microsoft-com:xml-diffgram-v1"
      attributeFormDefault="qualified"
      elementFormDefault="qualified">
      <s:attribute name="id" type="s:string"/>
      <s:element name="diffgr">
        <s:complexType>
          <s:sequence minOccurs="0" maxOccurs="1">
            <s:any namespace="##other" processContents="lax" minOccurs="0"/>
          </s:sequence>
        </s:complexType>
      </s:element>
    </s:schema>
    <s:schema elementFormDefault="qualified"
targetNamespace="urn:Microsoft.Search.Types">
      <s:simpleType name="SimilarToType">
        <s:restriction base="s:string">
          <s:pattern value="(\[([\w ])+,(([0-9]+)?\.)?[0-9]+\])+"/>
        </s:restriction>
      </s:simpleType>
      <s:simpleType name="String255">
        <s:restriction base="s:string">
          <s:minLength value="0"/>
          <s:maxLength value="255"/>
        </s:restriction>
      </s:simpleType>
      <s:simpleType name="String2048">
        <s:restriction base="s:string">
          <s:minLength value="0"/>
          <s:maxLength value="2048"/>
        </s:restriction>
      </s:simpleType>
      <s:simpleType name="GUIDType">
        <s:restriction base="s:string">
          <s:pattern value="\{[A-Fa-f0-9]{8}-([A-Fa-f0-9]{4}-){3}[A-Fa-f0-
9]{12}\}"/>
          <s:pattern value="[A-Fa-f0-9]{8}-([A-Fa-f0-9]{4}-){3}[A-Fa-f0-9]{12}"/>
        </s:restriction>
      </s:simpleType>
      <s:simpleType name="StartAtType">
```

```
          <s:restriction base="s:unsignedInt">
            <s:minInclusive value="1"/>
          </s:restriction>
        </s:simpleType>
        <s:simpleType name="CategoryType">
          <s:restriction base="s:string">
            <s:enumeration value="INTRANET_GENERAL"/>
          </s:restriction>
        </s:simpleType>
      </s:schema>
      <s:schema elementFormDefault="qualified"
targetNamespace="urn:Microsoft.Search.Registration.Request">
        <s:element name="RegistrationRequest">
          <s:complexType>
            <s:sequence>
              <s:any minOccurs="0" maxOccurs="unbounded" processContents="skip"/>
            </s:sequence>
            <s:anyAttribute processContents="skip"/>
          </s:complexType>
        </s:element>
      </s:schema>
      <s:schema elementFormDefault="qualified"
targetNamespace="urn:Microsoft.Search.Registration.Response">
        <s:import namespace="urn:Microsoft.Search.Types"/>
        <s:import namespace="urn:Microsoft.Search.Response"/>
        <s:element name="ProviderUpdate">
          <s:complexType>
            <s:sequence>
              <s:element name="Status" type="r:StatusType"/>
              <s:element name="DebugErrorMessage" type="s:string" minOccurs="0"/>
              <s:element name="Providers" minOccurs="0">
                <s:complexType>
                  <s:sequence>
                    <s:element name="Provider">
                      <s:complexType>
                        <s:sequence>
                          <s:element name="Id" type="t:GUIDType"/>
                          <s:element name="Name" type="t:String255"/>
                          <s:element name="QueryPath" type="s:anyURI"/>
                          <s:element name="Type" type="rrs:ProviderType"/>
                          <s:element name="Services" minOccurs="0">
                            <s:complexType>
                              <s:sequence>
                                <s:element name="Service">
                                  <s:complexType>
                                    <s:sequence>
                                      <s:element name="Id" type="t:GUIDType"/>
                                      <s:element name="Name" type="t:String255"/>
                                      <s:element name="Category"
type="t:CategoryType"/>
                                      <s:element name="Description"
type="t:String2048" minOccurs="0"/>
                                      <s:element name="Copyright" type="t:String2048"
minOccurs="0"/>
                                      <s:element name="Display"
type="rrs:DisplayType" minOccurs="0"/>
                                    </s:sequence>
                                  </s:complexType>
                                </s:element>
                              </s:sequence>
                            </s:complexType>
```

```
                        </s:element>
                      </s:sequence>
                    </s:complexType>
                  </s:element>
                </s:sequence>
              </s:complexType>
            </s:element>
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:simpleType name="ProviderType">
        <s:restriction base="s:string">
          <s:enumeration value="SOAP"/>
        </s:restriction>
      </s:simpleType>
      <s:simpleType name="DisplayType">
        <s:restriction base="s:string">
          <s:enumeration value="On"/>
        </s:restriction>
      </s:simpleType>
    </s:schema>
    <s:schema elementFormDefault="qualified"
targetNamespace="urn:Microsoft.Search.Response.Document">
      <s:element name="Document">
        <s:complexType>
          <s:sequence>
            <s:element name="Title" type="s:string" minOccurs="0"/>
            <s:element name="Action">
              <s:complexType>
                <s:sequence>
                  <s:element name="LinkUrl">
                    <s:complexType>
                      <s:simpleContent>
                        <s:extension base="s:string">
                          <s:attribute name="size" type="s:unsignedInt"
use="optional"/>
                          <s:attribute name="fileExt" type="s:string"
use="optional"/>
                        </s:extension>
                      </s:simpleContent>
                    </s:complexType>
                  </s:element>
                </s:sequence>
              </s:complexType>
            </s:element>
            <s:choice>
              <s:sequence>
                <s:element name="Description" type="s:string"/>
                <s:element name="Date" type="s:dateTime"/>
              </s:sequence>
              <s:element name="Properties">
                <s:complexType>
                  <s:sequence>
                    <s:element name="Property" minOccurs="0" maxOccurs="unbounded">
                      <s:complexType>
                        <s:sequence>
                          <s:element name="Name" type="s:string"/>
                          <s:element name="Type" type="d:PropertyType"/>
                          <s:element name="Value" type="s:string"/>
                        </s:sequence>
                      </s:complexType>
```

```
                    </s:element>
                  </s:sequence>
                </s:complexType>
              </s:element>
            </s:choice>
          </s:sequence>
          <s:attribute name="relevance" type="s:double"/>
        </s:complexType>
      </s:element>
      <s:simpleType name="PropertyType">
        <s:restriction base="s:string">
          <s:enumeration value="Boolean"/>
          <s:enumeration value="Byte"/>
          <s:enumeration value="Char"/>
          <s:enumeration value="DateTime"/>
          <s:enumeration value="Double"/>
          <s:enumeration value="Int16"/>
          <s:enumeration value="Int32"/>
          <s:enumeration value="Int64"/>
          <s:enumeration value="Single"/>
          <s:enumeration value="String"/>
          <s:enumeration value="UInt16"/>
          <s:enumeration value="UInt32"/>
          <s:enumeration value="UInt64"/>
        </s:restriction>
      </s:simpleType>
    </s:schema>
    <s:schema elementFormDefault="qualified"
targetNamespace="urn:Microsoft.Search.Query">
      <s:import namespace="urn:Microsoft.Search.Types"/>
      <s:element name="QueryPacket">
        <s:complexType>
          <s:all>
            <s:element name="Query">
              <s:complexType>
                <s:all>
                  <s:element name="QueryId" type="t:GUIDType" minOccurs="0"/>
                  <s:element name="OriginatorId" type="t:GUIDType" minOccurs="0"/>
                  <s:element name="SupportedFormats">
                    <s:complexType>
                      <s:sequence>
                        <s:any processContents="skip" minOccurs="0"
maxOccurs="unbounded"/>
                      </s:sequence>
                    </s:complexType>
                  </s:element>
                  <s:element name="Context">
                    <s:complexType>
                      <s:sequence>
                        <s:element name="QueryText">
                          <s:complexType>
                            <s:simpleContent>
                              <s:extension base="s:string">
                                <s:attribute name="language" type="s:language"
use="optional"/>
                                <s:attribute name="type" type="q:QueryType"
use="optional" default="STRING"/>
                              </s:extension>
                            </s:simpleContent>
                          </s:complexType>
                        </s:element>
```

```
                        <s:element name="LanguagePreference" type="s:language"
minOccurs="0"/>
                        <s:element name="Requery" minOccurs="0">
                          <s:complexType>
                            <s:sequence>
                              <s:any processContents="skip" minOccurs="0"
maxOccurs="unbounded"/>
                            </s:sequence>
                          </s:complexType>
                        </s:element>
                        <s:element name="OriginatorContext" minOccurs="0">
                          <s:complexType>
                            <s:sequence>
                              <s:any processContents="skip" minOccurs="0"
maxOccurs="unbounded"/>
                            </s:sequence>
                          </s:complexType>
                        </s:element>
                      </s:sequence>
                    </s:complexType>
                  </s:element>
                  <s:element name="Range" minOccurs="0">
                    <s:complexType>
                      <s:sequence>
                        <s:element name="StartAt" type="t:StartAtType" default="1"
minOccurs="0"/>
                        <s:element name="Count" type="s:unsignedInt"
minOccurs="0"/>
                      </s:sequence>
                      <s:attribute name="id" type="s:string" use="optional"/>
                    </s:complexType>
                  </s:element>
                  <s:element name="Keywords" minOccurs="0">
                    <s:complexType>
                      <s:sequence>
                        <s:any minOccurs="0" maxOccurs="unbounded"
processContents="skip"/>
                      </s:sequence>
                      <s:anyAttribute processContents="skip"/>
                    </s:complexType>
                  </s:element>
                  <s:element name="OfficeContext" minOccurs="0">
                    <s:complexType>
                      <s:sequence>
                        <s:any minOccurs="0" maxOccurs="unbounded"
processContents="skip"/>
                      </s:sequence>
                      <s:anyAttribute processContents="skip"/>
                    </s:complexType>
                  </s:element>
                  <s:element name="Properties" minOccurs="0">
                    <s:complexType>
                      <s:sequence>
                        <s:element name="Property" minOccurs="0"
maxOccurs="unbounded">
                          <s:complexType>
                            <s:attribute name="name" type="s:string"/>
                          </s:complexType>
                        </s:element>
                      </s:sequence>
                    </s:complexType>
```

```
                    </s:element>
                    <s:element name="SortByProperties" minOccurs="0">
                      <s:complexType>
                        <s:sequence>
                          <s:element name="SortByProperty" minOccurs="0"
maxOccurs="unbounded">
                            <s:complexType>
                              <s:attribute name="name" type="s:string"/>
                              <s:attribute name="direction" type="q:DirectionType"
use="optional"/>
                            </s:complexType>
                          </s:element>
                        </s:sequence>
                      </s:complexType>
                    </s:element>
                    <s:element name="ImplicitAndBehavior" type="s:boolean"
minOccurs="0"/>
                    <s:element name="RelevanceModel" type="t:GUIDType"
minOccurs="0"/>
                    <s:element name="EnableStemming" type="s:boolean" minOccurs="0"/>
                    <s:element name="EnableNicknames" type="s:boolean"
minOccurs="0"/>
                    <s:element name="EnablePhonetic" type="s:boolean" minOccurs="0"/>
                    <s:element name="TrimDuplicates" minOccurs="0">
                      <s:complexType>
                        <s:simpleContent>
                          <s:extension base="s:boolean">
                            <s:attribute name="onproperty" type="s:string"/>
                            <s:attribute name="keepcount" type="s:unsignedInt"/>
                            <s:attribute name="includeid" type="s:unsignedInt"/>
                          </s:extension>
                        </s:simpleContent>
                      </s:complexType>
                    </s:element>
                    <s:element name="IncludeSpecialTermResults" type="s:boolean"
minOccurs="0"/>
                    <s:element name="PreQuerySuggestions" type="s:boolean"
minOccurs="0"/>
                    <s:element name="HighlightQuerySuggestions" type="s:boolean"
minOccurs="0"/>
                    <s:element name="CapitalizeFirstLetters" type="s:boolean"
minOccurs="0"/>
                    <s:element name="ResultProvider" type="s:string" minOccurs="0" />
                    <s:element name="ResubmitFlags" minOccurs="0">
                      <s:complexType>
                        <s:sequence>
                          <s:element name="ResubmitFlag" minOccurs="0"
maxOccurs="unbounded">
                            <s:complexType>
                              <s:attribute name="value" type="s:string"/>
                            </s:complexType>
                          </s:element>
                        </s:sequence>
                      </s:complexType>
                    </s:element>
                    <s:element name="EnableSpellcheck" type="s:string" minOccurs="0"
/>
                    <s:element name="UserContext" minOccurs="0">
                      <s:complexType>
                        <s:attribute name="includeuserprofile" type="s:boolean"
use="optional"/>
```

```
                    <s:sequence>
                      <s:element name="UserContextData" type="s:string"
minOccurs="0" />
                    </s:sequence>
                  </s:complexType>
                </s:element>
                <s:element name="FindSimilar" minOccurs="0">
                  <s:complexType>
                    <s:sequence>
                      <s:element name="SimilarType" type="s:string"
minOccurs="0"/>
                      <s:element name="SimilarTo" type="t:SimilarToType"
minOccurs="0"/>
                      <s:element name="SortSimilar" type="s:boolean"
minOccurs="0"/>
                    </s:sequence>
                  </s:complexType>
                </s:element>
                <s:element name="IncludeRefinementResults" minOccurs="0">
                  <s:complexType>
                    <s:sequence>
                      <s:element name="Refiners" minOccurs="0">
                        <s:complexType>
                          <s:sequence>
                            <s:element name="Refiner" type="s:string"
minOccurs="0" />
                          </s:sequence>
                        </s:complexType>
                      </s:element>
                      <s:element name="MaxShallowRefinementHits" type="s:int"
minOccurs="0"/>
                    </s:sequence>
                  </s:complexType>
                </s:element>
                <s:element name="RefinementFilters" minOccurs="0">
                  <s:complexType>
                    <s:sequence>
                      <s:element name="RefinementFilter" type="s:string"
minOccurs="0" maxOccurs="unbounded">
                      </s:element>
                    </s:sequence>
                  </s:complexType>
                </s:element>
                <s:element name="IgnoreAllNoiseQuery" type="s:boolean"
minOccurs="0"/>
                <s:element name="IncludeRelevantResults" type="s:boolean"
minOccurs="0"/>
                <s:element name="IncludeHighConfidenceResults" type="s:boolean"
minOccurs="0"/>
              </s:all>
              <s:attribute name="domain" type="t:String255" use="optional"/>
            </s:complexType>
          </s:element>
        </s:all>
        <s:attribute name="revision" type="s:unsignedInt" use="optional"/>
        <s:attribute name="build" type="t:String255" use="optional"/>
      </s:complexType>
    </s:element>
    <s:simpleType name="QueryType">
      <s:restriction base="s:string">
        <s:enumeration value="MSSQLFT"/>
```

```
            <s:enumeration value="STRING"/>
          </s:restriction>
        </s:simpleType>
        <s:simpleType name="DirectionType">
          <s:restriction base="s:string">
            <s:enumeration value="Ascending"/>
            <s:enumeration value="Descending"/>
          </s:restriction>
        </s:simpleType>
      </s:schema>
      <s:schema elementFormDefault="qualified"
  targetNamespace="urn:Microsoft.Search.Response">
        <s:import namespace="urn:Microsoft.Search.Types"/>
        <s:import namespace="urn:Microsoft.Search.Response.Document"/>
        <s:element name="ResponsePacket">
          <s:complexType>
            <s:sequence>
              <s:element name="Response">
                <s:complexType>
                  <s:sequence>
                    <s:element name="QueryId" type="t:GUIDType" minOccurs="0"/>
                    <s:element name="Range" minOccurs="0">
                      <s:complexType>
                        <s:sequence>
                          <s:element name="StartAt" type="t:StartAtType"/>
                          <s:element name="Count" type="s:unsignedInt"/>
                          <s:element name="TotalAvailable" type="s:unsignedInt"/>
                          <s:element name="Results" minOccurs="0">
                            <s:complexType>
                              <s:sequence>
                                <s:element ref="d:Document" maxOccurs="unbounded"/>
                              </s:sequence>
                            </s:complexType>
                          </s:element>
                        </s:sequence>
                      </s:complexType>
                    </s:element>
                    <s:element name="Status" type="r:StatusType"/>
                    <s:element name="DebugErrorMessage" type="t:String2048"
  minOccurs="0"/>
                  </s:sequence>
                  <s:attribute name="domain" type="t:String255" use="optional"/>
                </s:complexType>
              </s:element>
            </s:sequence>
          </s:complexType>
        </s:element>
        <s:simpleType name="StatusType">
          <s:restriction base="s:string">
            <s:enumeration value="SUCCESS"/>
            <s:enumeration value="ERROR_ALL_NOISE"/>
            <s:enumeration value="ERROR_NO_RESPONSE"/>
            <s:enumeration value="ERROR_BAD_QUERY"/>
            <s:enumeration value="ERROR_BAD_PROPERTY"/>
            <s:enumeration value="ERROR_BAD_SCOPE"/>
            <s:enumeration value="ERROR_BAD_REQUEST"/>
            <s:enumeration value="ERROR_NO_RESULTS_FOUND"/>
            <s:enumeration value="ERROR_NO_QUERY"/>
            <s:enumeration value="ERROR_NO_AUTHORIZATION"/>
            <s:enumeration value="ERROR_SERVER"/>
          </s:restriction>
```

```
      </s:simpleType>
    </s:schema>
    <s:schema elementFormDefault="qualified"
targetNamespace="urn:Microsoft.Search">
      <s:element name="Query">
        <s:complexType>
          <s:sequence>
            <s:element name="queryXml" type="s:string"/>
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="QueryResponse">
        <s:complexType>
          <s:sequence>
            <s:element name="QueryResult" type="s:string"/>
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="Registration">
        <s:complexType>
          <s:sequence>
            <s:element name="registrationXml" type="s:string"/>
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="RegistrationResponse">
        <s:complexType>
          <s:sequence>
            <s:element name="RegistrationResult" type="s:string"/>
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="Status">
        <s:complexType/>
      </s:element>
      <s:element name="StatusResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="StatusResult"
type="s:string"/>
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="RecordClick">
        <s:complexType>
          <s:sequence>
            <s:element name="clickInfoXml" type="s:string"/>
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="RecordClickResponse">
        <s:complexType/>
      </s:element>
    </s:schema>
    <s:schema elementFormDefault="qualified"
targetNamespace="http://microsoft.com/webservices/OfficeServer/QueryService">
      <s:import namespace="urn:Microsoft.Search.Types"/>
      <s:import namespace="urn:schemas-microsoft-com:xml-diffgram-v1"/>
      <s:element name="QueryEx">
        <s:complexType>
          <s:sequence>
```

```
            <s:element name="queryXml" type="s:string"/>
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="QueryExResponse">
        <s:complexType>
          <s:sequence>
            <s:element name="QueryExResult" minOccurs="0">
              <s:complexType>
                <s:sequence>
                  <s:element ref="s:schema"/>
                  <s:any/>
                </s:sequence>
              </s:complexType>
            </s:element>
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="GetPortalSearchInfo">
        <s:complexType/>
      </s:element>
      <s:element name="GetPortalSearchInfoResponse">
        <s:complexType>
          <s:sequence>
            <s:element name="GetPortalSearchInfoResult" type="s:string"
minOccurs="0"/>
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="GetQuerySuggestions">
        <s:complexType>
          <s:sequence>
            <s:element name="queryXml" type="s:string" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="GetQuerySuggestionsResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="GetQuerySuggestionsResult"
type="tns:ArrayOfString" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:complexType name="ArrayOfString">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="unbounded" name="string"
nillable="true" type="s:string" />
        </s:sequence>
      </s:complexType>
      <s:element name="GetSearchMetadata">
        <s:complexType/>
      </s:element>
      <s:element name="GetSearchMetadataResponse">
        <s:complexType>
          <s:sequence>
            <s:element name="GetSearchMetadataResult">
              <s:complexType>
                <s:sequence>
                  <s:element ref="s:schema"/>
                  <s:any/>
```

```
              </s:sequence>
            </s:complexType>
          </s:element>
        </s:sequence>
      </s:complexType>
    </s:element>
    <s:element name="SiteConfigInfo">
      <s:complexType>
        <s:sequence>
          <s:element name="Name" type="s:string"/>
          <s:element name="Id" type="t:GUIDType"/>
          <s:element name="Scopes">
            <s:complexType>
              <s:sequence>
                <s:element name="Scope" minOccurs="0" maxOccurs="unbounded">
                  <s:complexType>
                    <s:sequence>
                      <s:element name="Name" type="s:string"/>
                    </s:sequence>
                  </s:complexType>
                </s:element>
              </s:sequence>
            </s:complexType>
          </s:element>
        </s:sequence>
      </s:complexType>
    </s:element>
  </s:schema>
</wsdl:types>
<wsdl:message name="QuerySoapIn">
  <wsdl:part name="parameters" element="s0:Query"/>
</wsdl:message>
<wsdl:message name="QuerySoapOut">
  <wsdl:part name="parameters" element="s0:QueryResponse"/>
</wsdl:message>
<wsdl:message name="QueryExSoapIn">
  <wsdl:part name="parameters" element="tns:QueryEx"/>
</wsdl:message>
<wsdl:message name="QueryExSoapOut">
  <wsdl:part name="parameters" element="tns:QueryExResponse"/>
</wsdl:message>
<wsdl:message name="RegistrationSoapIn">
  <wsdl:part name="parameters" element="s0:Registration"/>
</wsdl:message>
<wsdl:message name="RegistrationSoapOut">
  <wsdl:part name="parameters" element="s0:RegistrationResponse"/>
</wsdl:message>
<wsdl:message name="StatusSoapIn">
  <wsdl:part name="parameters" element="s0:Status"/>
</wsdl:message>
<wsdl:message name="StatusSoapOut">
  <wsdl:part name="parameters" element="s0:StatusResponse"/>
</wsdl:message>
<wsdl:message name="GetPortalSearchInfoSoapIn">
  <wsdl:part name="parameters" element="tns:GetPortalSearchInfo"/>
</wsdl:message>
<wsdl:message name="GetPortalSearchInfoSoapOut">
  <wsdl:part name="parameters" element="tns:GetPortalSearchInfoResponse"/>
</wsdl:message>
<wsdl:message name="GetQuerySuggestionsSoapIn">
  <wsdl:part name="parameters" element="tns:GetQuerySuggestions" />
```

```
</wsdl:message>
<wsdl:message name="GetQuerySuggestionsSoapOut">
  <wsdl:part name="parameters" element="tns:GetQuerySuggestionsResponse" />
</wsdl:message>
<wsdl:message name="GetSearchMetadataSoapIn">
  <wsdl:part name="parameters" element="tns:GetSearchMetadata"/>
</wsdl:message>
<wsdl:message name="GetSearchMetadataSoapOut">
  <wsdl:part name="parameters" element="tns:GetSearchMetadataResponse"/>
</wsdl:message>
<wsdl:message name="RecordClickSoapIn">
  <wsdl:part name="parameters" element="s0:RecordClick"/>
</wsdl:message>
<wsdl:message name="RecordClickSoapOut">
  <wsdl:part name="parameters" element="s0:RecordClickResponse"/>
</wsdl:message>
<wsdl:portType name="QueryServiceSoap">
  <wsdl:operation name="Query">
    <wsdl:input message="tns:QuerySoapIn"/>
    <wsdl:output message="tns:QuerySoapOut"/>
  </wsdl:operation>
  <wsdl:operation name="QueryEx">
    <wsdl:input message="tns:QueryExSoapIn"/>
    <wsdl:output message="tns:QueryExSoapOut"/>
  </wsdl:operation>
  <wsdl:operation name="Registration">
    <wsdl:input message="tns:RegistrationSoapIn"/>
    <wsdl:output message="tns:RegistrationSoapOut"/>
  </wsdl:operation>
  <wsdl:operation name="Status">
    <wsdl:input message="tns:StatusSoapIn"/>
    <wsdl:output message="tns:StatusSoapOut"/>
  </wsdl:operation>
  <wsdl:operation name="GetPortalSearchInfo">
    <wsdl:input message="tns:GetPortalSearchInfoSoapIn"/>
    <wsdl:output message="tns:GetPortalSearchInfoSoapOut"/>
  </wsdl:operation>
  <wsdl:operation name="GetQuerySuggestions">
    <wsdl:input message="tns:GetQuerySuggestionsSoapIn" />
    <wsdl:output message="tns:GetQuerySuggestionsSoapOut" />
  </wsdl:operation>
  <wsdl:operation name="GetSearchMetadata">
    <wsdl:input message="tns:GetSearchMetadataSoapIn"/>
    <wsdl:output message="tns:GetSearchMetadataSoapOut"/>
  </wsdl:operation>
  <wsdl:operation name="RecordClick">
    <wsdl:input message="tns:RecordClickSoapIn"/>
    <wsdl:output message="tns:RecordClickSoapOut"/>
  </wsdl:operation>
</wsdl:portType>
<wsdl:binding name="QueryServiceSoap" type="tns:QueryServiceSoap">
  <soap:binding transport="http://schemas.xmlsoap.org/soap/http"/>
  <wsdl:operation name="Query">
    <soap:operation soapAction="urn:Microsoft.Search/Query" style="document"/>
    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal"/>
    </wsdl:output>
  </wsdl:operation>
```

```
    <wsdl:operation name="QueryEx">
      <soap:operation
soapAction="http://microsoft.com/webservices/OfficeServer/QueryService/QueryEx"
style="document"/>
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="Registration">
      <soap:operation soapAction="urn:Microsoft.Search/Registration"
style="document"/>
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="Status">
      <soap:operation soapAction="urn:Microsoft.Search/Status" style="document"/>
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetPortalSearchInfo">
      <soap:operation
soapAction="http://microsoft.com/webservices/OfficeServer/QueryService/GetPortalSea
rchInfo" style="document"/>
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation><wsdl:operation name="GetQuerySuggestions">
      <soap:operation
soapAction="http://microsoft.com/webservices/OfficeServer/QueryService/GetQuerySugg
estions" style="document"/>
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>


<wsdl:operation name="GetSearchMetadata">
 <soap:operation
soapAction="http://microsoft.com/webservices/OfficeServer/QueryService/GetSearc
hMetadata" style="document"/>

      <wsdl:input>
        <soap:body use="literal"/>
```

```
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="RecordClick">
      <soap:operation soapAction="urn:Microsoft.Search/RecordClick"
style="document"/>
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
  </wsdl:binding>
  <wsdl:binding name="QueryServiceSoap12" type="tns:QueryServiceSoap">
    <soap12:binding transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="Query">
      <soap12:operation soapAction="urn:Microsoft.Search/Query" style="document"/>
      <wsdl:input>
        <soap12:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="QueryEx">
      <soap12:operation
soapAction="http://microsoft.com/webservices/OfficeServer/QueryService/QueryEx"
style="document"/>
      <wsdl:input>
        <soap12:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="Registration">
      <soap12:operation soapAction="urn:Microsoft.Search/Registration"
style="document"/>
      <wsdl:input>
        <soap12:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="Status">
      <soap12:operation soapAction="urn:Microsoft.Search/Status" style="document"/>
      <wsdl:input>
        <soap12:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetPortalSearchInfo">
      <soap12:operation
soapAction="http://microsoft.com/webservices/OfficeServer/QueryService/GetPortalSea
rchInfo" style="document"/>
```

```
      <wsdl:input>
        <soap12:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>

    <wsdl:operation name="GetSearchMetadata">
      <soap12:operation
soapAction="http://microsoft.com/webservices/OfficeServer/QueryService/GetSearchMet
adata" style="document"/>
      <wsdl:input>
        <soap12:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="RecordClick">
      <soap12:operation soapAction="urn:Microsoft.Search/RecordClick"
style="document"/>
      <wsdl:input>
        <soap12:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
  </wsdl:binding>
</wsdl:definitions>
```

7 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 Search Server 2010
- Microsoft FAST Search Server 2010
- The 2007 Microsoft Office system
- Microsoft Office 2010 suites
- Microsoft SharePoint Foundation 2010
- Windows SharePoint Services 2.0
- Windows SharePoint Services 3.0
- Microsoft Office 2013
- Microsoft SharePoint Foundation 2013
- Microsoft Office 2016
- Microsoft SharePoint Server 2016

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.

<1> Section 2.1: In Windows SharePoint Services 3.0, the correct suffix is "/_vti_bin/spsearch.asmx". In SharePoint Foundation 2010, the correct suffix is "/_vti_bin/psearch.asmx".

<2> Section 2.2.3.1: This element is ignored by Windows SharePoint Services 2.0, Windows SharePoint Services 3.0, SharePoint Foundation 2010, The 2007 Office system, Microsoft Office SharePoint Server 2007, Microsoft SharePoint Server 2010, Microsoft Search Server 2010, FAST Search Server 2010, and Office 2010.

<3> Section 2.2.3.1: This element is ignored by Windows SharePoint Services 2.0, Windows SharePoint Services 3.0, SharePoint Foundation 2010, The 2007 Office system, Office SharePoint Server 2007, SharePoint Server 2010, Search Server 2010, FAST Search Server 2010, and Office 2010.

<4> Section 2.2.3.1: This doesn't apply to FAST Search Server 2010.

<5> Section 2.2.3.1: For FAST Search Server 2010 the protocol server MUST return the status code "ERROR_SERVER"

<6> Section 2.2.3.1: This element is only supported in SharePoint Server 2010, Search Server 2010, and SharePoint Foundation 2010.

<7> Section 2.2.3.1: This element is only supported in SharePoint Server 2010 and Search Server 2010.

<8> Section 2.2.3.1: This element is only supported in SharePoint Server 2010 and Search Server 2010.

<9> Section 2.2.3.1: This element is ignored by Office SharePoint Server 2007, Windows SharePoint Services 2.0, Windows SharePoint Services 3.0, and SharePoint Foundation 2010.

<10> Section 2.2.3.1: The "FASTSearch" result provider is only supported by FAST Search Server 2010.

<11> Section 2.2.5.1: This value is only supported in FAST Search Server 2010.

<12> Section 2.2.5.3: This value is not supported in Office 2013 and SharePoint Foundation 2013.

<13> Section 2.2.5.3: This value is only supported in FAST Search Server 2010.

<14> Section 2.2.10: This version of SharePoint Search Keyword Syntax applies to Office SharePoint Server 2007 and Windows SharePoint Services 3.0.

<15> Section 2.2.11: This version of SharePoint Search Keyword Syntax applies to SharePoint Server 2010, Search Server 2010, FAST Search Server 2010, and SharePoint Foundation 2010.

<16> Section 2.2.11.9: In Search Server 2010 and SharePoint Server 2010, if the ANY restriction is used together with the ALL restriction, unqualified tokens, tokens qualified with +, or a combination thereof, it matches an item if it contains zero or more of the tokens supplied between the parentheses. In other words, it matches all items, and the supplied tokens are used only to supplement search relevance.

<17> Section 2.2.12: This version of SharePoint Search SQL Syntax applies to Office SharePoint Server 2007 and Windows SharePoint Services 3.0.

<18> Section 2.2.12.2: The 2007 Office system, Office SharePoint Server 2007 and Windows SharePoint Services 3.0 don't have a limit on the length of a query. They are only limited by the resources of the hardware on which they are running.

<19> Section 2.2.13: This version of SharePoint Search SQL Syntax applies to SharePoint Server 2010, Search Server 2010, and SharePoint Foundation 2010.

<20> Section 2.2.13.2: The 2007 Office system, Office SharePoint Server 2007 and Windows SharePoint Services 3.0 don't have a limit on the length of a query in SQL syntax. They are only limited by the resources of the hardware on which they are running.

<21> Section 3.1.1.1: In Office SharePoint Server 2007, Windows SharePoint Services 2.0 and Windows SharePoint Services 3.0, the token can also be enclosed in <c1> through <c9> tags.

<22> Section 3.1.1.1: In Office SharePoint Server 2007, Windows SharePoint Services 2.0 and Windows SharePoint Services 3.0, the token can also be enclosed in <c1> through <c9> tags.

<23> Section 3.1.1.1: In Office 2013 this property is of type Double and is not bounded. The higher value means the higher relevance.

<24> Section 3.1.1.1: This property doesn't exist in FAST Search Server 2010, Windows SharePoint Services 2.0, Windows SharePoint Services 3.0, and SharePoint Foundation 2010.

<25> Section 3.1.4: This operation does not exist in Windows SharePoint Services 2.0, Windows SharePoint Services 3.0 and SharePoint Foundation 2010.

<26> Section 3.1.4: This operation is only available in Search Server 2010 and SharePoint Server 2010.

<27> Section 3.1.4: This operation does not exist in Windows SharePoint Services 2.0, Windows SharePoint Services 3.0 and SharePoint Foundation 2010.

<28> Section 3.1.4: This operation does not exist in Windows SharePoint Services 2.0, Windows SharePoint Services 3.0 and SharePoint Foundation 2010.

<29> Section 3.1.4: This operation does not exist in Windows SharePoint Services 2.0, Windows SharePoint Services 3.0 and SharePoint Foundation 2010.

<30> Section 3.1.4.3.2.2: For FAST Search Server 2010 the **DataSet** also contains the FASTSearchProperties Table.

<31> Section 3.1.4.3.2.2.2: This is only supported for FAST Search Server 2010.

<32> Section 3.1.4.4.2.1: If the property of the crawled item is multivalue property, then SharePoint Server 2010, Search Server 2010, and SharePoint Foundation 2010 set this element to "Object".

8 Change Tracking

This section identifies changes that were made to this document since the last release. Changes are classified as New, Major, Minor, Editorial, or No change.

The revision class **New** means that a new document is being released.

The revision class **Major** means that the technical content in the document was significantly revised. Major changes affect protocol interoperability or implementation. Examples of major changes are:

- A document revision that incorporates changes to interoperability requirements or functionality.
- The removal of a document from the documentation set.

The revision class **Minor** means that the meaning of the technical content was clarified. Minor changes do not affect protocol interoperability or implementation. Examples of minor changes are updates to clarify ambiguity at the sentence, paragraph, or table level.

The revision class **Editorial** means that the formatting in the technical content was changed. Editorial changes apply to grammatical, formatting, and style issues.

The revision class **No change** means that no new technical changes were introduced. Minor editorial and formatting changes may have been made, but the technical content of the document is identical to the last released version.

Major and minor changes can be described further using the following change types:

- New content added.
- Content updated.
- Content removed.
- New product behavior note added.
- Product behavior note updated.
- Product behavior note removed.
- New protocol syntax added.
- Protocol syntax updated.
- Protocol syntax removed.
- New content added due to protocol revision.
- Content updated due to protocol revision.
- Content removed due to protocol revision.
- New protocol syntax added due to protocol revision.
- Protocol syntax updated due to protocol revision.
- Protocol syntax removed due to protocol revision.
- Obsolete document removed.

Editorial changes are always classified with the change type **Editorially updated**.

Some important terms used in the change type descriptions are defined as follows:

- **Protocol syntax** refers to data elements (such as packets, structures, enumerations, and methods) as well as interfaces.
- **Protocol revision** refers to changes made to a protocol that affect the bits that are sent over the wire.

The changes made to this document are listed in the following table. For more information, please contact dochelp@microsoft.com.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
7 Appendix B: Product Behavior	Updated list of supported products.	Y	Content updated due to protocol revision.

9 Index

T

V

W