[MS-SQLPADM2]:
SQL Administration
Version 2 Protocol Specification

Revision Summary

Date	Revision History	Revision Class	Comments
07/13/2009	0.1	Major	Initial Availability
08/28/2009	0.2	Editorial	Revised and edited the technical content
11/06/2009	0.3	Editorial	Revised and edited the technical content
02/19/2010	1.0	Major	Updated and revised the technical content
03/31/2010	1.01	Editorial	Revised and edited the technical content
04/30/2010	1.02	Editorial	Revised and edited the technical content
06/07/2010	1.03	Editorial	Revised and edited the technical content
06/29/2010	1.04	Minor	Clarified the meaning of the technical content.
07/23/2010	1.05	Editorial	Changed language and formatting in the technical content.
09/27/2010	1.05	No change	No changes to the meaning, language, or formatting of the technical content.
11/15/2010	1.06	Editorial	Changed language and formatting in the technical content.
12/17/2010	1.06	No change	No changes to the meaning, language, or formatting of the technical content.
03/18/2011	1.06	No change	No changes to the meaning, language, or formatting of the technical content.
06/10/2011	1.06	No change	No changes to the meaning, language, or formatting of the technical content.
01/20/2012	1.7	Minor	Clarified the meaning of the technical content.
04/11/2012	1.7	No change	No changes to the meaning, language, or formatting of the technical content.
07/16/2012	1.7	No change	No changes to the meaning, language, or formatting of the technical content.

Table of Contents

1 Introduction

Specifies the SQL Administration Protocol, which are the communication sequences that are used by the protocol client (Web and application servers) to perform data query and update commands on the protocol server in relation to search administration functions.

Sections 1.8, 2, and 3 of this specification are normative and can contain the terms MAY, SHOULD, MUST, MUST NOT, and SHOULD NOT as defined in RFC 2119. Sections 1.5 and 1.9 are also normative but cannot contain those terms. All other sections and examples in this specification are informative.

1.1 Glossary

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

Coordinated Universal Time (UTC)
GUID
HRESULT
language code identifier (LCID)
update sequence number (USN)

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

application server
authority level
authority page
back-end database server
best bet
binary large object (BLOB)
catalog
content source
crawl
crawl log
crawl status
crawl store
crawled property
crawled property category
crawled property set identifier
crawler
datetime
display URL
extracted definition
federated location
folder
front-end Web server
full-text index catalog
host name
index server
item
keyword
keyword consumer
keyword consumer group
keyword synonym
locale
managed property

managed property alias
mapping order
metadata index
metadata schema
post-query suggestions
pre-query suggestions
property identifier
proxy
query
query independent rank
query result
query text
ranking model
result set
return code
search application
search catalog
search query
search scope
search scope compilation
search scope consumer
search scope display group
search scope index
search scope rule
search scope rule value
search scopes system
search service application
site
site collection
site collection administrator
static rank
stored procedure
Structured Query Language (SQL)
subdomain
token
Transact-Structured Query Language (T-SQL)
Uniform Resource Identifier (URI)
Uniform Resource Locator (URL)
variant type
visualization
XML fragment
XML namespace
XML namespace prefix

The following terms are specific to this document:

alternate access mapping: A mapping of URLs to Web applications. Incoming alternate access mappings are used to provide multiple URL entry points for the same set of content. Outgoing alternate access mappings are used to ensure that content is rendered in the correct URL context.

extracted term: A term that an extracted definition applies to.

search property mapping: A mapping that defines the relationship between a crawled property and a managed property. See also mapping order.

search user preference: A unique group of search settings that is associated with a specific user of a specific search service application. The settings include whether to provide query suggestions and whether to display a list of search query languages from which the user can choose.

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

1.2 References

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

1.2.1 Normative References

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

[Iseminger] Microsoft Corporation, "SQL Server 2000 Architecture and XML/Internet Support", Volume 1 of Microsoft SQL Server 2000 Reference Library, Microsoft Press, 2001, ISBN 0-7356-1280-3, http://www.microsoft.com/mspress/books/5001.aspx

[MS-CIFO] Microsoft Corporation, "Content Index Format Structure Specification".

[MSDN-TSQL-Ref] Microsoft Corporation, "Transact-SQL Reference", http://msdn.microsoft.com/en-us/library/ms189826(SQL.90).aspx

[MS-ERREF] Microsoft Corporation, "Windows Error Codes".

[MS-SEARCH] Microsoft Corporation, "Search Protocol Specification".

[MS-SQLPGAT2] Microsoft Corporation, "SQL Gatherer Version 2 Protocol Specification".

[MS-SQLPQ2] Microsoft Corporation, "Search Service Database Query Version 2 Protocol Specification".

[MS-SRCHTP] Microsoft Corporation, "Search Topology Protocol Specification".

[MS-TDS] Microsoft Corporation, "Tabular Data Stream Protocol Specification".

[MS-WSSFO2] Microsoft Corporation, "Windows SharePoint Services (WSS): File Operations Database Communications Version 2 Protocol Specification".

[OpenSearch1.1-Draft3] DeWitt, C., "Specifications/OpenSearch/1.1/Draft 3", http://www.opensearch.org/Specifications/OpenSearch/1.1

[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

[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.S., Ed., Beech, D., Ed., Maloney, M., Ed., and Mendelsohn, N., Ed., "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

[MSDN-ESCRXT] Microsoft Corporation, "Enterprise Search Core Results XSLT Transformation", http://msdn.microsoft.com/en-us/library/ms584121.aspx

[MS-GLOS] Microsoft Corporation, "Windows Protocols Master Glossary".

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

1.3 Protocol Overview (Synopsis)

This protocol specifies the communication between the **front-end Web server** and the **back-end database server** used to satisfy requests for common search administration tasks. This server-to-server protocol uses the Tabular Data Stream Protocol, as specified in [MS-TDS], as its transport between the front-end Web server and the back-end database server.

1.3.1 Metadata Schema

The protocol allows clients to add, change, retrieve, and delete **metadata schema** information from a store on the back-end database server. Metadata schema consists of **crawled properties**, the **crawled property categories** that contain the crawled properties, managed properties, and the **managed property aliases** that refer to the managed properties.

The protocol allows clients to retrieve a list of all crawled property categories defined in the metadata schema. Clients can add a crawled property category and change its attributes.

The protocol allows clients to query for crawled properties using a filter either within a single crawled property category or over all crawled property categories. The protocol allows clients to add a crawled property to the crawled property category, configure it to map string data to the **full-text index catalog**, as well as retrieve a list of **items** that contain the crawled property.

The protocol allows clients to retrieve a list of all managed properties defined in the metadata schema. Clients can add a managed property and change its attributes. Clients can also associate a managed property alias with a managed property, retrieve a list of all managed property aliases associated with a managed property, or delete the association.

The protocol allows clients to map a crawled property to a managed property. Clients can retrieve a list of mappings for either a crawled property or a managed property**.** When new mappings for a managed property are being added, a Pending Mappings Set, as specified in section 3.1.1.1, is used to hold the uncommitted mappings while they are being assembled. First, pending mappings for the managed property are deleted. Then mappings are added one at a time. If the managed property is configured to respect the order priority of the crawled properties, then the order in which the mappings are added is preserved.

The protocol allows clients to delete a managed property alias associated with a managed property. The protocol clients can delete all the mappings between a managed property and a crawled property. The protocol allows clients to delete a managed property if the managed property has no mappings to a crawled property. The set of crawled properties within a crawled property category

can be deleted in one call if they have no mappings and are not mapped to the full-text index. Finally, when a crawled property category is empty, it can be deleted.

1.3.2 Best Bets and Keywords

The protocol allows clients to customize query handling by defining specific words or phrases as **keywords**. For those keywords, clients can define **best bets** and **keyword synonyms** to be returned as part of the query results.

The protocol allows clients to add, modify, and delete keywords from a store on the back-end database server. The clients can retrieve the list of keywords and their totals.

The protocol allows clients to add, change, and delete best bets information from a store on the back-end database server. The protocol allows a keyword to have more than one best bet associated with it, and a best bet can be associated with multiple keywords. The order of this association can be retrieved and updated. Clients configure the best bets list separately from the keywords list. After a best bet is added to this list, it can be associated with the appropriate entries in the keywords list. The protocol also allows clients to retrieve the list of best bets**,** and their totals.

The protocol allows clients to add, and delete keyword synonyms information from a store on the back-end database server. The protocol allows clients to retrieve the list keyword synonyms and their totals.

1.3.3 Crawl Log

To support the logging of crawls, the protocol allows retrieval of Crawl Url History from the **crawl log.**

To support crawl administration, the protocol client allows the retrieval of success, error, warning, delete, and top level error counts:

- For all content sources or for specified **content source** for past crawls.
- For items as of the last time they were crawled by the **index server** which can be filtered by **display URL**, content source, **host name**, error type or date range.
- For all hosts processed by the index server. The client can retrieve the details about errors and warnings.

1.3.4 Scopes

The protocol allows clients to manage **search scope** information using a store on the back-end database server. Overviews of the three general types of scope-related tasks are given in this section:

- Scope and rule data flow, which is used to manage the makeup of scopes.
- Consumer-related scope data flow, which is used to manage the relationship between search scopes and **search scope consumers**.
- Display group data flow, which is used to manage how scopes are displayed in the user interface.

The protocol allows clients to add, change and delete search scope information from a store on the back-end database server. The protocol clients can retrieve search scopes and their totals. The client can retrieve the status of the search scope, such as the state of the compilation, the time the compilation started, and the percentage of compilation completed.

The protocol allows clients to retrieve and change **search scopes system** information which governs the **search scope compilation** schedule. The type of search scope compilation schedule can be set to either automatic or on-demand update. The compilation of search scopes can be started on demand even if the search scope compilation schedule is set to automatic, a compilation is currently in progress or the compilation is being stopped.

The protocol allows clients to add, change and delete search scope rule information for the specified search scope. The protocol clients can retrieve search scope rules and their totals. A search scope rule or a search scope can be deleted.

The protocol allows the client to add a search scope consumer. The client can retrieve the names of all search scope consumers. The **search scope display groups**, used and unused search scopes owned by a search scope consumer can be listed. The list of search scope consumers who own search scope display groups, search scopes or **search scope rules** of search scopes that have changed after a specific version can be retrieved. Search scopes, search scope display groups, and ranks of compiled search scopes within search scope display groups owned by a specific search scope consumer can be listed.

The protocol allows clients to add, change and delete search scope display group information from a store on the back-end database server. The protocol clients can retrieve search scope display groups information and their totals. The search scopes are grouped together in an ordered set within a search scope display group. A search scope is added to a search scope display group with the specified order. To do this, the protocol client first prepares the list of search scopes for the search scope display group, then adds each search scope to the list, and finally commits the list. The list is prepared by deleting any search scope for the specified search scope display group that has a negative order value. When the search scope is added to the list, the order value is negative until the search scope list is committed at which point the order value is changed to positive.

1.3.5 Relevance

The protocol allows the client to administer the data which drives query relevance in the following ways:

- The protocol allows the client to add, modify, retrieve and delete **ranking models** used to compute a score for ranking items to be returned as query results.
- The protocol allows the client to add, retrieve and delete language resources used in language-related relevance calculations.
- The protocol allows the client to retrieve a list of **authority pages**, to add an authority page, to delete an authority page, to retrieve the set of ranking parameters, and to change the ranking parameter values. An item's distance to authority pages is used to compute the **static rank**.

1.3.6 Federated Search

The protocol allows clients to add, modify, retrieve, and delete **federated location** and **visualization** information from a store on the back-end database server.

The protocol allows the client to add, change and delete a federated location. The client can add a visualization for the federated location. The protocol allows the client to retrieve the federated location configuration and visualizations associated with it. The client can delete the specified federated location which also deletes visualizations associated with it.

1.3.7 Search User Preference

The protocol allows clients to add, modify, and retrieve search preference settings per user.

1.4 Relationship to Other Protocols

This protocol relies on [MS-TDS] as its transport protocol to call **stored procedures** to inspect and manipulate item properties via **result sets** and **return codes**.

This relationship is illustrated in the following figure:

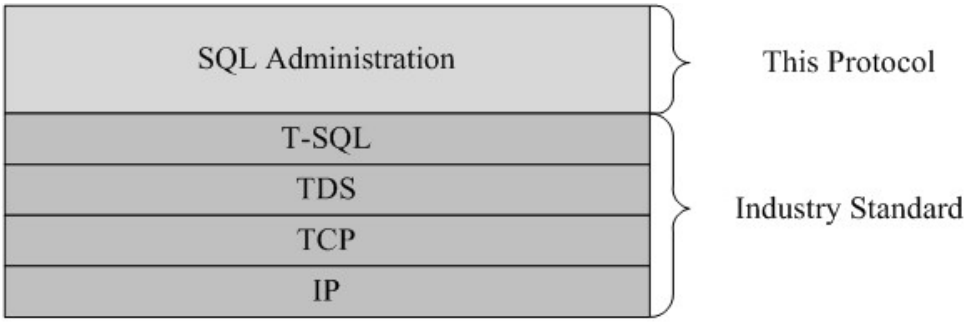

Figure 1: This protocol in relation to other protocols

1.5 Prerequisites/Preconditions

Unless otherwise specified, the stored procedures and any related tables are present in the database that is being queried on the back-end database server. The tables in the database contain valid data in a consistent state to be queried successfully by the stored procedures.

1.6 Applicability Statement

This protocol is only applicable to front-end Web server servers when communicating with the back-end database server to satisfy requests for search administration tasks as part of search administration.

1.7 Versioning and Capability Negotiation

Version Negotiation

Versions of the data structures or stored procedures in the database require the same calling parameters and return code values that are expected by the front-end Web server in order for the stored procedures to be called correctly. If the stored procedures are not provided the expected calling parameters or return code values, the results of the call are indeterminate. The **application server** uses stored procedure **proc_MSS_GetDatabaseSchemaVersion** (see [MS-SRCHTP] section 3.1.5.31) to retrieve version of the protocol implemented on the back-end database server and continues using that server only if that version is supported.

This document covers versioning issues in the following areas:

Security and Authentication Methods

This protocol supports SSPI and SQL Authentication with the back-end database server. These authentication methods are defined in [MS-TDS].

1.8 Vendor-Extensible Fields

This protocol uses **HRESULT** values as defined in [MS-ERREF] section 2.1. Vendors can define their own HRESULT values, provided they set the C bit (0x20000000) for each vendor-defined value, indicating the value is a customer code.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

[MS-TDS] is the transport protocol used to call the stored procedures, query **SQL** Views or SQL Tables and return codes and result sets.

2.2 Common Data Types

This section contains common definitions used by this protocol.

2.2.1 Simple Data Types and Enumerations

The following subsections define the simple data types for this specification.

2.2.1.1 Authentication Type

Authentication Type specifies the different types of authentication supported for federation. The value MUST be an 8-bit integer listed in the following table:

Value	Description
0	Anonymous
1	Local Windows Authentication
2	Single Account Basic Authentication
3	Single Account Form Based Authentication
4	Single Account Cookie Based Authentication
5	Application Pool Identity Based Authentication
7	Kerberos Authentication
8	Per User Form Based Authentication
9	Per User Cookie Based Authentication
10	Single Sign On Authentication
11	Single Account NTLM Based Authentication
12	Per User NTLM Based Authentication
13	Single User Digest Based Authentication
14	Per User Digest Based Authentication
15	Per User Basic Authentication
16	Custom Authentication
17	Per User Custom Authentication

2.2.1.2 Best Bet Filter Type

Best Bets Filter: int. Identifies which attributes of the best bet are used for filtering the list of best bets**.** The attributes are specified as part of Best Bet Set in section 3.1.1.2. The value MUST be set to an integer listed in the following table:

Value	Description
0	Filter based on Title
1	Filter based on URL
2	Filter based on both Title and URL

2.2.1.3 Compilation Schedule Type

Compilation Schedule Type: smallint. Identifies the type of schedule for the search scope compilation. The value MUST be an integer listed in the following table:

Value	Description
0	Search scope compilation only occurs on demand.
1	Search scope compilation occurs automatically on a self-adjusting schedule
2	Search scope compilation occurs based on the schedule specified by the administrator.

2.2.1.4 Compilation State

Compilation State: smallint. Identifies the state of search scope compilation for a given search scope. The value MUST be an integer listed in the following table:

Value	Description
0	Empty: There are no search scope rules.
1	Invalid: The search scope rules have not included any attributes for compilation.
2	Query Expanded: There are not enough search scope rules (fewer than 25) and the compilation rules have been set to conditional compile, so that compilation will not happen.
3	Needs compilation.
4	Compiled.
5	Needs to be recompiled.

2.2.1.5 Compilation Type

Compilation Type: smallint. Identifies the type of search scope compilation. The value MUST be an integer listed in the following table:

Value	Description
0	Search scope will be compiled on demand.

Value	Description
1	Search scope will always be compiled.

2.2.1.6 Crawl Change Status

The Crawl Change Status represents the transfer status for crawl log entry. The value MUST be an integer listed in the following table:

Value	Description
1	Add new entry.
2	Delete entry.
3	Reuse existing entry.
4	Update existing entry.

2.2.1.7 Crawl Log Error Level

The Crawl Log Error Level represents the type of error occurred when crawling the item. The value MUST be an integer listed in the following table:

Value	Description
0	The item is crawled successfully.
1	The item is crawled with warnings.
2	An error occurred when crawling the item.
3	The item is deleted from the **search catalog**.

2.2.1.8 Crawl Log Level

The Crawl Log Level represents the level of the item in the site hierarchy. The value MUST be an integer listed in the following table:

Value	Description
0	Item
1	Folder/Site
2	Top Site

2.2.1.9 DisplayInAdminUI

DisplayInAdminUI: bit. Indicates whether the search scope display group is displayed in the Administration user interface. The value MUST be one of the values listed in the following table:

Value	Description
0	Search scope display group is not displayed in the Administration UI.
1	Search scope display group is displayed in the Administration UI.

2.2.1.10 Filter Wildcard Rules

Filter wildcard rules are used for pattern-matching on property names defined in the metadata schema according to the following rules:

- The "%" character is used as a wildcard that can both begin and end the substring.
- "%%" will match all crawled properties.

2.2.1.11 Keyword Filter Type

Keyword Filter Type specifies which attributes of the keyword are used for filtering the list of keywords**.** The attributes are described in Section 3.1.1.2. The value MUST be an integer listed in the following table:

Value	Description
0	Apply filter to values in Term attribute in Keywords Set.
1	Apply filter to values in Term attribute in Synonyms Set.
2	Apply filter to values in Title attribute in Best Bet Set for entries associated with the keyword.
3	Apply filter to values in URL attribute in Best Bet Set for entries associated with the keyword.
4	Apply filter to values in Contact attribute in Keywords Set.

2.2.1.12 Keyword Type

Keyword Type specifies the type of keywords that the stored procedure operates on or includes in the result set. The attributes are described as part of the Keyword Set definition in Section 3.1.1.2. The value MUST be an integer listed in the following table:

Value	Description
0	All. Include all keywords.
1	Expired. Include keywords with EndDate attribute value earlier than the current date and time.
2	Pending Review. Include keywords with ReviewDate attribute value earlier than the current date and time.

2.2.1.13 Language Resource Type

Language Resource Type identifies the type of a language resource. The value of the Language Resource Type determines the meaning of the Phrase and Mapping elements in the Language Resources Set (section 3.1.1.8). The value is an integer which MUST be one of the values listed in the following table:

Value	Meaning of the Phrase and Mapping elements of the language resource
1	The language resource is part of the block list for pre-query suggestions and **post-query suggestions**. The Phrase element in the Language Resources Set MUST contain a token that is part of the block list. The value of the Mapping element MUST be the empty string.
2	The language resource is part of the always-suggest list for pre-query suggestions and post-query suggestions. The Phrase element MUST contain a token that is part of the always-suggest list. The value of the Mapping element MUST be the empty string.
3	The language resource is part of the block list for spelling suggestions (see [MS-SEARCH], section 2.2.3.1 . The Phrase element MUST contain a token that is part of the block list. The value of the Mapping element MUST be the empty string.
5	The language resource is part of the **Nickname** mappings. The Phrase element MUST contain a token that is a nickname that a person might have. The Mapping element MUST contain a token that might be the same person's actual name.
6	The language resource is a mapping from a person's last name to a phonetic representation of the name. The Phrase element MUST contain a last name. The Mapping element MUST contain a phonetic representation of the same name.
7	The language resource is a mapping from a person's first name to a phonetic representation of the name. The Phrase element MUST contain a first name. The Mapping element MUST contain a phonetic representation of the same name.
8	The language resource is part of the always-suggest list for spelling suggestions (see [MS-SEARCH], section 2.2.3.1. The Phrase element MUST contain a misspelled token for which the spelling correction is the value of the Mapping element.
9	The language resource is a mapping from a single character to a phonetic representation of that character. The Phrase element MUST contain a single character. The Mapping element MUST contain a phonetic representation of that character.

2.2.1.14 Location Type

Location Type: tinyint. Identifies the protocol used to connect to a federated location. The value MUST be an 8-bit integer listed in the following table:

Value	Description
0	Local SharePoint Search
2	Open Search 1.0/1.1. See [OpenSearch1.1-Draft3].
5	MUST be ignored by the client
255	User defined location

2.2.1.15 Managed Type

Managed Type identifies the data type of the managed property. The value is an integer listed in the following table:

Value	Description
1	String which is a Unicode character array of arbitrary length.

Value	Description
2	64 bit integer.
3	64 bit decimal.
4	64 bit **Coordinated Universal Time (UTC)** date/time representing the number of 100-nanosecond intervals after January 1, 1601.
5	A Boolean value, where -1 is TRUE and everything else is FALSE.
6	**Binary large object (BLOB)**.

2.2.1.16 Properties

The list of properties that is returned with each search result, if available, from the federated location.

2.2.1.17 SampleData

It contains the sample data that is used to provide a visual preview of how search results from federated location will look with the given XSL.

2.2.1.18 ScopeFilterBehavior

ScopeFilterBehavior: smallint**.** Identifies how a search scope rule will filter items. The value MUST be an integer listed in the following table:

Value	Description
0	Items matching this rule will be included in the search scope.
1	Items that don't match this rule will be excluded from the search scope.
2	Items matching this rule will be excluded from this search scope.

2.2.1.19 ScopeRuleType

ScopeRuleType: smallint**.** Identifies the type of the search scope rule. The value MUST be an integer listed in the following table:

Value	Description
0	The search scope rule includes all items.
1	The search scope rule includes items whose **folder**, host name, or **subdomain** matches a value specified by the user.
2	The search scope rule includes items which contain a specific value for a specific managed property.

2.2.1.20 Undeletable

Undeletable: bit. Specifies if the search scope display group can be deleted. The value MUST be listed in the following table:

Value	Description
0	Search scope display group can be deleted.
1	Search scope display group cannot be deleted.

2.2.1.21 UrlRuleType

UrlRuleType: smallint. Identifies which part of the item **URL** is matched against a value specified by the user. The value MUST be an integer listed in the following table:

Value	Description
0	folder
1	host name
2	subdomain

2.2.1.22 XSL

The XSL that transforms structured XML search results returned by the federated location into HTML and defines how the search results from the federated location will be displayed. For additional information about the fields, see [MSDN-ESCRXT].

2.2.2 Bit Fields and Flag Structures

None.

2.2.3 Binary Structures

None.

2.2.4 Result Sets

The following subsections define the result sets for this specification

2.2.4.1 Best Bet Result Set

The Best Bet result set returns information about best bets. The result set MUST contain zero or more rows, each corresponding to a single best bet.

The **Transact-Structured Query Language (T-SQL)** syntax for the result set is as follows:

```
BestBetID       int,
Title           nvarchar(100),
Url             nvarchar(2048),
Description     nvarchar(500);
```

BestBetID: The unique identifier of the best bet. This value MUST NOT be NULL.

Title: The title for the best bet. This value can be NULL.

Url: The URL of the best bet. This value MUST NOT be NULL.

Description: The description of the best bet. This value can be NULL.

2.2.4.2 Content Source Log Result Set

The Content Source Log result set returns the crawl statistics for the specified content source from the Crawl History Set (section 3.1.1.3)**.** The result set MUST contain zero or more rows, each corresponding to a single entry in the Crawl History Set. The rows in the result set MUST be sorted by *StartTime* attribute in descending order.

The T-SQL syntax for the result set is as follows:

```
CrawlID             int,
CrawlType           int,
ContentSourceID     int,
Status              int,
StartTime           datetime,
EndTime             datetime,
Success             int,
Warning             int,
Errors              int,
Deletes             int,
NotModified         int,
SecurityUpdates     int,
TopLevelError       int;
```

CrawlID: The unique identifier of the **crawl**.

CrawlType: The type of the crawl. The value MUST be a Crawl Type data type as specified in [MS-SQLPGAT2] section 2.2.1.2.

ContentSourceID: The unique identifier of the content source crawled.

Status: The status of the crawl. The value MUST be a Crawl Status data type as specified in [MS-SQLPGAT2] section 2.2.1.4.

StartTime: The date and time when the crawl started.

EndTime: The date and time when the crawl finished.

Success: The number of items crawled successfully. This value corresponds to the SuccessCount field in the Crawl History Set.

Warning: The number of items crawled with warnings. This value corresponds to the WarningCount field in the Crawl History Set.

Errors: The number of items crawled with errors. This value corresponds to the ErrorCount field in the Crawl History Set.

Deletes: The number of items removed from the search catalog. This value corresponds to the DeleteCount field in the Crawl History Set.

NotModified: The numbers of items processed by the **crawler** that did not result in modifications to the search catalog. This value corresponds to the NotModifiedCount field in the Crawl History Set.

SecurityUpgates: The number of items for which only security descriptors are processed by the crawler.

TopLevelError: The number errors generated when crawling items with LogLevel value set to 2 as specified in section 2.2.1.8. This value corresponds to the LevelHighErrorCount field in the Crawl History Set.

2.2.4.3 Crawled Properties Result Set

The Crawled Properties result set returns information about crawled properties associated with the specified crawled property category**.** The result set MUST contain zero or more rows, each corresponding to a single crawled property.

The T-SQL syntax for the result set is as follows:

```
CategoryName        nvarchar(64),
Propset             uniqueidentifier,
PropertyName        nvarchar(440),
PropertyNameIsEnum  bit,
IsMappedToContent   bit,
IsSampleCacheFull   bit,
VariantType         int,
CrawledPropertyId   int;
```

CategoryName: The name of the crawled property category associated with the crawled property. This value MUST NOT be NULL.

Propset: The **crawled property set identifier** for the crawled property category. This value MUST NOT be NULL.

PropertyName: The name of the crawled property. This value MUST NOT be NULL.

PropertyNameIsEnum: The value MUST be set 1 if the @PropertyName string value was converted from an integer, otherwise 0.

IsMappedToContent: The value MUST be set to 1 if the data type of the crawled property is a string, and data from this crawled property is put in the full-text index catalog. Otherwise, it MUST be set to 0.

IsSampleCacheFull: This value MUST be ignored by the client.

VariantType: An unsigned 16-bit integer that indicates the data type for the crawled property as described in [MS-OAUT]. This value MUST NOT be NULL.

CrawledPropertyId: The unique identifier of the crawled property. This value MUST NOT be NULL.

2.2.4.4 Error Messages Result Set

The Error Messages result set contains information about unique successes, warnings and errors that were encountered during the crawl. The result set MUST contain zero or more rows, each row corresponding to a single entry in the Crawl Error Set as specified in section 3.1.1.3.

The T-SQL syntax for the result set is as follows:

```
ErrorID         int,
ErrorMsg        nvarchar(2000),
Hrresult        int,
ErrorLevel      int;
```

ErrorID: The unique identifier of the error.

ErrorMsg: The error message.

Hrresult: The HRESULT value of the error.

ErrorLevel: The type of the error. The value MUST be a Crawl Log Error Level data type as specified in section 2.2.1.7.

2.2.4.5 Scope Display Groups Result Set

The Scope Display Groups result set returns information about search scope display groups. The result set MUST contain zero or more rows, each row corresponding to a single search scope display group.

The T-SQL syntax for the result set is as follows:

```
DisplayGroupID   int,
Name             nvarchar(60),
Description      nvarchar(300),
ConsumerName     nvarchar(60),
DisplayInAdminUI bit,
Undeletable      bit,
DefaultScopeID   int,
LastModifiedTime datetime,
LastModifiedBy   nvarchar(60);
```

DisplayGroupID: The unique identifier of the search scope display group. This value MUST NOT be NULL.

Name: The name of the search scope display group. This value MUST NOT be NULL.

Description: The description of the search scope display group. This value MUST NOT be NULL.

ConsumerName: The name of the search scope consumer who is the owner of the search scope display group. This value MUST NOT be NULL.

DisplayInAdminUI: The bit flag indicating whether the search scope display group is displayed in the Administration user interface. The value MUST be a data type, as specified in section 2.2.1.9.

Undeletable: The bit flag indicating whether the search scope display group can be deleted. The value MUST be an Undeletable data type as specified in section 2.2.1.20**.**

DefaultScopeID: The unique identifier of the default search scope for the search scope display group. This value MUST NOT be NULL.

LastModifiedTime: The date and time of the last change to the search scope display group. This value MUST NOT be NULL.

LastModifiedBy: The name of the user who last changed the search scope display group. This value MUST NOT be NULL.

2.2.4.6 Scopes Result Set

The Scopes result set returns information about the search scopes. The result set MUST contain zero or more rows, each row corresponding to a single search scope.

The T-SQL syntax for the result set is as follows:

```
ScopeID                     int,
Name                        nvarchar(60),
Description                 nvarchar(300),
ConsumerName                nvarchar(60),
DisplayInAdminUI            bit,
AlternateResultsPageURL     nvarchar(2047),
CompilationType             smallint,
CompilationState            smallint,
LastCompilationTime         datetime,
LastModifiedTime            datetime,
LastModifiedBy              nvarchar(60),
Filter                      nvarchar(2047);
```

ScopeID: The unique identifier of the search scope.

Name: The name of the search scope.

Description: The description of the search scope.

ConsumerName: The name of the search scope consumer who is the owner of the search scope.

DisplayInAdminUI: The bit flag indicating whether the search scope display group is displayed in the Administration user interface. The value MUST be a DisplayInAdminUI data type as specified in section 2.2.1.9**.**

AlternateResultsPageUrl: The alternate results page URL of the search scope. This value can be NULL.

CompilationType: The compilation type of the search scope. The value MUST be a Compilation Type data type, as specified in section 2.2.1.5**.**

CompilationState: The search scope compilation state of the given search scope. The value MUST be a Compilation State data type, as specified in section 2.2.1.4.

LastCompilationTime: The date and time when the search scope was last compiled. This value can be NULL.

LastModifiedTime: The date and time of the last change to the search scope.

LastModifiedBy: The name of the user who last changed the search scope.

Filter: Reserved. This value MUST be ignored by the client.

2.2.4.7 Special Term Result Set

The Special Term result set returns information about the keyword. The result set MUST contain zero or more rows, each corresponding to a single keyword.

The T-SQL syntax for the result set is as follows:

```
SpecialTermId     int,
Term              nvarchar(100),
Definition        nvarchar(500),
Contact           nvarchar(50),
StartDate         datetime,
```

```
EndDate           datetime,
ReviewDate        datetime;
```

SpecialTermId: The unique identifier of the keyword. This value MUST NOT be NULL.

Term: The term for the keyword. This value MUST NOT be NULL.

Definition: The description of the keyword. This value can be NULL.

Contact: The contact name for the keyword. This value can be NULL.

StartDate: The date and time when the keyword begins to appear in search result. This value MUST NOT be NULL.

EndDate: The date and time when the keyword stops appearing in the search result. This value can be NULL.

ReviewDate: The date and time when the keyword is expected to be reviewed. This value can be NULL.

2.2.4.8 Synonym Result Set

The Synonym result set contains information about the keyword synonyms associated with a specified keyword. The result set MUST contain zero or more rows, each corresponding to a single keyword synonym.

The T-SQL syntax for the result set is as follows:

```
Term              nvarchar(100);
```

Term: The term for the keyword synonym. This value MUST NOT be NULL.

2.2.5 Tables and Views

The following subsections define the tables and structures for this specification.

2.2.5.1 MSSQLogSessionSearchCounts

The MSSQLogSessionSearchCounts table stores the values of popular **queries**, specified in section 3.1.1.9, which are used for pre-query suggestions and post-query suggestions.

The T-SQL syntax for the table is as follows:

```
TABLE MSSQLogSessionSearchCounts (
   QueryId          int NULL,
   QueryCount       int NULL,
   TermCount        int NULL,
   QueryTerm        nvarchar(200) NOT NULL,
   QueryTermHash    int NULL,
   IsNew            bit NULL
);
```

QueryId: The identifier associated with the query.

QueryCount: The number of times the query is executed.

TermCount: The number of **tokens** extracted from the **query text**.

QueryTerm: A token extracted from the query text.

QueryTermHash: The identifier of the query term.

IsNew: MUST be set to 1.

2.2.5.2 MSSSessionAcronyms

The MSSSessionAcronyms table stores the values of popular acronyms, specified in section 3.1.1.9, which are used for **pre-query suggestions** and post-query suggestions.

The T-SQL syntax for the table is as follows:

```
TABLE MSSSessionAcronyms (
   Definition       nvarchar(1000) NULL,
   Acronym          nvarchar(50) NULL,
   Count            int NOT NULL,
   TermCount        int NULL,
   AcronymHash      int NULL,
   DefinitionHash   int NULL,
   PropertyStoreId  uniqueidentifier NULL,
   Term             nvarchar(200) NULL,
   TermHash         int NULL,
   AcronymId        int NULL
);
```

Definition: The **extracted definition** for the corresponding **extracted term**, if it is an acronym.

Acronym: The extracted term, if it is an acronym.

Count: The number of occurrences of the acronym.

TermCount: The number of tokens extracted from the definition.

AcronymHash: The identifier for the acronym string.

DefinitionHash: The identifier for the definition string.

PropertyStoreId: A unique identifier associated with the **metadata index**.

Term: A token extracted from the extracted definition.

TermHash: The identifier of the term string.

AcronymId: An identifier associated with the acronym. This is used to indicate if an acronym or its associated token is added to the Popular Acronym Set or the Popular Acronym Term Set, specified in section 3.1.1.9.

2.2.6 XML Structures

The syntax of the definitions in this section uses XML Schema as defined in [XMLSCHEMA1] and [XMLSCHEMA2].

2.2.6.1 Namespaces

This protocol specifies and references **XML namespaces** using the mechanisms specified in [XMLNS]. Although this document associates a **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
xs	http://www.w3.org/2001/XMLSchema	[XMLSCHEMA1] [XMLSCHEMA2]
irm	urn:Microsoft.Search.Ranking.Model.2NN	

2.2.6.2 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
GUIDType	Specifies a **GUID**.
pidType	Specifies a **property identifier**.
HiddenNodesCountType	Specifies a number of hidden nodes in a stage ranking model.
KType	Specifies a saturation parameter.
TransformTypeType	Specifies a type of transform function.
BM25WType	Specifies a weight parameter.
BM25BType	Specifies a length normalization parameter.
LanguageIdentifiersType	Specifies a space-delimited list of **language code identifiers (LCIDs)**.

2.2.6.2.1 GUIDType

Target namespace: urn:Microsoft.Search.Ranking.Model.2NN

Referenced by: RankingModel2NNType, RankingModel2StageType

This type specifies a GUID.

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

2.2.6.2.2 pidType

Target namespace: urn:Microsoft.Search.Ranking.Model.2NN

Referenced by: MinSpanType, StreamLengthType, StaticType, TAUCType, BucketedStaticType, LanguageType, BM25PropertyType

This type specifies a property identifier.

```
<xs:simpleType name="pidType">
  <xs:restriction base="xs:unsignedInt">
    <xs:minInclusive value="1"/>
  </xs:restriction>
</xs:simpleType>
```

2.2.6.2.3 HiddenNodesCountType

Target namespace: urn:Microsoft.Search.Ranking.Model.2NN

Referenced by: HiddenNodesType

This type specifies a number of hidden nodes in a stage ranking model.

```
<xs:simpleType name="HiddenNodesCountType">
  <xs:restriction base="xs:unsignedInt">
    <xs:minInclusive value="1"/>
    <xs:maxInclusive value="8"/>
  </xs:restriction>
</xs:simpleType>
```

2.2.6.2.4 KType

Target namespace: urn:Microsoft.Search.Ranking.Model.2NN

Referenced by: TransformType, BM25Type

This type specifies a saturation parameter.

```
<xs:simpleType name="KType">
  <xs:restriction base="xs:float">
    <xs:minExclusive value="0"/>
  </xs:restriction>
</xs:simpleType>
```

2.2.6.2.5 TransformTypeType

Target namespace: urn:Microsoft.Search.Ranking.Model.2NN

Referenced by: TransformType

This type specifies a type of transform function.

Value	Meaning
Rational	A transform function is rational.
InvRational	A transform function is inverted rational.

Value	Meaning
Linear	A transform function is linear.
Logarithmic	A transform function is logarithmic.

```
<xs:simpleType name="TransformTypeType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Rational"/>
    <xs:enumeration value="InvRational"/>
    <xs:enumeration value="Linear"/>
    <xs:enumeration value="Logarithmic"/>
  </xs:restriction>
</xs:simpleType>
```

2.2.6.2.6 BM25WType

Target namespace: urn:Microsoft.Search.Ranking.Model.2NN

Referenced by: BM25PropertyType

This type specifies a weight parameter.

```
<xs:simpleType name="BM25WType">
  <xs:restriction base="xs:float">
    <xs:minInclusive value="0"/>
  </xs:restriction>
</xs:simpleType>
```

2.2.6.2.7 BM25BType

Target namespace: urn:Microsoft.Search.Ranking.Model.2NN

Referenced by: BM25PropertyType

This type specifies a length normalization parameter.

```
<xs:simpleType name="BM25BType">
  <xs:restriction base="xs:float">
    <xs:minInclusive value="0"/>
  </xs:restriction>
</xs:simpleType>
```

2.2.6.2.8 LanguageIdentifiersType

Referenced by: UserPreferenceType

This type specifies a space delimited list of language code identifiers (LCIDs). The number of LCIDs MUST be greater than 0 and less than 6.

```
 <xs:simpleType name="LanguageIdentifiersType">
    <xs:restriction base="xs:string">
      <xs:pattern
value="((1025|1093|1026|1027|2052|1028|1050|1029|1030|1043|1033|1035|1036|1031|1032|1095|1037
```

```
|1081|1038|1039|1057|1040|1041|1099|1042|1062|1063|1086|1100|1102|1044|1045|1046|2070|1094|10
48|1049|3098|2074|1051|1060|3082|2058|1053|1097|1098|1054|1055|1058|1056|1066)\x20){0,4}(1025
|1093|1026|1027|2052|1028|1050|1029|1030|1043|1033|1035|1036|1031|1032|1095|1037|1081|1038|10
39|1057|1040|1041|1099|1042|1062|1063|1086|1100|1102|1044|1045|1046|2070|1094|1048|1049|3098|
2074|1051|1060|3082|2058|1053|1097|1098|1054|1055|1058|1056|1066)"/>
    </xs:restriction>
  </xs:simpleType>
```

2.2.6.3 Complex Types

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

Complex type	Description
TransformType	Specifies a transform function.
PrecomputeForRangeType	Specifies the recommended interval for an implementation of static ranking ingredient to use for performance optimization.
NormalizeType	Specifies normalization parameters.
WeightsType	Specifies a vector of weights.
ThresholdsType	Specifies a vector of thresholds.
AddsType	Specifies a vector of addition items.
BucketType	Specifies one bucket.
MinSpanType	Specifies parameters for a proximity ranking ingredient.
StreamLengthType	Specifies parameters for a property length ranking ingredient.
StaticType	Specifies parameters for a general static ranking ingredient.
TAUCType	Specifies parameters for an edit distance ranking ingredient.
SocialDistanceType	Specifies parameters for a social distance ranking ingredient.
BucketedStaticType	Specifies parameters for a bucketed static ranking ingredient.
LanguageType	Specifies parameters for a language matching ranking ingredient.
BM25PropertyType	Specifies a property and its parameters for a BM25 ranking ingredient.
BM25PropertiesType	Specifies a set of properties that participate in a BM25 ranking ingredient.
BM25Type	Specifies parameters for a BM25 ranking ingredient.
RankingFeaturesType	Specifies a set of ranking ingredients of various types that form a stage ranking model.
HiddenNodesType	Specifies a configuration of hidden nodes within a stage ranking model.
RankingModel2NNType	Specifies a stage ranking model.
RankingModel2StageType	Specifies a ranking model.

Complex type	Description
UserPreferenceType	Specifies an XML representation of a **search user preference** instance.

2.2.6.3.1 TransformType

Target namespace: urn:Microsoft.Search.Ranking.Model.2NN

Referenced by: MinSpanType, StaticType, TAUCType

This type specifies a transform function.

Attributes:

type : A TransformTypeType attribute that specifies a type of transform function. If the value is "Rational" or "InvRational" then k attribute MUST be presented and attributes a, b, and maxx MUST NOT be presented. If the value is "Linear" then attributes a, b, and maxx MUST be presented and attribute k MUST NOT be presented. If the value is "Logarithmic" then attributes b, and maxx MUST be presented and attributes k and a MUST NOT be presented.

k : A KType attribute that specifies a saturation parameter.

a : A float (as specified in [XMLSCHEMA2]) attribute that specifies a slope parameter.

b : A float (as specified in [XMLSCHEMA2]) attribute that specifies a y-intercept parameter.

maxx : An unsignedInt attribute that specifies a cut-off value.

```
<xs:complexType name="TransformType">
  <xs:attribute name="type" type="irm:TransformTypeType"/>
  <xs:attribute name="k" type="irm:KType" use="optional"/>
  <xs:attribute name="a" type="xs:float" use="optional"/>
  <xs:attribute name="b" type="xs:float" use="optional"/>
  <xs:attribute name="maxx" type="xs:unsignedInt" use="optional"/>
</xs:complexType>
```

2.2.6.3.2 PrecomputeForRangeType

Target namespace: urn:Microsoft.Search.Ranking.Model.2NN

Referenced by: StaticType

This type specifies an interval that implementation of static ranking ingredient is recommended to use for performance optimization. That is, build lookup table static ranking feature to ranking ingredient value in memory and compute ranking ingredient by looking up in that table rather than use full computation.

Attributes:

from : An unsignedInt (as specified in [XMLSCHEMA2]) attribute that specifies a lower inclusive endpoint of the interval.

to : An unsignedInt (as specified in [XMLSCHEMA2]) attribute that specifies an upper exclusive endpoint of the interval.

```
<xs:complexType name="PrecomputeForRangeType">
  <xs:attribute name="from" type="xs:unsignedInt"/>
  <xs:attribute name="to" type="xs:unsignedInt"/>
</xs:complexType>
```

2.2.6.3.3 NormalizeType

Target namespace: urn:Microsoft.Search.Ranking.Model.2NN

Referenced by: MinSpanType, StreamLengthType, StaticType, TAUCType, BM25Type

This type specifies normalization parameters.

Attributes:

SDev : A float (as specified in [XMLSCHEMA2]) attribute that specifies a standard deviation parameter.

Mean : A float (as specified in [XMLSCHEMA2]) attribute that specifies a mean parameter.

```
<xs:complexType name="NormalizeType">
  <xs:attribute name="SDev" type="xs:float"/>
  <xs:attribute name="Mean" type="xs:float"/>
</xs:complexType>
```

2.2.6.3.4 WeightsType

Target namespace: urn:Microsoft.Search.Ranking.Model.2NN

Referenced by: MinSpanType, StreamLengthType, StaticType, TAUCType, BM25Type, HiddenNodesType

This type specifies a vector of weights. The number of child elements MUST be equal to the number of hidden nodes specified for the stage ranking model that includes element of this type.

Child Elements:

Weight : A float (as specified in [XMLSCHEMA2]) element that specifies a weight for one hidden node. The order of these elements corresponds to that of the hidden nodes.

```
<xs:complexType name="WeightsType">
  <xs:sequence>
    <xs:element name="Weight" minOccurs="1" maxOccurs="8" type="xs:float"/>
  </xs:sequence>
</xs:complexType>
```

2.2.6.3.5 ThresholdsType

Target namespace: urn:Microsoft.Search.Ranking.Model.2NN

Referenced by: HiddenNodesType

This type specifies a vector of thresholds. The number of child elements MUST be equal to the number of hidden nodes specified for the stage ranking model that includes element of this type.

Child Elements:

Threshold : A float (as specified in [XMLSCHEMA2]) element that specifies a threshold for one hidden node. The order of these elements corresponds to that of the hidden nodes.

```
<xs:complexType name="ThresholdsType">
  <xs:sequence>
    <xs:element name="Threshold" minOccurs="1" maxOccurs="8" type="xs:float"/>
  </xs:sequence>
</xs:complexType>
```

2.2.6.3.6 AddsType

Target namespace: urn:Microsoft.Search.Ranking.Model.2NN

Referenced by: BucketType

This type specifies a vector of addition items. The number of child elements MUST be equal to the number of hidden nodes specified for the stage ranking model that includes element of this type.

Child Elements:

Add : A float (as specified in [XMLSCHEMA2]) element that specifies an addition item for one hidden node. The order of these elements corresponds to that of the hidden nodes.

```
<xs:complexType name="AddsType">
  <xs:sequence>
    <xs:element name="Add" minOccurs="1" maxOccurs="8" type="xs:float"/>
  </xs:sequence>
</xs:complexType>
```

2.2.6.3.7 BucketType

Target namespace: urn:Microsoft.Search.Ranking.Model.2NN

Referenced by: SocialDistanceType, BucketedStaticType, LanguageType

This type specifies one bucket, that is what additions are to be applied to the hidden nodes when a static ranking feature is equal to the value attribute of this type.

Child Elements:

HiddenNodesAdds : An AddsType element that specifies a vector of addition items for this bucket.

Attributes:

name : A string (as specified in [XMLSCHEMA2]) attribute that specifies a name of the bucket.

value : An unsignedInt (as specified in [XMLSCHEMA2]) attribute that specifies a value of the bucket.

```
<xs:complexType name="BucketType">
  <xs:sequence>
    <xs:element name="HiddenNodesAdds" type="irm:AddsType"/>
  </xs:sequence>
```

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

2.2.6.3.8 MinSpanType

Target namespace: urn:Microsoft.Search.Ranking.Model.2NN

Referenced by: RankingFeaturesType

This type specifies parameters for a proximity ranking ingredient.

Child Elements:

Transform : A TransformType element that specifies a transform function.

Normalize : A NormalizeType element that specifies normalization constants for this ranking ingredient.

Layer1Weights : A WeightsType element that specifies a vector of weights for this ranking ingredient.

Attributes:

name : A string (as specified in [XMLSCHEMA2]) attribute that specifies a name of this ranking ingredient.

pid : A pidType attribute that specifies a property identifier.

default : A float (as specified in [XMLSCHEMA2]) attribute that specifies a default argument for transform function.

maxMinSpan : An unsignedInt (as specified in [XMLSCHEMA2]) attribute that specifies a length of a maximum proximity interval.

isExact : An unsignedInt (as specified in [XMLSCHEMA2]) attribute. MUST be equal to 0 or 1. If not present, the protocol server MUST behave as if the value of 0 was specified. If attribute equals to 1, the ranking ingredient considers only matches with same term order as in original query. If attribute equals to 0, the ranking ingredient considers matches with any query term order.

isDiscounted : An unsignedInt (as specified in [XMLSCHEMA2]) attribute. MUST be equal to 0 or 1. If not present, the protocol server MUST behave as if the value of 0 was specified. If attribute equals to 1, the ranking ingredient considers the number of matches in a stream. If attribute equals to 0, the ranking ingredient does not consider the number of matches in a stream.

```
<xs:complexType name="MinSpanType">
  <xs:all>
    <xs:element name="Transform" type="irm:TransformType"/>
    <xs:element name="Normalize" type="irm:NormalizeType" minOccurs="0"/>
    <xs:element name="Layer1Weights" type="irm:WeightsType"/>
  </xs:all>
  <xs:attribute name="name" type="xs:string" use="optional"/>
  <xs:attribute name="pid" type="irm:pidType" use="required"/>
  <xs:attribute name="default" type="xs:float" use="required"/>
  <xs:attribute name="maxMinSpan" type="xs:unsignedInt" use="required"/>
  <xs:attribute name="isExact" type="xs:unsignedInt" use="optional"/>
```

```
    <xs:attribute name="isDiscounted" type="xs:unsignedInt" use="optional"/>
  </xs:complexType>
```

2.2.6.3.9 StreamLengthType

Target namespace: urn:Microsoft.Search.Ranking.Model.2NN

Referenced by: RankingFeaturesType

This type specifies parameters for a property length ranking ingredient. This ingredient is for the case when the nature of a static ranking feature is a length.

Child Elements:

Normalize : A NormalizeType element that specifies normalization constants for this ranking ingredient.

Layer1Weights : A WeightsType element that specifies a vector of weights for this ranking ingredient.

Attributes:

name : A string (as specified in [XMLSCHEMA2]) attribute that specifies a name of this ranking ingredient.

pid : A pidType attribute that specifies a property identifier of a static ranking feature.

default : An unsignedInt (as specified in [XMLSCHEMA2]) attribute that specifies a default length.

k : A float (as specified in [XMLSCHEMA2]) attribute that specifies a saturation parameter for length transformation.

```
  <xs:complexType name="StreamLengthType">
    <xs:all>
      <xs:element name="Normalize" type="irm:NormalizeType" minOccurs="0"/>
      <xs:element name="Layer1Weights" type="irm:WeightsType"/>
    </xs:all>
    <xs:attribute name="name" type="xs:string" use="optional"/>
    <xs:attribute name="pid" type="irm:pidType" use="required"/>
    <xs:attribute name="default" type="xs:unsignedInt" use="required"/>
    <xs:attribute name="k" type="xs:float" use="required"/>
  </xs:complexType>
```

2.2.6.3.10 StaticType

Target namespace: urn:Microsoft.Search.Ranking.Model.2NN

Referenced by: RankingFeaturesType

This type specifies parameters for a general static ranking ingredient.

Child Elements:

Transform : A TransformType element that specifies a transform function.

Normalize : A NormalizeType element that specifies normalization constants for this ranking ingredient.

Layer1Weights : A WeightsType element that specifies a vector of weights for this ranking ingredient.

PrecomputeForRange : A PrecomputeForRangeType element that specifies an optimization interval.

Attributes:

name : A string (as specified in [XMLSCHEMA2]) attribute that specifies a name of this ranking ingredient.

pid : A pidType attribute that specifies a property identifier of a static ranking feature.

default : An unsignedInt (as specified in [XMLSCHEMA2]) attribute that specifies a default argument for transform function.

```
<xs:complexType name="StaticType">
  <xs:all>
    <xs:element name="Transform" type="irm:TransformType"/>
    <xs:element name="Normalize" type="irm:NormalizeType" minOccurs="0"/>
    <xs:element name="Layer1Weights" type="irm:WeightsType"/>
    <xs:element name="PrecomputeForRange" type="irm:PrecomputeForRangeType" minOccurs="0"/>
  </xs:all>
  <xs:attribute name="name" type="xs:string" use="optional"/>
  <xs:attribute name="pid" type="irm:pidType" use="required"/>
  <xs:attribute name="default" type="xs:unsignedInt" use="required"/>
</xs:complexType>
```

2.2.6.3.11 TAUCType

Target namespace: urn:Microsoft.Search.Ranking.Model.2NN

Referenced by: RankingFeaturesType

This type specifies parameters for an edit distance ranking ingredient.

Child Elements:

Transform : A TransformType element that specifies a transform function.

Normalize : A NormalizeType element that specifies normalization constants for this ranking ingredient.

Layer1Weights : A WeightsType element that specifies a vector of weights for this ranking ingredient.

Attributes:

name : A string (as specified in [XMLSCHEMA2]) attribute that specifies a name of this ranking ingredient.

pid : A pidType attribute that specifies a property identifier.

default : A float (as specified in [XMLSCHEMA2]) attribute that specifies a default argument for transform function.

count : An unsignedInt (as specified in [XMLSCHEMA2]) attribute that specifies a number of consecutive property identifiers starting from pid and going up.

weightins : A float (as specified in [XMLSCHEMA2]) attribute that specifies a weight of insertion for edit distance.

weightdel : A float (as specified in [XMLSCHEMA2]) attribute that specifies a weight of deletion for edit distance.

```
<xs:complexType name="TAUCType">
  <xs:all>
    <xs:element name="Transform" type="irm:TransformType"/>
    <xs:element name="Normalize" type="irm:NormalizeType" minOccurs="0"/>
    <xs:element name="Layer1Weights" type="irm:WeightsType"/>
  </xs:all>
  <xs:attribute name="name" type="xs:string" use="optional"/>
  <xs:attribute name="pid" type="irm:pidType" use="required"/>
  <xs:attribute name="default" type="xs:float" use="required"/>
  <xs:attribute name="count" type="xs:unsignedInt" use="required"/>
  <xs:attribute name="weightins" type="xs:float" use="required"/>
  <xs:attribute name="weightdel" type="xs:float" use="required"/>
</xs:complexType>
```

2.2.6.3.12 SocialDistanceType

Target namespace: urn:Microsoft.Search.Ranking.Model.2NN

Referenced by: RankingFeaturesType

This type specifies parameters for a social distance ranking ingredient, here the social distance is a distance from a user to a crawled item.

Child Elements:

Bucket : A BucketType element that specifies a default bucket value. There MUST be only Bucket elements with value attribute listed in the following table:

Value	Meaning
0	Social distance is undefined
1	Social distance is 1
2	Social distance is 2

Attributes:

name : A string (as specified in [XMLSCHEMA2]) attribute that specifies a name of this ranking ingredient.

```
<xs:complexType name="SocialDistanceType">
  <xs:sequence>
    <xs:element name="Bucket" type="irm:BucketType" minOccurs="1" maxOccurs="unbounded"/>
```

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

2.2.6.3.13 BucketedStaticType

Target namespace: urn:Microsoft.Search.Ranking.Model.2NN

Referenced by: RankingFeaturesType

This type specifies parameters for a bucketed static ranking ingredient. This ingredient is for the case when the nature of a static ranking feature is a finite set of values, specified in Bucket child elements.

Child Elements:

Bucket : A BucketType element that specifies one bucket.

Attributes:

name : A string (as specified in [XMLSCHEMA2]) attribute that specifies a name of this ranking ingredient.

pid : A pidType attribute that specifies a property identifier of a static ranking feature.

default : An unsignedInt (as specified in [XMLSCHEMA2]) attribute that specifies a default bucket value. MUST be equal to the value attribute of one of the Bucket child elements.

```
<xs:complexType name="BucketedStaticType">
  <xs:sequence>
    <xs:element name="Bucket" type="irm:BucketType" minOccurs="1" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="name" type="xs:string" use="optional"/>
  <xs:attribute name="pid" type="irm:pidType" use="required"/>
  <xs:attribute name="default" type="xs:unsignedInt" use="required"/>
</xs:complexType>
```

2.2.6.3.14 LanguageType

Target namespace: urn:Microsoft.Search.Ranking.Model.2NN

Referenced by: RankingFeaturesType

This type specifies parameters for a language matching ranking ingredient. This ingredient is for the case when the nature of a static ranking feature is a language identifier.

Child Elements:

Bucket : A BucketType element that specifies one bucket. There MUST be only Bucket elements with value attribute equal to 0 or 1. Value 0 of the Bucket corresponds to the case when language of the query does not match a language stored in a static ranking feature. Value 1 corresponds to the case when languages math.

Attributes:

name : A string (as specified in [XMLSCHEMA2]) attribute that specifies a name of this ranking ingredient.

pid : A pidType attribute that specifies a property identifier.

default : An unsignedInt (as specified in [XMLSCHEMA2]) attribute that specifies a default bucket value. MUST be equal to the value attribute of one of the Bucket child elements. MUST be one of the values listed in the following table:

Value	Meaning
0	Languages do not match.
1	Languages match.

```
<xs:complexType name="LanguageType">
  <xs:sequence>
    <xs:element name="Bucket" type="irm:BucketType" minOccurs="1" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="name" type="xs:string" use="optional"/>
  <xs:attribute name="pid" type="irm:pidType" use="required"/>
  <xs:attribute name="default" type="xs:unsignedInt" use="required"/>
</xs:complexType>
```

2.2.6.3.15 BM25PropertyType

Target namespace: urn:Microsoft.Search.Ranking.Model.2NN

Referenced by: BM25PropertiesType

This type specifies a property and its parameters for BM25 ranking ingredient.

Attributes:

name : A string (as specified in [XMLSCHEMA2]) attribute that specifies a name of this property.

pid : A pidType attribute that specifies a property identifier.

w : A BM25WType attribute that specifies a weight parameter.

b : A BM25BType attribute that specifies a length normalization parameter.

inanchor : A Boolean (as specified in [XMLSCHEMA2]) attribute that specifies whether this property applies to the anchor text catalog, as specified in [MS-CIFO] section 2.18.2. The value MUST be "true" if the property applies to the anchor text catalog, otherwise it MUST be "false".

extractOccurrence : A Boolean (as specified in [XMLSCHEMA2]) attribute that specifies whether this property is used in either TAUC or MinSpan elements within the same stage ranking model that element of this type belongs to. The value MUST be "true" if the property is used in either TAUC or MinSpan elements, otherwise it MUST be "false".

```
<xs:complexType name="BM25PropertyType">
  <xs:attribute name="name" type="xs:string" use="optional"/>
  <xs:attribute name="pid" type="irm:pidType" use="required"/>
  <xs:attribute name="w" type="irm:BM25WType" use="required"/>
  <xs:attribute name="b" type="irm:BM25BType" use="required"/>
  <xs:attribute name="inanchor" type="xs:boolean" use="optional"/>
```

```
  <xs:attribute name="extractOccurrence" type="xs:boolean" use="optional"/>
</xs:complexType>
```

2.2.6.3.16 BM25PropertiesType

Target namespace: urn:Microsoft.Search.Ranking.Model.2NN

Referenced by: BM25Type

This type specifies a set of properties that participate in BM25 ranking ingredient.

Child Elements:

Property : A BM25PropertyType element that specifies one property.

```
<xs:complexType name="BM25PropertiesType">
  <xs:sequence>
    <xs:element name="Property" type="irm:BM25PropertyType" minOccurs="1"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

2.2.6.3.17 BM25Type

Target namespace: urn:Microsoft.Search.Ranking.Model.2NN

Referenced by: RankingFeaturesType

This type specifies parameters for BM25 ranking ingredient.

Child Elements:

Normalize : A NormalizeType element that specifies normalization constants for this ranking ingredient.

Layer1Weights : A WeightsType element that specifies a vector of weights for this ranking ingredient.

Properties : A BM25PropertiesType element that specifies a set of properties that participate in this ranking ingredient.

Attributes:

name : A string (as specified in [XMLSCHEMA2]) attribute that specifies a name of this ranking ingredient.

k1 : A KType attribute that specifies k1 parameter.

```
<xs:complexType name="BM25Type">
  <xs:all>
    <xs:element name="Normalize" type="irm:NormalizeType" minOccurs="0"/>
    <xs:element name="Layer1Weights" type="irm:WeightsType" minOccurs="0"/>
    <xs:element name="Properties" type="irm:BM25PropertiesType"/>
  </xs:all>
  <xs:attribute name="name" type="xs:string" use="optional"/>
```

```
    <xs:attribute name="k1" type="irm:KType" use="required"/>
  </xs:complexType>
```

2.2.6.3.18 RankingFeaturesType

Target namespace: urn:Microsoft.Search.Ranking.Model.2NN

Referenced by: RankingModel2NNType

This type specifies a set of ranking ingredients of various types that form a stage ranking model.

Child Elements:

BM25Main : A BM25Type element that specifies a BM25 ranking ingredient.

BucketedStatic : A BucketedStaticType element that specifies a bucketed static ranking ingredient.

Static : A StaticType element that specifies a general static ranking ingredient.

SocialDistance : A SocialDistanceType element that specifies a social distance ranking ingredient.

TAUC : A TAUCType element that specifies an edit distance ranking ingredient.

MinSpan : A MinSpanType element that specifies a proximity ranking ingredient.

Language : A LanguageType element that specifies a language matching ranking ingredient.

StreamLength : A StreamLengthType element that specifies a property length ranking ingredient.

```
<xs:complexType name="RankingFeaturesType">
  <xs:sequence>
    <xs:element name="BM25Main" type="irm:BM25Type"/>
    <xs:choice minOccurs="0" maxOccurs="unbounded">
      <xs:element name="BucketedStatic" type="irm:BucketedStaticType"/>
      <xs:element name="Static" type="irm:StaticType"/>
      <xs:element name="SocialDistance" type="irm:SocialDistanceType"/>
      <xs:element name="TAUC" type="irm:TAUCType"/>
      <xs:element name="MinSpan" type="irm:MinSpanType"/>
      <xs:element name="Language" type="irm:LanguageType"/>
      <xs:element name="StreamLength" type="irm:StreamLengthType"/>
    </xs:choice>
  </xs:sequence>
</xs:complexType>
```

2.2.6.3.19 HiddenNodesType

Target namespace: urn:Microsoft.Search.Ranking.Model.2NN

Referenced by: RankingModel2NNType

This type specifies a configuration of hidden nodes within a stage ranking model.

Child Elements:

Thresholds : A ThresholdsType element that specifies a vector of thresholds for hidden nodes.

Layer2Weights : A WeightsType element that specifies a vector of weights for hidden nodes.

Attributes:

count : A HiddenNodesCountType attribute that specifies a number of hidden nodes.

```
<xs:complexType name="HiddenNodesType">
  <xs:all>
    <xs:element name="Thresholds" type="irm:ThresholdsType"/>
    <xs:element name="Layer2Weights" type="irm:WeightsType"/>
  </xs:all>
  <xs:attribute name="count" type="irm:HiddenNodesCountType" use="required"/>
</xs:complexType>
```

2.2.6.3.20 RankingModel2NNType

Target namespace: urn:Microsoft.Search.Ranking.Model.2NN

Referenced by: RankingModel2StageType

This type specifies a stage ranking model.

Child Elements:

HiddenNodes : A HiddenNodesType element that specifies a configuration of hidden nodes.

RankingFeatures : A RankingFeaturesType element that specifies a set of ranking ingredients that form a stage ranking model.

Attributes:

id : A GUIDType attribute that specifies a unique identifier of a stage ranking model.

maxStageWidCount : An unsignedInt (as specified in [XMLSCHEMA2]) attribute that specifies a maximum number of crawled items this stage ranking model to be applied to.

```
<xs:complexType name="RankingModel2NNType">
  <xs:sequence>
    <xs:element name="HiddenNodes" type="irm:HiddenNodesType"/>
    <xs:element name="RankingFeatures" type="irm:RankingFeaturesType"/>
  </xs:sequence>
  <xs:attribute name="id" type="irm:GUIDType" use="required"/>
  <xs:attribute name="maxStageWidCount" type="xs:unsignedInt" use="optional"/>
</xs:complexType>
```

2.2.6.3.21 RankingModel2StageType

Target namespace: urn:Microsoft.Search.Ranking.Model.2NN

Referenced by: RankingModel2Stage

This type specifies a ranking model.

Child Elements:

RankingModel2NN : A RankingModel2NNType element that specifies a stage ranking model.

<xs:any> : Any number of elements from other than a target namespace with information that the protocol client implementation wants to keep together with the ranking model. The choice of any particular information is implementation-specific for the protocol client and not significant for interoperability.

Attributes:

id : A GUIDType attribute that specifies a unique identifier of a ranking model.

name : A string (as specified in [XMLSCHEMA2]) attribute that specifies a name of a ranking model.

description : A string (as specified in [XMLSCHEMA2]) attribute that specifies a description of a ranking model.

```
<xs:complexType name="RankingModel2StageType">
  <xs:sequence>
    <xs:element name="RankingModel2NN" type="irm:RankingModel2NNType" minOccurs="1"
maxOccurs="2"/>
    <xs:any namespace="##other" processContents="skip" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="id" type="irm:GUIDType" use="required"/>
  <xs:attribute name="name" type="xs:string" use="optional"/>
  <xs:attribute name="description" type="xs:string" use="optional"/>
</xs:complexType>
```

2.2.6.3.22 UserPreferenceType

Referenced by: UserPreference

This type specifies a XML representation of a search user preference instance.

Attributes:

f: An unsignedLong (as specified in [XMLSCHEMA2]) attribute that specifies whether pre-query suggestions feature is enabled. MUST be one of the following values when present:

Value	Meaning
0	pre-query suggestions feature is disabled
1	pre-query suggestions feature is enabled

If not present, the protocol server MUST behave as if the value of 1 was specified.

l: A LanguageIdentifiersType attribute that ,when present, specifies a list of **search query** languages from which the user can choose the language of each search query. If not present, the protocol server MUST assume that the user has not specified a search query language and a default language MUST be used instead.

c: A unsignedLong (as specified in [XMLSCHEMA2]) attribute that is reserved for the protocol client. The protocol server MUST not interpret this attribute. The protocol server MUST not modify this attribute.

s: A string (as specified in [XMLSCHEMA2]) attribute that is reserved for the protocol client. The protocol server MUST not interpret this attribute. The protocol server MUST not modify this attribute.

v: A int (as specified in [XMLSCHEMA2]) attribute that holds the server configuration version of the search user preference instance at the time of the most recent retrieval or update. This attribute MUST be present and MUST contain a value greater than or equal to -1.

```
<xs:complexType name="UserPreferenceType">
  <xs:attribute name="f" type="xs:unsignedLong" use="optional"/>
  <xs:attribute name="l" type="LanguageIdentifiersType " use="optional"/>
  <xs:attribute name="c" type="xs:unsignedLong" use="optional"/>
  <xs:attribute name="s" type="xs:string" use="optional"/>
  <xs:attribute name="v" type="xs:int" use="required"/>
</xs:complexType>
```

2.2.6.4 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
RankingModel2Stage	Specifies a ranking model.
UserPreference	Specifies a search user preference.

2.2.6.4.1 RankingModel2Stage

Target namespace: urn:Microsoft.Search.Ranking.Model.2NN

A RankingModel2StageType element that specifies a ranking model.

```
<xs:element name="RankingModel2Stage" type="irm:RankingModel2StageType"/>
```

2.2.6.4.2 UserPreference

An UserPreferenceType element that specifies a search user preference.

<xs:element name="u" type="UserPreferenceType"/>

2.2.6.5 Attributes

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

2.2.6.6 Groups

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

2.2.6.7 Attribute Groups

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

3 Protocol Details

3.1 Server Details

The Microsoft® SharePoint® 2010 Products and Technologies role is described in this section. The SharePoint 2010 Products role serves requests for common search administration tasks.

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 Metadata Schema

The following diagram describes the abstract data model for the metadata schema. 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.

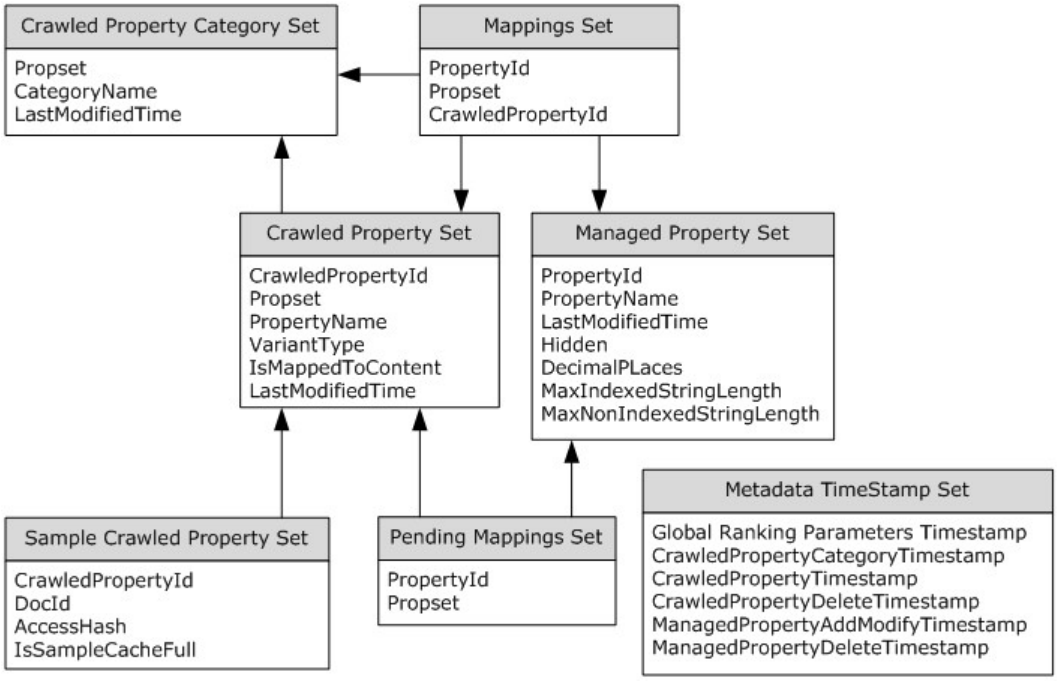

Figure 2: Metadata Schema Abstract Data Model

Crawled Property Category Set: A collection of entries corresponding to the crawled property categories defined in the metadata schema. Each entry MUST be uniquely identified by its Propset, and it MUST include the following elements:

- **Propset**: The crawled property set identifier associated with the crawled property category**.**
- **CategoryName**: The name for the crawled property category**.**

- **LastModifiedTime** : The date and time the crawled property was last changed. The value MUST be stored in the local time of the server.

Crawled Property Set: A collection of entries corresponding to the crawled properties defined in the metadata schema. Each entry MUST be uniquely identified by its CrawledPropertyId, and it MUST include the following elements:

- **CrawledPropertyId**: The unique identifier of the crawled property.
- **Propset**: The reference to the corresponding crawled property category.
- **PropertyName**: The name for the crawled property.
- **VariantType**: The **variant type (2)** for the crawled property.
- **IsMappedToContent**: The flag indicating whether the crawled property data is written to the full-text index catalog.
- **LastModifiedTime** : The date and time the crawled property was last changed. The value MUST be stored in the local time of the server.

Managed Property Set: A collection of entries corresponding to the managed properties defined in the metadata schema**.** Each entry MUST be uniquely identified by its PropertyId, and it MUST include the following elements:

- **PropertyId**: The unique identifier of the managed property**.**
- **PropertyName**: The name for the managed property**.**
- **LastModifiedTime**: The date and time the managed property was last changed. The value MUST be stored in the local time of the server.
- **Hidden**: The flag indicating whether the managed property is hidden in the administration user interface**.**
- **DecimalPlaces:** The number of floating point decimal places that must be honored in the metadata index.
- **MaxIndexedStringLength:** The maximum number of characters persisted in the strval column in the *MSSDocProps* table defined in [MS-SQLPQ2] section 2.2.5.2.
- **MaxNonIndexedStringLength:** The maximum number of characters persisted in the strval2 column in the *MSSDocProps* table defined in [MS-SQLPQ2] section 2.2.5.2.

Mappings Set: A collection of entries each describing the mapping of crawled property to managed property**.** Each entry MUST include the following elements:

- **PropertyId**: The reference to the managed property**.**
- **Propset**: The reference to the crawled property category.
- **CrawledPropertyId**: The reference to the crawled property.

Pending Mappings Set: A collection of entries each describing the mapping of crawled property to managed property that is not yet persisted in the Mappings Set. Each entry MUST include the following elements:

- **PropertyId**: The reference to the managed property**.**

- **Propset**: The reference to the crawled property category.

Sample Crawled Property Set: A collection of relationships between crawled properties and sample items that contain them. Each entry MUST be uniquely identified by the combination of its CrawledPropertyId and DocId. Each entry MUST include the following elements:

- **CrawledPropertyId**: The reference to the crawled property**.**
- **DocId**: The unique identifier of the item.
- **AccessHash**: The identifier of the item access URL.
- **IsSampleCacheFull**: A bit flag indicating whether the Sample Crawled Properties Set is complete and no more entries needed for the crawled property.

Metadata Timestamp Set: Used to track date and time of the last modification to the metadata schema entities. Metadata Timestamp Set MUST include exactly one entry. The entry MUST include the following elements:

- **Global Ranking Parameters Timestamp**: The date and time of the last change to the Global Ranking Parameters Set as specified in section 3.1.1.6.
- **CrawledPropertyCategoryTimestamp**: The date and time of the last change to the Crawled Property Category Set. The value MUST be stored in the local time of the server.
- **CrawledPropertyTimestamp**: The date and time of the last change to the Crawled Property Set**.** The value MUST be stored in the local time of the server.
- **CrawledPropertyDeleteTimestamp**: The date and time when an entry in the Crawled Property Set was last deleted. The value MUST be stored in the local time of the server.
- **ManagedPropertyAddModifyTimestamp**: The date and time when an entry in the Managed Property Set was last added or changed. The value MUST be stored in the local time of the server.
- **ManagedPropertyDeleteTimestamp**: The date and time when an entry in the Managed Property Set was last deleted. The value MUST be stored in the local time of the server.

3.1.1.2 Best Bets and Keywords

The following diagram in describes the abstract data model for best bets and keywords. 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.

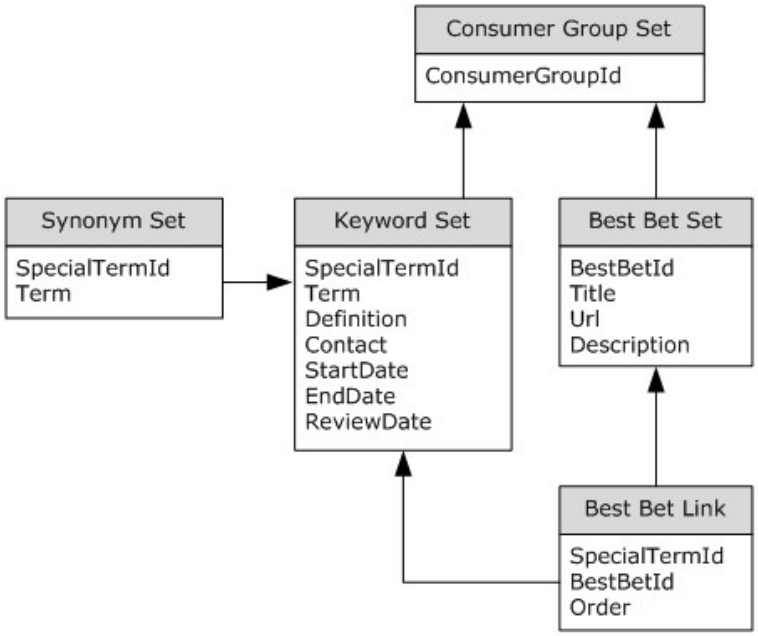

Figure 3: Best Bets and Keywords Abstract Data Model

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.

There MUST NOT be two best bet entries with the same URL associated with the same keyword consumer.

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**.**

Best Bet Link Set: A collection of entries representing associations between best bets and keywords. Each entry MUST include the following elements:

- **SpecialTermId**: The unique identifier of the keyword associated with the best bet.
- **BestBetId**: The unique identifier of the best bet**.**
- **Order:** The order of the best bet among all best bets for the corresponding keyword.

3.1.1.3 Crawl Log

The following diagram describes the abstract data model for crawl log. 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.

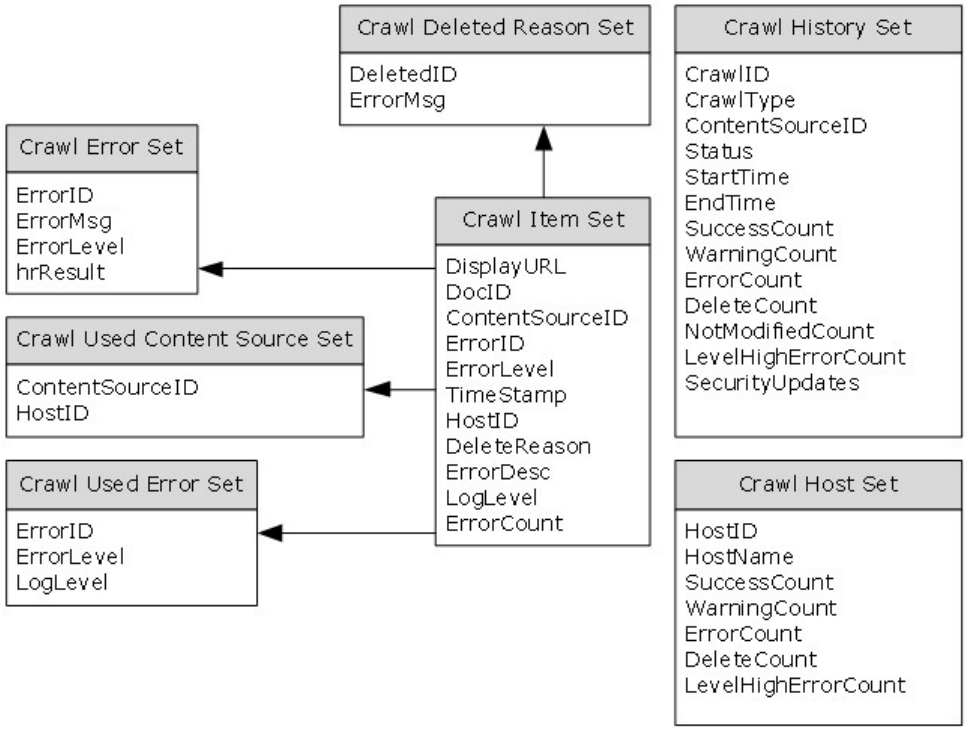

Figure 4: Crawl Log Data Flow Diagram

Crawl Error Set: A collection of entries corresponding to the errors that can occur during the crawl. Each entry MUST be uniquely identified by its ErrorID and MUST include the following elements:

- **ErrorID:** The unique identifier of the error.
- **ErrorMsg:** The error message.
- **ErrorLevel:** The type of the error. The value MUST be a Crawl Log Error Level data type as specified in section 2.2.1.7.
- **hrResult:** The HRESULT value of the error.

Crawl Used Error Set: A collection of entries corresponding to successes, warnings and errors that occurred during the crawl and which are referenced by Crawl Item Set. Each entry MUST be uniquely identified by its ErrorID and MUST include the following elements:

- **ErrorID:** The unique identifier of the issue.
- **ErrorLevel:** The type of the error. The value MUST be a Crawl Log Error Level data type as specified in section 2.2.1.7
- **LogLevel:** The level of the item in the **site** hierarchy. The value MUST be a Crawl Log Level data type as specified in section 2.2.1.8

Crawl Deleted Reason Set: A collection of entries corresponding to the error messages that describe various reasons why items can be removed from the search catalog during the crawl. Each entry MUST include the following elements:

- **DeletedID:** The unique identifier of the error.
- **ErrorMsg:** The message describing the reason for deleting the item from the search catalog.

Crawl Item Set: A periodically archived collection of the crawl information corresponding to the entries in the Crawl Url History set as described in [MS-SQLPGAT2] section 3.1.1.3. Each entry in the Crawl Item Set represents an item processed by the crawler and contains a subset of the crawl Url attributes from the Crawl Url History set necessary to support crawl log functionality.

Each entry MUST be identified by its DocID and it MUST include the following elements:

- **DocID:** The integer identifier of the item.
- **ContentSourceID:** The unique identifier of the content source this item is associated with.
- **DisplayURL**: The URL of the item.
- **ErrorID**: The unique identifier of the error occurred when crawling the item. The value MUST be one of the values listed in the Crawl Error Set.
- **ErrorLevel**: The type of error occurred when crawling the item. The value MUST be a Crawl Log Error Level data type as specified in section 2.2.1.7.
- **TimeStamp**: The **UTC** date and time when the item was last crawled.
- **HostID:** The unique identifier of the host this item is associated with.
- **DeleteReason:** The unique identifier of the reason for removing the item from the search catalog. The value MUST be one of the values listed in the Crawl Deleted Reason Set**.**

- **ErrorDesc:** The description of the error occurred when crawling the item.
- **LogLevel:** The level of the item in the site hierarchy. The value MUST be a Crawl Log Level data type as specified in section 2.2.1.8.
- **ErrorCount:** The number of errors occurred when crawling the item.

Crawl Used Content Source Set: A collection of entries corresponding to content sources and hosts that were crawled and which are referenced by Crawl Item Set. Each entry MUST include the following elements:

- **ContentSourceID:** The unique identifier of the content source.
- **HostID:** The unique identifier of the host.

Crawl Host Set: A collection of entries corresponding to the crawl statistics for each host processed by the crawler. Each entry represents aggregated data for a host. It MUST be uniquely identified by its HostID and it MUST include the following elements:

- **HostID:** The unique identifier of the host.
- **HostName:** The host name.
- **SuccessCount:** The number of items crawled successfully.
- **WarningCount:** The number of items crawled with warnings.
- **ErrorCount:** The number of items crawled with errors.
- **DeleteCount:** The number of items removed from the search catalog.
- **LevelHighErrorCount:** The number errors generated when crawling items with LogLevel value set to 2.

Crawl History Set: A collection of entries representing the statistics for each crawl. Each entry MUST be uniquely identified by its CrawlID and it MUST include the following elements:

- **CrawlID:** The unique identifier of the crawl.
- **ContentSourceID:** The unique identifier of the content source crawled.
- **CrawlType:** The type of the crawl. The value MUST be a Crawl Type data type as specified in [MS-SQLPGAT2] section 2.2.1.2.
- **Status:** The status of the crawl. The value MUST be a Crawl Status data type as specified in [MS-SQLPGAT2] section 2.2.1.4.
- **StartTime:** The date and time when crawl started.
- **EndTime:** The date and time when the crawl finished.
- **SuccessCount:** The number of items crawled successfully.
- **WarningCount:** The number of items crawled with warnings.
- **ErrorCount:** The number of items crawled with errors.
- **DeleteCount:** The number of items removed from the search catalog.

- **NotModifiedCount:** The numbers of items processed by the crawler that did not result in modifications to the search catalog.
- **LevelHighErrorCount:** The number errors generated when crawling items with LogLevel value set to 2.
- **SecurityUpdates:** The number of items for which only security descriptors are processed by the crawler.

3.1.1.4 Scopes

The following diagram describes the abstract data model for search scopes. 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.

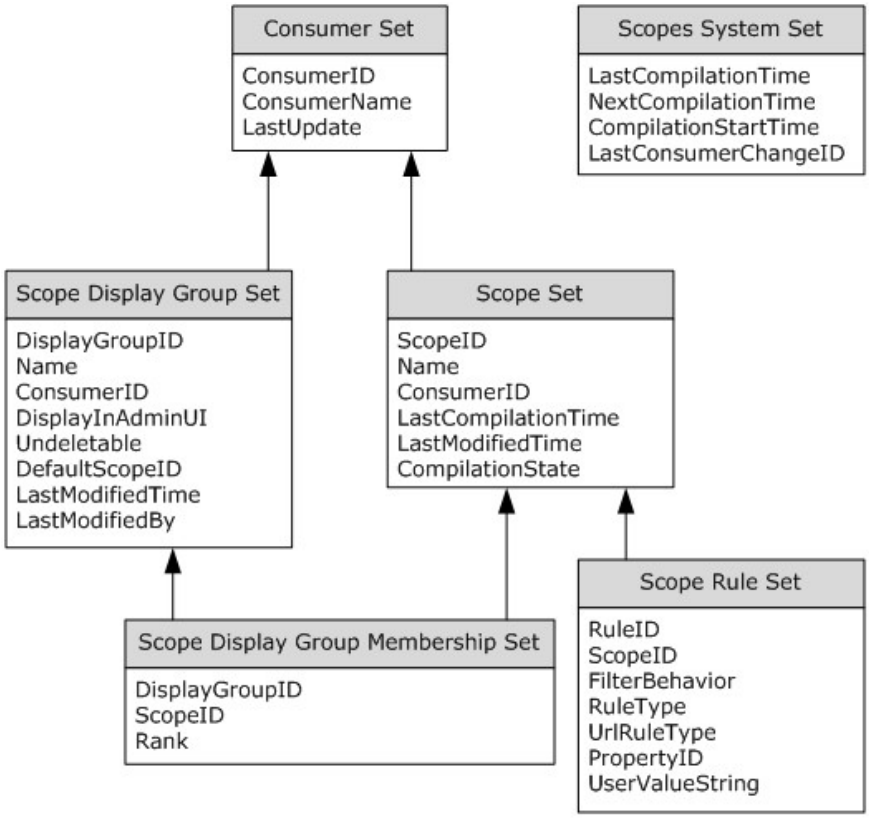

Figure 5: Search Scopes Abstract Data Model

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

- **ConsumerID:** The unique identifier of the search scope consumer.
- **ConsumerName:** The name for the search scope consumer.
- **LastUpdate:** Represents a version. When any change is made to the search scope display group, search scopes or search scope rules of the search scopes owned by the specific search scope

consumer this attribute is set to the value of the LastConsumerChangeID attribute of the Scope System Set indicating that the specific search scope consumer has changed

Scope Set: A collection of entries corresponding to search scopes. Each entry MUST be uniquely identified by its ScopeID and it MUST include the following elements:

- **ScopeID:** The unique identifier of the search scope.
- **Name:** The name for the search scope.
- **ConsumerID:** The reference to the corresponding search scope consumer.
- **LastCompilationTime:** The date and time when the search scope was last compiled. The value MUST be stored in the local time of the server.
- **LastModifiedTime:** The date and time when the search scope was last changed. The value MUST be stored in the local time of the server.
- **CompilationState:** The compilation state of the search scope. The value MUST be a Compilation State data type defined in Section 2.2.1.4.

Scope Rule Set: A collection of entries corresponding to search scope rules. Each entry MUST be uniquely identified by its RuleID and it MUST include the following elements:

- **RuleID:** The unique identifier of the search scope rule.
- **ScopeID:** The reference to the corresponding search scope.
- **FilterBehavior**: The value MUST be a ScopeFilterBehavior data type as specified in section 2.2.1.18.
- **RuleType**: The type of search scope rule which MUST be a ScopeRuleType data as specified in section 2.2.1.19. This attribute specifies which items are included in the scope. If the value of this attribute is set to 1, the search scope rule MUST include items whose folder, host name, or subdomain matches a value specified by UserValueString attribute. If the value of this attribute is set to 2 the search scope rule MUST include items whose managed property value matches the value of UserValueString attribute and whose managed property identifier matches the value of PropertyId attribute.
- **UrlRuleType**: Identifies which part of the item URL is matched against a value specified by UserValueString attribute. The value MUST be a UrlRuleType data as specified in section 2.2.1.21.
- **PropertyID**: The property identifier, which is a reference to a managed property.
- **UserValueString**: The search scope rule value to use in the search scope rule.

Scope Display Group Set: A collection of entries corresponding to search scope display groups. Each entry MUST be uniquely identified by its DisplayGroupID and it MUST include the following elements:

- **DisplayGroupID:** The unique identifier of the search scope display group.
- **Name:** The name for the search scope display group.
- **Description:** The description of the search scope display group.
- **ConsumerID:** The reference to the corresponding search scope consumer.

- **DisplayInAdminUI:** The flag indicating whether the search scope display group is displayed in the Administrator user interface.
- **Undeletable:** The flag indicating whether the search scope display group can be deleted.
- **DefaultScopeID:** The reference to the corresponding default search scope.
- **LastModifiedTime:** The date and time when the search scope display group was last changed. The value MUST be stored in the local time of the server.
- **LastModifiedBy:** The name of the user who last changed the search scope display group.

Scope Display Group Membership Set: A collection of relationships between search scope display groups and the search scopes they contain. Each entry MUST be uniquely identified by the combination of its DisplayGroupID and ScopeID. Each entry MUST include the following elements:

- **DisplayGroupID:** The reference to the search scope display group.
- **ScopeID:** The reference to the search scope.
- **Rank:** The identifier of the search scope rank within the search scope display group.

Scopes System Set: Used to track date and time and version of the last change to search scope entities. Scopes System Set MUST include exactly one entry. The entry MUST include the following elements:

- **LastCompilationTime:** The date and time when search scopes in the Scope Set were last compiled. The value MUST be stored in the local time of the server.
- **NextCompilationTime:** The date and time when search scopes in the Scope Set will be compiled next. The value MUST be stored in the local time of the server.
- **CompilationStartTime**: The date and time when the compilation of search scopes was started for the current compilation. The value MUST be stored in the local time of the server.
- **LastConsumerChangeID:** Represents a version. When any change is made to the search scope display groups, search scopes or search scope rules of any search scope owned by any search scope consumer, this value is incremented by 1, indicating that a search scope consumer has changed.

3.1.1.5 Ranking Model

The ranking model is comprised of the Ranking Model and Static Ranking Features Set.

3.1.1.5.1 Ranking Model Set

The ranking model set contains ranking models. Each entry MUST contain the following attributes:

- A GUID **identifier** for the ranking model. The identifier MUST be unique among all ranking models in the ranking model set.
- A flag indicating whether the ranking model is the **default** ranking model. There MUST be only one default ranking model in the ranking model set.
- A string representation of the ranking model**'s weights and parameters**.

3.1.1.5.2 Static Ranking Features Set

The static ranking features set contains static ranking features. Each entry MUST contain the following attributes:

- A property identifier
- A numeric **default value**
- A **modifier** flag for the static ranking feature value.
- A flag indicating whether the static ranking feature **applies to the anchor text catalog**, as specified in [MS-CIFO] section 2.18.2.

3.1.1.6 Ranking Parameters

The **Authority Pages Set** is a list of authority pages which are used in page ranking. Each entry MUST include the following elements:

- The valid URL of the authority page.
- The **authority level** of the authority page.

The Global Ranking Parameters Set is a list of name/value pairs for variables used in ranking. Each entry MUST include the following elements:

- The name of the variable
- The value for the variable.

3.1.1.7 Federated Search

The following diagram describes the abstract data model for federation. 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.

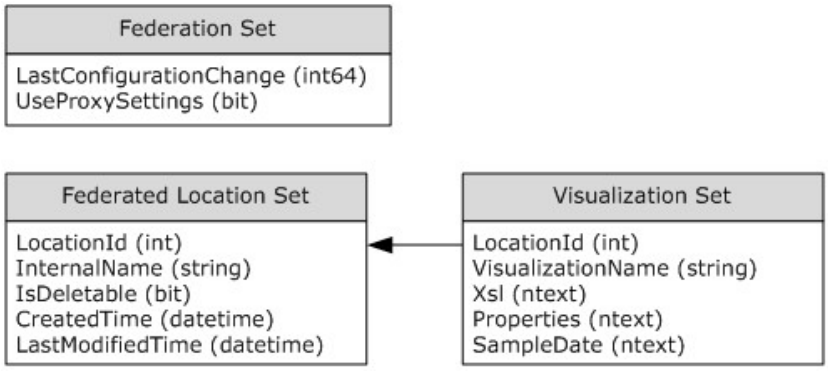

Figure 6: Federated Search Data Flow Diagram

Federation Set: Used to track date and time and version of the last change to federated location and visualization entities. The Federation Set MUST include exactly one entry. The entry MUST include the following elements:

- **LastConfigurationChange:** The version.

- **UseProxySettings:** A 1-bit number that indicates whether federated locations are configured to use a **proxy** when retrieving search results.

Federated Location Set: A collection of entries corresponding to federated locations. Each entry MUST be uniquely identified by its LocationID and it MUST include the following elements:

- **LocationID:** The unique identifier of the federated location.
- **InternalName:** The unique internal name of the federated location.
- **IsDeletable:** A flag that indicates that the federated location and the visualizations associated with it can be deleted.
- **CreatedTime:** The UTC **datetime** when the federated location was created.
- **LastModifiedTime:** The UTC datetime when the federated location was last modified.

Visualization Set: A collection of entries corresponding to attributes used to display federated locations. Each entry MUST be uniquely identified by its LocationID and it MUST include the following elements:

- **LocationID:** The unique identifier of the federated location.
- **VisualizationIName:** The name of the visualization.
- **Xsl:** The Xsl for this visualization. This MUST be an Xsl Data Type as specified in section 2.2.1.22
- **Properties:** The properties for this visualization. This MUST be a Properties Data Type as specified in section 2.2.1.16.
- **SampleData:** The sample data for this visualization. This MUST be a SampleData Data Type as specified in 2.2.1.17.

3.1.1.8 Language Resources

The **Language Resources Set** is a list of language resources that are used throughout the farm. Each language resource MUST include the following elements:

- **Phrase**: A value whose meaning depends on the value of the **Type** element. See section 2.2.1.13 for the definition of the Language Resource Type.
- **Mapping**: A value whose meaning depends on the value of the **Type** element (section 2.2.1.13).
- **Type:** A Language Resource Type (section 2.2.1.13) that determines the meanings of the **Phrase** and **Mapping** elements.
- **Locale**: The LCID to which the language resource applies.
- **Author**: The author of the language resource.
- **Created**: The date and time at which the language resource was created.

There MUST NOT be two language resources with the same **Phrase**, **Mapping**, **Type** and **Locale** values.

Each **Language Resource Set** MUST also contain an **update sequence number (USN)**.

3.1.1.9 Query Suggestions

The following diagram describes the abstract data model for query suggestions. 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.

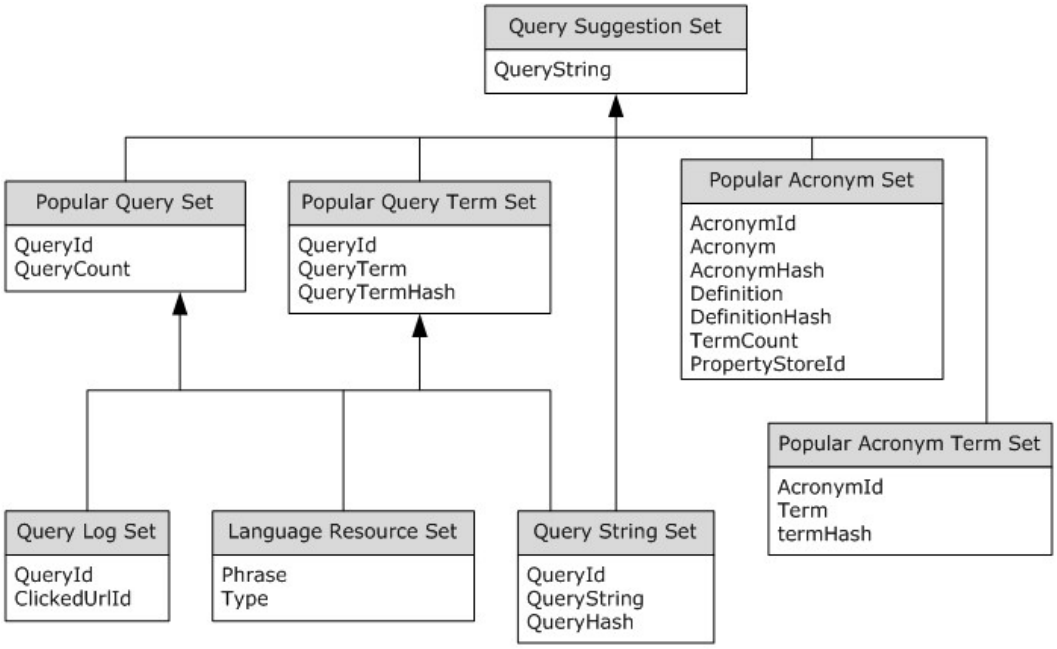

Figure 7: Query Suggestions Abstract Data Model

Query Log Set: A collection of entries, each corresponding to a search query that was executed. Each entry must include the following elements:

- **QueryId:** A unique identifier associated with each query.
- **ClickedUrlId:** A reference to the **query result URI** that was clicked.

Language Resource Set: A collection of entries corresponding to elements of language resources, as specified in section 3.1.1.8. Each entry MUST include the following elements:

- **Phrase:** A value whose meaning depends on the value of the **Type** element. See section 2.2.1.13 for the definition of the Language Resource Type.
- **Type:** A Language Resource Type (section 2.2.1.13) that determines the meanings of the Phrase and Mapping elements.

Query String Set: A collection of entries corresponding to unique search queries that were executed. Each entry MUST be uniquely identified by its QueryId, and it MUST include the following elements:

- **QueryId:** The unique identifier associated with the query.
- **QueryString:** The query text that was executed.
- **QueryHash:** The identifier of the query string.

Popular Query Set: A collection of entries corresponding to queries that are **popular**. Queries are popular if they are executed frequently and their search results are clicked on. Popular queries also include phrases which belong to the Language Resource Set and have their Type set to 2, as specified in section 3.1.1.8. Each entry MUST include the following items:

- **QueryId:** The identifier associated with the query.
- **QueryCount:** The number of times the query is executed.

Popular Query Term Set: A collection of entries corresponding to query terms associated with each **popular** query. Each entry MUST include the following items:

- **QueryId:** The identifier associated with the query.
- **QueryTerm:** A token extracted from the query string.
- **QueryTermHash:** The identifier of the query term.

Popular Acronym Set: A collection of entries corresponding to acronyms that are popular. Acronyms are popular if they have a high frequency of occurrence in the full-text index catalog. Each entry MUST include the following items:

- **AcronymId:** A identifier associated with the acronym.
- **Acronym:** The extracted term, if it is an acronym.
- **AcronymHash:** The identifier for the acronym string.
- **Definition:** The extracted definition for the corresponding extracted term.
- **DefinitionHash:** The identifier for the definition string.
- **TermCount:** The number of tokens extracted from the extracted definition.
- **PropertyStoreId:** A unique identifier associated with the metadata index.

Popular Acronym Term Set: A collection of entries corresponding to tokens associated with each popular acronym. Each entry MUST include the following items:

- **AcronymId:** An identifier associated with the acronym.
- **Term:** A token extracted from the extracted definition.
- **TermHash:** The identifier of the term string.

Query Suggestion Set: A collection of entries corresponding to query suggestions derived for an input query text. Each entry MUST include the following item:

- **QueryString:** The query suggestion string.

3.1.1.10 Search User Preference

There is one instance of search user preference per search user per **search service application**. Each search user preference instance MUST include the following items:

- A flag that specifies if the pre-query suggestions feature is enabled.
- A list of search query languages from which the user chooses the language of each search query.

- A list of custom setting flags that are not interpreted by the server.
- A custom setting string that is not interpreted by the server.
- A server configuration version number.

3.1.2 Timers

None.

3.1.3 Initialization

A connection that uses the underlying protocol layers, as specified in section 1.4, is established before using this protocol, as specified in [MS-TDS].

Listening endpoints are set up on the back-end database server to handle inbound TDS requests.

Authentication of the TDS connection to the back-end database server MUST occur before this protocol can be used.

The data structures, stored procedures, and actual data are persisted by the back-end database server within databases, so any operations to initialize the state of the database MUST occur before the back-end database server can use this protocol.

3.1.4 Higher-Layer Triggered Events

None.

3.1.5 Message Processing Events and Sequencing Rules

Unless otherwise specified, all stored procedure input parameters MUST NOT be NULL. As stored procedures use the input parameters for data retrieval from tables, failure to provide valid values will (unless otherwise specified) cause an error as specified in [MS-TDS] section 2.2.2.7 that MUST be handled appropriately by the protocol client or the system behavior is indeterminate.

Unless otherwise specified, all fields returned in the result sets MUST NOT be NULL. For definitional clarity, a name has been assigned to any columns in the result sets that do not have a defined name in their current implementation. This does not affect the operation of the result set, as the ordinal position of any column with no defined name is expected by the front-end Web server. Such names are designated in the text using curly braces in the form {*name*}.

The data processing specified in this section references most of the elements of the abstract data model, as specified in section 3.1.1.

3.1.5.1 proc_MSS_AddAuthorityPage

The **proc_MSS_AddAuthorityPage** stored procedure is called to add an authority page.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_AddAuthorityPage(
   @Url              nvarchar(2048),
   @Hash             int,
   @AuthorityLevel   int
);
```

@Url: The URL of the authority page.

@Hash: The identifier of the URL. This value MUST NOT be NULL.

@AuthorityLevel: The authority level of the authority page. This parameter MUST be set to an integer listed in the following table.

Value	Description
0	First-level authority page.
100	Second-level authority page.
200	Third-level authority page.

Return Code Values: An integer which MUST be listed in the following table:

Value	Description
0	Successful execution.
547	The URL is already defined as an authority page.

Result Sets: SHOULD NOT<1> return any result sets. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.2 proc_MSS_AddBestBet

The **proc_MSS_AddBestBet** stored procedure is called to create a new best bet and associate it with the specified keyword.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_AddBestBet(
   @Title          nvarchar(100),
   @Url            nvarchar(2048),
   @Description    nvarchar(500)
   @SpecialTermId  int,
   @ConsumerGpId   nvarchar(50),
   @Order          int,
   @BestBetId      int output
);
```

@Title: The title for the best bet. This value MUST NOT be NULL.

@Url: The URL for the best bet. This value MUST NOT be NULL.

@Description: The description for the best bet.

@SpecialTermId: The unique identifier of the keyword associated with the newly created best bet. This value MUST NOT be NULL.

@ConsumerGpId: The unique identifier of the keyword consumer group for the best bet.

@Order: The order of the best bet among all best bets for a specified keyword. This value MUST NOT be NULL.

@BestBetId: Upon return from this stored procedure, this parameter MUST be set to the unique identifier of the newly added best bet.

Return Code Values: A value which MUST be listed in the following table:

Value	Description
0	Successful execution.
2627	The identifier of the best bet already exists.
'SPECIAL TERM ERROR'	A string error value that indicates that a best bet with @Url and @ConsumerGpId already exists.

Result Sets: SHOULD NOT<2> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.3 proc_MSS_AddBestBetLink

The **proc_MSS_AddBestBetLink** stored procedure is called to add an association between a best bet and a keyword.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_AddBestBetLink(
   @SpecialTermId     int,
   @BestBetId         int,
   @Order             int
);
```

@SpecialTermId: The unique identifier of the keyword associated with the newly created best bet. This value MUST NOT be NULL.

@BestBetId: The unique identifier of the best bet. This value MUST NOT be NULL.

@Order: The order of the best bet among all best bets for specified keyword. This value MUST NOT be NULL.

Return Code Values: An integer which MUST be listed in the following table:

Value	Description
0	Successful execution
2627	The association between the best bet and the keyword already exists

Result Sets: SHOULD NOT<3> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.4 proc_MSS_AddConsumer

The **proc_MSS_AddConsumer** stored procedure is called to add a search scope consumer to the list of search scope consumers to view, add, change or delete search scopes and search scope display groups within it.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_AddConsumer (
   @ConsumerName      nvarchar(60)
);
```

@ConsumerName: The name that uniquely identifies the search scope consumer. This value MUST NOT be NULL.

Return Code Values: An integer which MUST be listed in the following table:

Value	Description
0	Successful execution.
1	The search scope consumer already exists.

Result Sets: SHOULD NOT<4> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.5 proc_MSS_AddCrawledProperty

The **proc_MSS_AddCrawledProperty** stored procedure is called to add a crawled property to the Crawled Property Set, specified in section 3.1.1.1.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_AddCrawledProperty (
   @Propset                uniqueidentifier,
   @PropertyName           nvarchar(440),
   @PropertyNameIsEnum     bit,
   @IsMappedToContent      bit,
   @VariantType            int,
   @URI                    nvarchar(2048),
   @CrawledPropertyId      int OUTPUT
);
```

@Propset: The crawled property set identifier associated with the crawled property category**.** This value MUST NOT be NULL.

@PropertyName: The name of the crawled property. This value MUST NOT be NULL.

@PropertyNameIsEnum: This **parameter** MUST be set to 1 if the *@PropertyName* string value was converted from an integer. Otherwise, it MUST be set to 0.

@IsMappedToContent: This value MUST be set to 1 if the variant type (2) is a string, and data from this crawled property is put in the full-text index catalog. Otherwise, it MUST be set to 0.

@VariantType: The variant type (2) for the crawled property. This value MUST NOT be NULL.

@URI: The URI associated with the crawled property.

@CrawledPropertyId: Upon return from this stored procedure, this parameter MUST be set to the unique identifier of the crawled property. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<5> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.6 proc_MSS_AddCrawledPropertyCategoryFromOM

The **proc_MSS_AddCrawledPropertyCategoryFromOM** stored procedure is called to add a crawled property category to the Crawled Property Category Set, specified in section 3.1.1.1. Upon successful execution this procedure updates the CrawledPropertyCategoryTimestamp attribute in the Metadata Timestamp Set with the current date and time in local time of the server.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_AddCrawledPropertyCategoryFromOM (
   @CategoryName       nvarchar(64),
   @Propset            uniqueidentifier
);
```

@CategoryName: The name of the crawled property category. This value MUST NOT be NULL.

@Propset: The crawled property set identifier associated with the crawled property category. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<6> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.7 proc_MSS_AddLanguageResource

The **proc_MSS_AddLanguageResource** stored procedure is called to add a language resource. The stored procedure adds an entry to the Language Resources Set and updates its elements with the values provided by the stored procedure parameters. See section 3.1.1.8 for the definition of Language Resources Set.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_AddLanguageResource(
   @Phrase       nvarchar(128),
   @Mapping      nvarchar(128),
   @Type         smallint,
   @Locale       int,
   @Author       nvarchar(256),
   @Created      datetime
);
```

@Phrase: The **Phrase** element of the language resource. This value MUST NOT be NULL.

@Mapping: The **Mapping** element of the language resource. This value MUST NOT be NULL.

@Type: The **Type** element of the language resource. The value MUST be a valid Language Resource Type as specified in section 2.2.1.13.

@Locale: The **Locale** element of the language resource. This value MUST NOT be NULL.

@Author: The **Author** element of the language resource. This value MUST NOT be NULL.

@Created: The **Created** element of the language resource. This value MUST NOT be NULL.

Return Code Values: An integer that MUST be listed in the following table.

Value	Description
0	Successful execution.
2627	A language resource with the same **Phrase**, **Mapping**, **Type** and **Locale** elements already exists.

Result Sets: SHOULD NOT<7> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

In addition to adding the language resource, the protocol server MUST increase the language resource set's **USN** by one.

3.1.5.8 proc_MSS_AddManagedPropertyEx

The **proc_MSS_AddManagedPropertyEx** stored procedure is called to add a **managed property** to the Managed Property Set, specified in section 3.1.1.1. Upon successful execution the procedure updates the ManagedPropertyAddModifyTimestamp attribute in the Metadata Timestamp Set, specified in section 3.1.1.1, to the current date and time in local time of the server.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_AddManagedPropertyEx (
   @PID                              int OUTPUT,
   @Name                             nvarchar (64),
   @PropertyDescription              nvarchar(2048),
   @ManagedType                      int,
   @FullTextQueriable                bit,
   @Retrievable                      bit,
   @Scoped                           bit,
   @RespectPriority                  bit,
   @MaxIndexedStringLength           int,
   @MaxNonIndexedStringLength        int,
   @MaxRetrievalLength               int,
   @DecimalPlaces                    tinyInt,
   @IsInDocProps                     bit,
   @IsInPropertyBlobOptimizedResults bit,
   @IsInFixedColumnOptimizedResults  bit,
   @SuppressStringNormalizer         bit,
   @Pronunciation                    bit,
   @Nickname                         bit,
   @SplitStringCharacters            nvarchar(64)
);
```

@PID: Upon return from this stored procedure, this parameter MUST be set to the unique identifier of the managed property.

@Name: A string that uniquely identifies the managed property. This value MUST NOT be NULL.

@PropertyDescription: The description of the managed property.

@ManagedType: The type of the managed property. The value MUST be a valid Managed Type data type, as specified in section 2.2.1.15.

@FullTextQueriable: This value MUST be 1 if the data for the managed property is kept in the full-text index catalog. Otherwise, it MUST be 0.

@Retrievable: This value MUST be 1 if the data for the managed property is kept in the metadata index. Otherwise, it MUST be 0.

@Scoped: This value MUST be 1 if the data for the managed property is kept in the **search scope index**. Otherwise, it MUST be 0.

@RespectPriority: This value MUST be 1 if only data from the crawled property mapped to this managed property with highest priority of its **mapping order** is used. It MUST be 0 if values from all crawled properties mapped to this managed property are used**.**

@MaxIndexedStringLength: The maximum number of characters persisted in the string value column in the *MSSDocProps* table defined in [MS-SQLPQ2] section 2.2.5.2. If the string value length is greater than MaxIndexedStringLength, only the first MaxIndexedStringLength characters of the string value are persisted. This value MUST NOT be NULL.

@MaxNonIndexedStringLength: If the string value length in the *MSSDocProps* table, defined in [MS-SQLPQ2] section 2.2.5.2, is greater than the size allowed by the MaxIndexedStringLength, only the first MaxIndexedStringLength characters of the string value are persisted and the string overflow is stored in the strVal2 column in the *MSSDocProps* table defined in [MS-SQLPQ2] section 2.2.5.2, up to the size allowed by the MaxNonIndexedStringLength. This value MUST NOT be NULL.

@MaxRetrievalLength: The maximum number of characters persisted for a fixed-length string property in the *MSSDocResults* table defined in [MS-SQLPQ2] section 2.2.5.3. This MUST NOT be NULL.

@DecimalPlaces: The number of floating point decimal places that must be honored in the metadata index. This value MUST NOT be NULL.

@IsInDocProps: This value MUST be 1 if the managed property is stored in the *MSSDocProps* table defined in [MS-SQLPQ2] section 2.2.5.2. Otherwise, MUST be 0.

@IsInPropertyBlobOptimizedResults: This value MUST be 1 if the managed property is persisted in the *PropertyBlob* column of the *MSSDocResults* table defined in [MS-SQLPQ2] section 2.2.5.3. Otherwise, MUST be 0.

@IsInFixedColumnOptimizedResults: This value MUST be 1 if the managed property is persisted in the *MSSDocResults* table defined in [MS-SQLPQ2] section 2.2.5.3. Otherwise, MUST be 0.

@SuppressStringNormalizer: This value MUST be 1 if the string normalization for the managed property is to be skipped. Otherwise, MUST be 0.

@Pronunciation: This value MUST be 1 if the managed property needs a pronunciation string. Otherwise, MUST be 0.

@NickNameExpansion: This value MUST be set to 1 if the managed property is compared to the **Nickname** mappings at query time. Otherwise, it MUST be 0.

@SplitStringCharacters: This value MUST contain characters which are used to split the string data for the managed property into separate strings which do not contain the SplitStringCharacters. Otherwise, it MUST be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<8> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure

3.1.5.9 proc_MSS_AddManagedPropertyAlias

The **proc_MSS_AddManagedPropertyAlias** stored procedure is called to add another name which is associated with the managed property to the metadata schema. Upon successful execution the stored procedure updates the value of ManagedPropertyAddModifyTimestamp in the Metadata Timestamp Set and the value of LastModifiedTime attribute in the Managed Property Set for the corresponding managed property to the current date and time in local time of the server. For the specification of Metadata Timestamp Set and Managed Property Set see section 3.1.1.1.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_AddManagedPropertyAlias (
   @PID                      int,
   @Name                     nvarchar(2048),
   @UpdateManagedProperty    bit = 1
);
```

@PID: The unique identifier of the managed property. This value MUST NOT be NULL.

@Name: An alternate string which identifies the managed property. This value MUST NOT be NULL.

@UpdateManagedProperty: This value MUST be set to 1.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<9> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.10 proc_MSS_AddMappingToPendingTable

The **proc_MSS_AddMappingToPendingTable** stored procedure is called to add a mapping between a crawled property and a managed property into the Pending Mappings Set, specified in Section 3.1.1.1, with the lowest mapping order.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_ AddMappingToPendingTable (
   @PID                    int,
   @CrawledPropset         uniqueidentifier,
   @CrawledPropertyName    nvarchar(440),
   @CrawledVariantType     int
);
```

@PID: The unique identifier of the managed property. This value MUST NOT be NULL.

@CrawledPropset: The crawled property set identifier associated with the crawled property. This value MUST NOT be NULL.

@CrawledPropertyName: The name of the crawled property. This value MUST NOT be NULL.

@CrawledVariantType: The variant type (2) for the crawled property. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<10> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.11 proc_MSS_AddRankingModelOM

The proc_MSS_AddRankingModelOM stored procedure creates a new ranking model and stores it in the Ranking Model Set, specified in section 3.1.1.5.1. The created ranking model MUST have a value of 0 set for the flag which indicates if this is a default ranking model. Immediately after adding the specified ranking model the stored procedure updates the Global Ranking Parameters Timestamp variable, specified in section 3.1.1.1, with the current date and time.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_AddRankingModelOM (
   @ModelId     uniqueidentifier,
   @ModelXml    nvarchar(max)
);
```

@ModelId: an identifier for the ranking model. This identifier MUST be unique within the Ranking Model Set, specified in section 3.1.1.5.1.

@ModelXml: a string representation of weights and parameters of the ranking model to be created. This value MUST NOT be NULL.

Return Code Values:

Value	Description
0	The ranking model was created successfully
2627	The ModelId specified was duplicated in the Ranking Model Set, specified in section 3.1.1.5.1. The ranking model was not created upon the return of the stored procedure.

Result Sets: MUST NOT return any result sets.

3.1.5.12 proc_MSS_AddScope

The **proc_MSS_AddScope** stored procedure is called to add a new search scope to the **search application.** Upon successful execution the stored procedure modifies the CompilationState value in the Scope Set. The stored procedure also increments the LastConsumerChangeID value in the Scope System Set and sets the LastUpdate value for the corresponding consumer in the Consumer Set to the resulting LastConsumerChangeID value. See section 3.1.1.4 for details.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_AddScope (
   @Name                       nvarchar(60),
   @Description                nvarchar(300),
   @ConsumerName               nvarchar(60),
   @DisplayInAdminUI           bit,
   @AlternateResultsPageUrl    nvarchar(2047)= null,
   @CompilationType            smallint,
   @ModifierName               nvarchar(60),
```

```
    @ScopeID                    int OUTPUT,
    @Filter                     nvarchar(2047) = null
    @LargeFilter                nvarchar(max) = null
);
```

@Name: The name of the search scope. This value MUST NOT be NULL.

@Description: The description of the search scope. This value MUST NOT be NULL.

@ConsumerName: The name of the search scope consumer which owns the search scope. This value MUST NOT be NULL.

@DisplayInAdminUI: A bit flag indicating if the search scope display group is displayed in the Administration user interface. The value MUST be a DisplayInAdminUI data type as specified in section 2.2.1.9.

@AlternateResultsPageUrl: The URL of an alternate web page to display the results of a search performed on this search scope.

@CompilationType: The compilation type of the search scope. The value MUST be a Compilation Type data type as specified in section 2.2.1.5.

@ModifierName: The name of the user who last changed the attributes of the search scope. This value MUST NOT be NULL.

@ScopeID: Upon return from this stored procedure, this parameter MUST be set to the unique identifier of the created search scope.

@Filter: Reserved. This parameter MUST be ignored by the client.

@LargeFilter: Reserved. This parameter MUST be ignored by the client.

Return Code Values: An integer which MUST be listed in the following table:

Value	Description
0	Successful execution. The search scope was successfully added to the search application.
1	Search scope consumer with the specified name could not be found. The search scope was not added to the search application.

Result Sets: SHOULD NOT<11> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.13 proc_MSS_AddScopeDisplayGroup

The **proc_MSS_AddScopeDisplayGroup** stored procedure is called to add a search scope display group for a search scope consumer. Upon successful execution the stored procedure increments the LastConsumerChangeID value in the Scope System Set and sets the LastUpdate value for the corresponding consumer in the Consumer Set to the resulting LastConsumerChangeID value. See section 3.1.1.4 for details.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_AddScopeDisplayGroup (
   @Name                    nvarchar(60),
```

```
    @Description            nvarchar(300),
    @ConsumerName           nvarchar(60),
    @DisplayInAdminUI       bit,
    @Undeletable            bit,
    @DefaultScopeID         int,
    @ModifierName           nvarchar(60),
    @DisplayGroupID         int OUTPUT
);
```

@Name: The name that uniquely identifies the search scope display group. This value MUST NOT be NULL.

@Description: The description of the search scope display group. This value MUST NOT be NULL.

@ConsumerName: The name of the search scope consumer who is the owner of the search scope display group. This value MUST NOT be NULL.

@DisplayInAdminUI: A bit flag indicating if the search scope display group is displayed in the Administration user interface. The value MUST be a DisplayInAdminUI data type as specified in section 2.2.1.9.

@Undeletable: A bit flag indicating if the search scope display group can be deleted. The value MUST be Undeletable data type as specified in section 2.2.1.20**.**

@DefaultScopeID: The unique identifier of the default search scope associated with the search scope display group. This value MUST NOT be NULL.

@ModifierName: The name of the user who last changed attributed of the search scope display group. This value MUST NOT be NULL.

@DisplayGroupID: Upon return from this stored procedure, this parameter MUST be set to the unique identifier of the search scope display group.

Return Code Values: An integer which MUST be listed in the following table:

Value	Description
0	Successful execution.
1	The search scope consumer does not exist.

Result Sets: SHOULD NOT<12> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.14 proc_MSS_AddScopeRule

The **proc_MSS_AddScopeRule** stored procedure is called to add a new search scope rule to the specified search scope**.** Upon successful execution the stored procedure modifies the CompilationState value in the Scope Set. The stored procedure also increments the LastConsumerChangeID value in the Scope System Set and sets the LastUpdate value for the corresponding consumer in the Consumer Set to the resulting LastConsumerChangeID value. See section 3.1.1.4 for details.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_AddScopeRule (
   @ScopeID              int,
   @FilterBehavior       smallint,
   @RuleType             smallint,
   @UrlRuleType          smallint,
   @PropertyID           int,
   @UserValueString      nvarchar(2047),
   @ModifierName         nvarchar(60),
   @RuleID               int OUTPUT
);
```

@ScopeID: The unique identifier of the search scope. This value MUST NOT be NULL.

@FilterBehavior: The filter behavior of the search scope rule. The value MUST be a valid ScopeFilterBehavior data type as specified in section 2.2.1.18.

@RuleType: The type of the search scope rule. The value MUST be a ScopeRuleType data type as specified in section 2.2.1.19.

@UrlRuleType: The type of the URL for the search scope rule. The value MUST be an UrlRuleType data type as specified in section 2.2.1.21.

@PropertyID: The unique identifier of the managed property to use in the search scope rule.

@UserValueString: The search scope rule value to use in the search scope rule.

@ModifierName: The name of the user adding the search scope rule. This value MUST NOT be NULL.

@RuleID: Upon return from this stored procedure, this parameter MUST be set to the unique identifier of the search scope rule.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<13> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.15 proc_MSS_AddSpecialTerm

The **proc_MSS_AddSpecialTerm** stored procedure is called to add a new keyword to the site collection**.**

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_AddSpecialTerm(
   @Term              nvarchar(100),
   @ConsumerGpId      nvarchar(50),
   @StartDate         datetime,
   @SpecialTermId     int output
);
```

@Term: The term for the keyword. This value MUST NOT be NULL.

@ConsumerGpId: The unique identifier of the **keyword consumer group** associated with the keyword.

@StartDate: The date and time when the keyword begins to appear in search result. This value MUST NOT be NULL.

@SpecialTermId: Upon return from this stored procedure, this parameter MUST be set to the unique identifier of the newly added keyword. This value MUST NOT be NULL.

Return Code Values: A value which MUST be listed in the following table:

Value	Description
Value of @SpecialTermId	Successful execution. This value MUST be ignored by the protocol client.
'SPECIAL TERM ERROR'	The keyword already exists in the site collection.
2627	The keyword already exists in the site collection.

Result Sets: SHOULD NOT<14> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.16 proc_MSS_AddSynonym

The **proc_MSS_AddSynonym** stored procedure is called to associate a synonym with the specified keyword in the site collection as specified in section 3.1.1.2.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_AddSynonym(
   @SpecialTermId      int,
   @Term               nvarchar(100)
);
```

@SpecialTermId: The unique identifier of the keyword associated with the keyword synonym. This value MUST NOT be NULL.

@Term: The term for the keyword synonym. This value MUST NOT be NULL.

Return Code Values: An integer which MUST be listed in the following table.

Value	Description
0	Successful execution.
2627	The synonym with the specified *@Term* is already defined for the specified keyword.

Result Sets: SHOULD NOT<15> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.17 proc_MSS_BeginScopeDisplayGroupList

The **proc_MSS_BeginScopeDisplayGroupList** stored procedure is called to remove all search scopes with negative order from the specified search scope display group. The stored procedure MUST be called before the first call to proc_MSS_SetScopeDisplayGroupListItem stored procedure. See section 3.1.5.158 for the specification of proc_MSS_SetScopeDisplayGroupListItem stored procedure.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_BeginScopeDisplayGroupList(
   @DisplayGroupID    int
);
```

@DisplayGroupID: The unique identifier of the search scope display group. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<16> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.18 proc_MSS_Cleanup

The **proc_MSS_Cleanup** stored procedure is called to delete entries from the Sample Crawled Property Set, specified in section 3.1.1.1, which are no longer associated with the items in the metadata index.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_Cleanup();
```

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<17> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.19 proc_MSS_ContainsManagedPropertyAlias

The **proc_MSS_ContainsManagedPropertyAlias** stored procedure is called to determine whether a managed property alias is defined in the metadata schema.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_ContainsManagedPropertyAlias (
   @PID              int,
   @alias            nvarchar(2048),
   @found            bit OUTPUT
);
```

@PID: The unique identifier of the managed property. This value MUST NOT be NULL.

@alias: An alternate string name which identifies a managed property. This value MUST NOT be NULL.

@found: Upon return from this stored procedure, this parameter MUST be set to 1 if the managed property alias is defined in the metadata schema; otherwise the value MUST be set to 0.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: This procedure MUST NOT return any result set.

3.1.5.20 proc_MSS_CreateCommand

The **proc_MSS_CreateCommand** stored procedure is called to add a new command to the command set specified in [MS-SQLPGAT2] section 3.1.1.42.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_CreateCommand (
    @ObjectName               nvarchar(1024),
    @Description              nvarchar(3072),
    @CreatedUtcTime           datetime,
    @CommandID                bigint OUTPUT
);
```

@ObjectName: The name of the object which will process the command.

@Description: The text that will be interpreted and acted upon by the named object.

@CreatedUtcTime: The current time.

@CommandID: A 64-bit integer equal to the commandID ([MS-SQLPGAT2] section 3.1.1.42) of the added command.

When this stored procedure is called, a new command MUST be added to the commands set ([MS-SQLPGAT2] section 3.1.1.42), where CommandID is a number higher than any other commandID value in the commands set ([MS-SQLPGAT2] section 3.1.1.42), ObjectName ([MS-SQLPGAT2] section 3.1.1.42) is set to *@ObjectName*, Description ([MS-SQLPGAT2] section 3.1.1.42) is set to *@Description*, and Completed ([MS-SQLPGAT2] section 3.1.1.42) is set to false.

Return Code Values: An integer which MUST be ignored.

Result Sets: MUST NOT return any result set.

3.1.5.21 proc_MSS_CrawlReportAddNewItems

The **proc_MSS_CrawlReportAddNewItems** stored procedure is called to copy data from the Crawl Url History set ([MS-SQLPGAT2] section 3.1.1.2) to the Crawl Item Set (section 3.1.1.3) for the new items that have been added by the crawler to the search catalog within the specified date and time.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_CrawlReportAddNewItems(
   @timepoint datetime
);
```

@timepoint: The maximum date and time in UTC time which MUST be greater than the time the item was created by the crawler. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<18> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.22 proc_MSS_CrawlReportCleanup

The **proc_MSS_CrawlReportCleanup** stored procedure is called to remove items from the Crawl Item Set for which the value of *TimeStamp* attribute is greater than the specified number of days. Crawl Used Error Set and Crawl Used Content Source Set MUST be updated in accordance with updated Crawl Item Set (see section 3.1.1.3). The stored procedure also removes entries from the following tables for crawls completed before the specified number of days:

- MSSCrawlDeletedURL, see [MS-SRCHTP] section 2.2.5.9.
- MSSCrawlLinksLog, see [MS-SRCHTP] section 2.2.5.12.
- MSSCrawlURLLog, see [MS-SRCHTP] section 2.2.5.8.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_CrawlReportCleanup(
   @CleanupInterval int
);
```

@CleanupInterval: The number of days to keep crawl log data. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<19> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.23 proc_MSS_Update_MSSCrawlUrlUsedErrorReport

The **proc_MSS_Update_MSSCrawlUrlUsedErrorReport** stored procedure is called to update Crawl Used Error Set in accordance with updated Crawl Item Set (see section 3.1.1.3).

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_Update_MssCrawlUrlUsedErrorReport();
```

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<20> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.24 proc_MSS_CrawlReportMarkDeletedItems

The **proc_MSS_CrawlReportMarkDeletedItems** stored procedure is called to update attributes in the Crawl Item Set for the items that have been removed from the search catalog before the specified date and time. See section 3.1.1.3 for the definition of Crawl Item Set. The stored procedure MUST set the value ErrorLevel attribute to 3 as specified in section 2.2.1.7. All other attributes of these entries should be updated in accordance with Crawl Url History sets ([MS-SQLPGAT2] section 3.1.1.3).

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_CrawlReportMarkDeletedItems(
   @timepoint datetime
```

```
);
```

@timepoint: The maximum date and time in UTC time when the item was removed from the search catalog. The stored procedure MUST only update items for which the value of Timestamp attribute is less than the specified parameter value. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<21> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.25 proc_MSS_CrawlReportPreprocessChanges

The **proc_MSS_CrawlReportPreprocessChanges** stored procedure is called to perform any preprocessing steps which are needed for efficient execution of proc_MSS_CrawlReportAddNewItems (section 3.1.5.21), proc_MSS_CrawlReportMarkDeletedItems (section 3.1.5.24), proc_MSS_CrawlReportReuseItems (section 3.1.5.26) and proc_MSS_CrawlReportUpdateItems (section 3.1.5.27) stored procedures. This stored procedure is called before the preceding mentioned stored procedures are called. Then all of these stored procedures are called with the same @timepoint parameter value.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_CrawlReportPreprocessChanges (
   @timepoint datetime
);
```

@timepoint: The maximum date and time in UTC time when the item was last crawled. The stored procedure MUST process only items for which the value of Timestamp attribute is less than the specified parameter value. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<22> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.26 proc_MSS_CrawlReportReuseItems

The **proc_MSS_CrawlReportReuseItems** stored procedure is called to copy data from the Crawl Url History set ([MS-SQLPGAT2] section 3.1.1.2) to the Crawl Item Set (section 3.1.1.3) for the items whose value of DocID attribute is reused by the crawler to create a new item in the search catalog. If the item already exists in Crawl Item Set, the stored procedure updates the item with the new data from Crawl Url History set. Otherwise, the stored procedure adds a new entry.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_CrawlReportReuseItems(
   @timepoint datetime
);
```

@timepoint: The maximum date and time in UTC time when the item was last crawled. The stored procedure MUST only copy items for which the value of Timestamp attribute is less than the specified parameter value. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<23> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.27 proc_MSS_CrawlReportUpdateItems

The **proc_MSS_CrawlReportUpdateItems** stored procedure is called to copy data from the Crawl Url History set ([MS-SQLPGAT2] section 3.1.1.2) to the Crawl Item Set (section 3.1.1.3) for the items that already exist in the search catalog and have been updated by the crawler within the specified date and time.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_CrawlReportUpdateItems(
   @timepoint datetime
);
```

@timepoint: The maximum date and time in UTC time when the item was last crawled. The stored procedure MUST only copy items for which the value of Timestamp attribute is less than the specified parameter value. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<24> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.28 proc_MSS_DefragSearchIndexes

The **proc_MSS_DefragSearchIndexes** stored procedure is called to check whether any maintenance of the metadata index is required and optionally repair found issues. The stored procedure does not affect the state of the protocol, and is dependent on the implementation of the back-end database server.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_DefragSearchIndexes (
   @CheckOnly bit = 0,
   @ProcessOffline int = 0,
   @maxdopLevel bit = 0,
   @defragThreshold float = 10.0
);
```

@CheckOnly: An integer which MUST be one of the values listed in the following table:

Value	Description
0	The procedure MUST check the state of back-end database server and repair found issues.
1	The procedure MUST only check the state of back-end database server.

@ProcessOffline: Back-end database maintenance parameter. The interpretation of this parameter is dependent on the implementation of the back-end database server. First, the stored procedure is called with @ProcessOffline parameter set to 0. If an SqlException with code 1712 is be thrown

during its execution then this stored procedure will be called one more time with this parameter set to 1.

@maxdopLevel: Back-end database maintenance parameter. The interpretation of this parameter is dependent on the implementation of the back-end database server.

@defragThreshold: Back-end database maintenance parameter. The interpretation of this parameter is dependent on the implementation of the back-end database server.

Return values: An integer which MUST be one of the values listed in the following table:

Value	Description
0	There are no issues to repair.
1	Repair is required.

Result Sets: SHOULD NOT<25> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.29 proc_MSS_DeleteAuthorityPage

The **proc_MSS_DeleteAuthorityPage** stored procedure is called to delete the authority page with the specified URL from the list of authority pages.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_DeleteAuthorityPage (
   @Url          nvarchar(2048)
);
```

@Url: The URL of the authority page to be deleted.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<26> return any result sets. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.30 proc_MSS_DeleteBestBetLink

The **proc_MSS_DeleteBestBetLink** stored procedure is called to delete an association between the specified keyword and the best bet.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_DeleteBestBetLink(
   @SpecialTermId         int,
   @BestBetID             int
);
```

@SpecialTermId: The unique identifier of the keyword. This value MUST NOT be NULL.

@BestBetID: The unique identifier of the best bet. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<27> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.31 proc_MSS_DeleteCrawledCategoryByName

The **proc_MSS_DeleteCrawledCategoryByName** stored procedure is called to delete the specified crawled property from the Crawled Property Category Set. The crawled property MUST have no crawled properties associated with it in the Crawled Property Set. This procedure also changes the CrawledPropertyCategoryTimestamp attribute in the Metadata Timestamp Set to the current date and time in local time of the server. For the specification of Crawled Property Category Set and Metadata Timestamp Set see Section 3.1.1.1.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_DeleteCrawledCategoryByName (
   @Name              nvarchar(64)
);
```

@Name: The name of the crawled property category. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<28> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.32 proc_MSS_DeleteCrawledPropertiesUnmappedForCategory

The **proc_MSS_DeleteCrawledPropertiesUnmappedForCategory** stored procedure is called to remove the set of crawled properties from the Crawled Property Set which conform to the following criteria:

- The crawled property belongs to the same crawled property set identifier as the specified crawled property category.
- The crawled property's data is not put in the full-text index catalog.
- The crawled property is not mapped to any managed property.

Upon successful execution the stored procedure updates the CrawledPropertyDeleteTimestamp attribute in the Metadata Timestamp Set to the current date and time in local time of the server. For the specification of the Crawled Property Set and the Metadata Timestamp Set see section 3.1.1.1.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_DeleteCrawledPropertiesUnmappedForCategory (
   @CategoryName            nvarchar(64)
);
```

@CategoryName: The name of the crawled property category. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<29> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.33 proc_MSS_DeleteLanguageResource

The **proc_MSS_DeleteLanguageResource** stored procedure is called to delete language resources. The stored procedure deletes entries from the Language Resources Set for which **Phrase**, **Mapping**, **Type** and **Locale** elements match the values provided by the stored procedure parameters. See section 3.1.1.8 for the definition of Language Resources Set.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_DeleteLanguageResource(
   @Phrase              nvarchar(128),
   @Mapping             nvarchar(128),
   @Type                smallint,
   @Locale              int
);
```

@Phrase: The **Phrase** element of the language resource to be deleted. This value MUST NOT be NULL.

@Mapping: The **Mapping** element of the language resource to be deleted.

@Type: The **Type** element of the language resource. The value MUST be a valid Language Resource Type as specified in section 2.2.1.13.

@Locale: The **Locale** element of the language resource to be deleted. This value MUST NOT be NULL.

Return Code Values: An integer which MUST be zero.

Result Sets: SHOULD NOT<30> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

In addition to deleting the language resource, the protocol server MUST increase the language resource set's USN by one.

3.1.5.34 proc_MSS_DeleteManagedProperty

The stored procedure is called to remove a managed property from the Managed Property Set. The managed property MUST have no crawled properties associated with it in the Mappings Set. This procedure also removes the entry associated with this managed property from the Static Ranking Feature Set. For the specification of Static Ranking Feature Set, see Section 3.1.1.5.2. Upon successful execution the procedure changes the ManagedPropertyDeleteTimestamp attribute in the Metadata Timestamp Set to the current date and time in local time of the server. For the specification of Managed Property Set, Mappings Set, and Metadata Timestamp Set see section 3.1.1.1.

```
PROCEDURE proc_MSS_DeleteManagedProperty (
   @PID         int
);
```

@PID: The unique identifier of the managed property. This value MUST NOT be NULL.

Return Code Values: An integer which MUST be listed in the following table:

Value	Description
0	Successful execution.
11	Failed to delete the managed property because there are crawled properties that are still mapped to it.

Result Sets: MUST NOT return any result set.

3.1.5.35 proc_MSS_DeleteManagedPropertyAlias

The **proc_MSS_DeleteManagedPropertyAlias** stored procedure is called to delete a managed property alias from the metadata schema**.** Upon successful execution this procedure sets the ManagedPropertyAddModifyTimestamp attribute in the Metadata Timestamp Set and the LastModifiedTime attribute in the Managed Property Set for the corresponding managed property to the current date and time of the server. For the specification of Managed Property Set and Metadata Timestamp Set see section 3.1.1.1.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_DeleteManagedPropertyAlias(
   @PID        int,
   @Alias      nvarchar(2048)
);
```

@PID: The unique identifier of the managed property. This value MUST NOT be NULL.

@Alias: The managed property alias. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<31> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.36 proc_MSS_DeletePropertyMappingsForManagedProperty

The **proc_MSS_DeletePropertyMappingsForManagedProperty** stored procedure is called to delete the **search property mappings** to this managed property from the Mappings Set. This procedure changes the CrawledPropertyTimestamp attribute in the Metadata Timestamp Set to the current date and time in local time of the server**.** For the specification of Mappings Set and Metadata Timestamp Set see Section 3.1.1.1.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_DeletePropertyMappingsForManagedProperty (
   @PID           int
);
```

@PID: The unique identifier of the managed property. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<32> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure

3.1.5.37 proc_MSS_DeletePropertyMappingsPendingForManagedProperty

The **proc_MSS_DeletePropertyMappingsPendingForManagedProperty** stored procedure is called to delete the search property mappings to this managed property from the Pending Mappings Set specified in Section 3.1.1.1.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_DeletePropertyMappingsPendingForManagedProperty (
   @PID         int
);
```

@PID: The unique identifier of the managed property. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<33> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.38 proc_MSS_DeleteRankingModelOM

The **proc_MSS_DeleteRankingModelOM** stored procedure removes a ranking model from the Ranking Model Set, specified in section 3.1.1.5.1. Immediately after removing the specified ranking model the stored procedure updates the Global Ranking Parameters Timestamp variable, specified in section 3.1.1.1, with the current date and time.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_DeleteRankingModelOM (
   @ModelId      uniqueidentifier
);
```

@ModelId: The identifier of the ranking model to be deleted. This value MUST NOT be NULL.

Return Code Values:

Value	Description
0	The operation was successful.
1	The ranking model specified by @ModelId was not found and no changes were made to the Ranking Model Set, specified in section 3.1.1.5.1.

Result Sets: MUST NOT return any result set.

3.1.5.39 proc_MSS_DeleteSpecialTerm

The **proc_MSS_DeleteSpecialTerm** stored procedure is called to delete a keyword from the site collection.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_DeleteSpecialTerm(
   @SpecialTermId         int
```

```
);
```

@SpecialTermId: The unique identifier of the keyword to delete. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<34> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure

3.1.5.40 proc_MSS_DeleteSynonym

The **proc_MSS_DeleteSynonym** stored procedure is called to delete a synonym of the specified keyword from the site collection.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_DeleteSynonym(
   @SpecialTermId             int,
   @Term                      nvarchar(100)
);
```

@SpecialTermId: The unique identifier of the keyword**.** This value MUST NOT be NULL.

@Term: The keyword synonym to be deleted. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<35> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.41 proc_MSS_DropConsumer

The **proc_MSS_DropConsumer** stored procedure is called to remove the specified search scope consumer from the list of search scope consumers.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_DropConsumer (
   @ConsumerName         nvarchar(60)
);
```

@ConsumerName: The name that uniquely identifies the search scope consumer. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<36> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.42 proc_MSS_DropScope

The **proc_MSS_DropScope** stored procedure is called to delete the specified search scope from the search application. The stored procedure MUST set the DefaultScopeID attribute in the Scope

Display Group Set to -1 for the search scope display groups that have the specified search scope set as their default search scope.

Upon successful execution the stored procedure modifies the CompilationState value in the Scope Set. The stored procedure also increments the LastConsumerChangeID value in the Scope System Set and sets the LastUpdate value for the corresponding consumer in the Consumer Set to the resulting LastConsumerChangeID value.

For the specification of Scope System Set, Scope Set, and Scope Display Group Set see section 3.1.1.4.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_DropScope(
   @ScopeID             int,
   @ModifierName        nvarchar(300)
);
```

@ScopeID: The unique identifier of the search scope to delete. This value MUST NOT be NULL.

@ModifierName: The name of the user deleting the search scope. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<37> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.43 proc_MSS_DropScopeDisplayGroup

The **proc_MSS_DropScopeDisplayGroup** stored procedure is called to delete a search scope display group. The stored procedure also deletes entries from the Scope Display Group Membership Set associated with the specified search scope display group, as specified in section 3.1.1.4

Upon successful execution the stored procedure increments the LastConsumerChangeID value in the Scope System Set and sets the LastUpdate value for the corresponding consumer in the Consumer Set to the resulting LastConsumerChangeID value. See section 3.1.1.4 for details.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_DropScopeDisplayGroup(
   @DisplayGroupID         int
);
```

@DisplayGroupID: The unique identifier of the search scope display group. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<38> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.44 proc_MSS_DropScopeRule

The **proc_MSS_DropScopeRule** stored procedure is called to delete an existing search scope rule from the search application.

Upon successful execution the stored procedure modifies the CompilationState value in the Scope Set. The stored procedure also increments the LastConsumerChangeID value in the Scope System Set and sets the LastUpdate value for the corresponding consumer in the Consumer Set to the resulting LastConsumerChangeID value. See section 3.1.1.4 for details.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_DropScopeRule(
   @RuleID                  int,
   @ModifierName            nvarchar(300)
);
```

@RuleID: The unique identifier of the search scope rule to delete. This value MUST NOT be NULL.

@ModifierName: The name of the user deleting the search scope. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<39> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.45 proc_MSS_EndScopeDisplayGroupList

The **proc_MSS_EndScopeDisplayGroupList** stored procedure is called to delete all the search scopes from the specified search scope display group that have positive order and change the negative order of the search scopes contained in the specified search scope display group to its positive value. The stored procedure MUST be called after the last call to proc_MSS_SetScopeDisplayGroupListItem stored procedure. See section 3.1.5.158 for the specification of proc_MSS_SetScopeDisplayGroupListItem stored procedure.

Upon successful execution the stored procedure increments the LastConsumerChangeID value in the Scope System Set and sets the LastUpdate value for the corresponding consumer in the Consumer Set to the resulting LastConsumerChangeID value. See section 3.1.1.4 for details.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_EndScopeDisplayGroupList(
   @DisplayGroupID         int
);
```

@DisplayGroupID: The unique identifier of the search scope display group. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<40> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.46 proc_MSS_FlushAcronyms

The **proc_MSS_FlushAcronyms** stored procedure is called to remove all entries from the Popular Acronym Set and all entries from the Popular Acronyms Term Set, as specified in section 3.1.1.9, that are not present in the result set generated from a call to proc_MSS_GetPopularAcronyms, specified in section 3.1.5.96. The procedure also adds acronyms and tokens to the Popular Acronym

Set and the Popular Acronym Term Set**,** that are present in the result set but do not exist in the corresponding set**.**

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_FlushAcronyms();
```

Return Code Value: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST NOT return any result set.

3.1.5.47 proc_MSS_FlushQueries

The **proc_MSS_FlushQueries** stored procedure is called to remove all entries from the Popular Query Set and all entries from the Popular Query Term Set, as specified in section 3.1.1.9, that are not present in the result set generated from a call to proc_MSS_GetPopularQueries, specified in section 3.1.5.97. The procedure also adds queries and query terms to the Popular Query Set and the Popular Query Term Set**,** that are present in the result set but do not exist in the corresponding set**.**

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_FlushQueries();
```

Return Code Value: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST NOT return any result set.

3.1.5.48 proc_MSS_GetAllBestBets

The **proc_MSS_GetAllBestBets** stored procedure is called to retrieve all best bets that belong to the specified keyword consumer group**.** Optionally, the list of best bets can be filtered based on *@Filter* value.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetAllBestBets(
   @ConsumerGpId       nvarchar(50),
   @Filter             int,
   @Value              nvarchar(2048) = null
);
```

@ConsumerGpId: The unique identifier of the keyword consumer group to which the best bets belong.

@Filter: Specifies which attributes of the best bets are used to apply the @Value filter. The value MUST be of type Best Bets Filter Type as specified in section 2.2.1.2.

@Value: A string that is compared against when performing the filtering. The value MUST be set to NULL to list all best bets for the specified keyword consumer group.

Return Code Values: This stored procedure returns an integer value that MUST be ignored

Result Sets: This procedure MUST return a single result set as specified in section 2.2.4.1. The rows in the result set MUST be sorted by Title in ascending order.

3.1.5.49 proc_MSS_GetAllBestBetsCount

The **proc_MSS_GetAllBestBetsCount** stored procedure is called to calculate the number of best bets for the specified keyword consumer group that match the specified filtering criteria.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetAllBestBetsCount (
   @ConsumerGpId      NVARCHAR(50),
   @Filter            int,
   @Value             NVARCHAR(2048) = null,
   @Count             int output
);
```

@ConsumerGpId: The unique identifier of the keyword consumer group to which the best bets belong.

@Filter: Specifies which attributes of the best bets are used to apply the *@Value* filter. The value MUST be of type Best Bets Filter Type as specified in section 2.2.1.2.

@Value: A string that is compared against when performing the filtering. The value MUST be set to NULL to calculate the number of all best bets for the specified keyword consumer group.

@Count: Upon return from this stored procedure, this parameter MUST be set to the number of best bets.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: This procedure MUST NOT return any result set.

3.1.5.50 proc_MSS_GetAllErrorMessages

The **proc_MSS_GetAllErrorMessages** stored procedure is called to retrieve all entries from the Crawl Error Set, as specified in section 3.1.1.3.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetAllErrorMessages();
```

Return values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return a single result set as specified in section 2.2.4.4.

3.1.5.51 proc_MSS_GetAndResetDocID

The **proc_MSS_GetAndResetDocID** stored procedure is called to reset the data for the item with the specified display URL in the MSSCrawlUrl table ([MS-SRCHTP] section 2.2.5.7) to prepare the item for subsequent reuse in the next crawl. If the item is found in the MSSCrawlUrl table and has the value of its IndexType attribute set to 1 ([MS-SRCHTP] section 2.2.1.15) and the value of DeletePending attribute set to 0 ([MS-SRCHTP] section 2.2.1.16), the procedure MUST set the value of DocPropsMD5 attribute to 1 and the value of DocPropsBlob attribute to NULL.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetAndResetDocID(
```

```
    @File nvarchar(4000),
    @FileHash int,
    @DocID int OUTPUT
);
```

@File: The display URL of the item.

@FileHash: The identifier of the @File string.

@DocID: Upon return from this stored procedure, this parameter MUST be set to the unique identifier of the item being updated. This parameter MUST be set to NULL if the item does not exist in the MSSCrawlUrl table.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<41> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.52 proc_MSS_GetAuthorityPages

The **proc_MSS_GetAuthorityPages** stored procedure is called to retrieve a list of authority pages.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetAuthorityPages();
```

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return the following result set:

3.1.5.52.1 Authority Pages Result Set

The Authority Pages result set returns information about authority pages. The result set MUST contain zero or more rows, each corresponding to a single authority page.

The T-SQL syntax for the result set is as follows:

```
DisplayUrl          nvarchar(2048),
DisplayHash         int,
AuthorityLevel      int;
```

DisplayUrl: URL of the authority page.

DisplayHash: The identifier of the authority page URL.

AuthorityLevel: The authority level of the new authority page.

3.1.5.53 proc_MSS_GetBestBet

The **proc_MSS_GetBestBet** stored procedure is called to retrieve a best bet for the specified URL and keyword consumer group.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetBestBet(
   @ConsumerGpId              nvarchar(50),
   @Url                       nvarchar(2048)
);
```

@ConsumerGpId: The unique identifier of the keyword consumer group.

@Url: The URL of the best bet. If the specified URL contains one or more '#' signs, this stored procedure will attempt to match on the portion of the URL before the first '#' sign, disregarding any differences after the first '#' sign. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return the result set as specified in Section 2.2.4.1.

3.1.5.54 proc_MSS_GetBestBetForSpecialTerm

The **proc_MSS_GetBestBetForSpecialTerm** stored procedure is called to retrieve the information about a best bet for the specified keyword consumer group**,** URL**,** and keyword

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetBestBetForSpecialTerm(
   @ConsumerGpId      nvarchar(50),
   @Url               nvarchar(2048),
   @SpecialTermId     int
);
```

@ConsumerGpId: The unique identifier of the keyword consumer group.

@Url: The URL of the best bet. This value MUST NOT be NULL.

@SpecialTermId: The unique identifier of the keyword**.** This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: This procedure MUST return a single result set as specified in section 2.2.4.1.

3.1.5.55 proc_MSS_GetBestBets

The **proc_MSS_GetBestBets** stored procedure is called to retrieve the list of best bets for the specified keyword.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetBestBets(
   @SpecialTermId      int
);
```

@SpecialTermId: The unique identifier of the keyword**.** This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: This procedure MUST return the following result set.

3.1.5.55.1 GetBestBets By Order Result Set

Best Bets by Order result set returns information about the best bets for the specified keyword sorted by Order value in ascending order. See section 3.1.1.2 for the definition of Order attribute in Best Bet Link Set. The result set contains zero or more rows, each corresponding to a single best bet.

The T-SQL syntax for the result set is as follows:

```
BestBetID           int,
Title               nvarchar(100),
Url                 nvarchar(2048),
Description         nvarchar(500),
Order               int;
```

BestBetID: The unique identifier of the best bet.

Title: The title for the best bet.

Url: The URL of the best bet.

Description: The description of the best bet.

Order: This value MUST be ignored by the protocol client.

3.1.5.56 proc_MSS_GetBestBetsCount

The **proc_MSS_GetBestBetsCount** stored procedure is called to calculate the number of best bets for the specified keyword.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetBestBetsCount(
   @SpecialTermId      int,
   @Count              int OUTPUT
);
```

@SpecialTermId: The unique identifier of the keyword for which the best bets count is retrieved.

@Count: Upon return from this stored procedure, this parameter MUST be set to the number of best bets for the specified keyword.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: This stored procedure MUST NOT return any result set.

3.1.5.57 proc_MSS_GetBestBetsOrder

The **proc_MSS_GetBestBetsOrder** stored procedure is called to retrieve a list of best bets associated with the specified keyword and the information about their priorities within this list. The priority of the best bet is determined by the order in which the best bets appears in the search result**.**

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetBestBetsOrder(
   @SpecialTermId      int
);
```

@SpecialTermId: The unique identifier of the keyword**.** This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: This procedure MUST return the following result set.

3.1.5.57.1 Best Bets Priority Result Set

The Best Bets Priorities result set returns information about the best bets associated with the specified keyword and the information about their priorities within this list. The rows in the result set MUST be sorted by Order in ascending order. See section 3.1.1.2 for definition of Order attribute in Best Bet Link Set.

The T-SQL syntax for the result set is as follows:

```
SpecialTermId          nvarchar(100),
BestBetId              int,
Order                  int;
```

SpecialTermId: This value MUST be ignored by the client.

BestBetId: The unique identifier of the best bet.

Order: This value MUST be ignored by the client.

3.1.5.58 proc_MSS_GetConfigurationProperty

The proc_MSS_GetConfigurationProperty stored procedure is specified in [MS-SQLPGAT2] section 3.1.5.49.

3.1.5.59 proc_MSS_GetConsumers

The **proc_MSS_GetConsumers** stored procedure is called to retrieve a list of names of all search scope consumers.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetConsumers();
```

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: This procedure MUST return the following result set:

3.1.5.59.1 Consumers Result Set

The Consumers result set returns information about search scope consumers. The result set MUST contain zero or more rows, each corresponding to a single search scope consumer.

The T-SQL syntax for the result set is as follows:

```
ConsumerName      nvarchar(60);
```

ConsumerName: The name of the search scope consumer.

3.1.5.60 proc_MSS_GetContainingScopeDisplayGroups

The **proc_MSS_GetContainingScopeDisplayGroups** stored procedure is called to retrieve a list of search scope display groups that contain the specified search scope.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetContainingScopeDisplayGroups(
   @ScopeID         int
);
```

@ScopeID: The unique identifier of the search scope. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: This procedure MUST return the following result set:

3.1.5.60.1 Scope Display Groups For Scope Result Set

The Scope Display Groups For Scope result set returns information about the search scope display groups that contain the specified search scope. The result set will contain zero or more rows, each corresponding to a single search scope display group that contains the specified search scope.

The T-SQL syntax for the result set is as follows:

```
DisplayGroupID int;
```

DisplayGroupID: The unique identifier of the search scope display group

3.1.5.61 proc_MSS_GetContentSourceCrawlLog

The **proc_MSS_GetContentSourceCrawlLog** stored procedure is called to retrieve crawl statistics for the specified content source from the Crawl History Set (section 3.1.1.3).

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetContentSourceCrawlLog(
    @ContentSourceID int,
    @Count int
);
```

@ContentSourceId: The unique identifier of the content source crawled. This value MUST NOT be NULL.

@Count: The maximum number of rows to include in the result set. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: This stored procedure MUST return a single result set as specified in section 2.2.4.2. The result set MUST contain zero or up to @Count rows.

3.1.5.62 proc_MSS_GetContentSources

The **proc_MSS_GetContentSources** stored procedure is called to retrieve a list of all content sources according to Crawl Used Content Source Set (section 3.1.1.3).

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetContentSources()
```

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: This procedure MUST return the following result set:

3.1.5.62.1 Content Sources Used In Crawl Log Result Set

The Content Sources Used In Crawl Log result set returns information about all content sources that are referenced by Crawl Item Set (section 3.1.1.3). The result set MUST contain zero or more rows, each corresponding to the distinct content source unique identifier.

The T-SQL syntax for the result set is as follows:

```
ContentSourceID        int;
```

ContentSourceID: The unique identifier of the content source.

3.1.5.63 proc_MSS_GetCrawledPropertiesAllForCategory

The **proc_MSS_GetCrawledPropertiesAllForCategory** stored procedure is called to retrieve a list of all crawled properties associated with the specified crawled property category from the metadata schema.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetCrawledPropertiesAllForCategory (
   @CategoryName         nvarchar(64)
);
```

@CategoryName: The name for the crawled property category. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return a single result set as specified in section 2.2.4.3. The rows in the result set MUST be ordered by PropertyName and then by Propset in ascending order.

3.1.5.64 proc_MSS_GetCrawledPropertiesForOM

The **proc_MSS_GetCrawledPropertiesForOM** stored procedure is called to retrieve a list of crawled properties associated with the specified crawled property category from the metadata schema. The stored procedure can be called multiple times to retrieve a subset of rows from the list of crawled properties matching the criteria imposed by this stored procedures parameters.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetCrawledPropertiesForOM(
   @CategoryName         nvarchar(64),
   @Filter               nvarchar(440),
   @MaxProps             int,
   @LastPropset          uniqueIdentifier,
   @LastPropertyName     nvarchar(440),
   @ForwardDirection     bit
);
```

@CategoryName: If the value is set to "%%", the result set MUST include crawled properties from any crawled property category; otherwise the result set MUST include crawled properties from the crawled property category matching this parameter value. This value MUST NOT be NULL.

@Filter: The filter value to which crawled property names are matched, according to the Filter Wildcard Rules as defined in Section 2.2.1.10.

@MaxProps: The maximum number of rows to be included in the result set. If equal to or less than zero, the stored procedure MUST return all rows.

@LastPropset: The value of the PropSet column (section 2.2.4.3) from the last row of the result set returned by the previous call to this stored procedure. This parameter value represents the crawled property set identifier used to disambiguate when multiple *@LastPropertyName* values exist in the Crawled Properties Set (section 3.1.1.1). This value MUST be set to null for the first call.

@LastPropertyName: The value of the PropertyName column (section 2.2.4.3) from the last row of the result set returned by the previous call to this stored procedure. This parameter value is used as the offset to determine the next property that will be returned in the result set. This value MUST be set to an empty string for the first call.

@ForwardDirection: If 1, result set MUST be ordered by PropertyName and PropSet columns (section 2.2.4.3) in ascending order; otherwise, it MUST be ordered by descending PropertyName and PropSet.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return a single result set as specified in section 2.2.4.3, subject to the conditions imposed by this stored procedures parameters.

3.1.5.65 proc_MSS_GetCrawledPropertiesUnmappedForCategory

The **proc_MSS_GetCrawledPropertiesUnmappedForCategory** stored procedure is called to retrieve crawled properties associated with the specified crawled property category which have no mappings to the full-text index catalog or the metadata index in the metadata schema. A crawled property is said to have no mappings when its CrawledPropertyId is not listed in the Mappings Set and its IsMappedToContent attribute is set to 0. See section 3.1.1.1 for the specification of Crawled Property Set and Mapping Set.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetCrawledPropertiesUnmappedForCategory (
   @CategoryName         nvarchar(64)
);
```

@CategoryName: The name of the crawled property category for which to retrieve unmapped crawled properties. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return a single result set as specified in section 2.2.4.3. The rows in the result set MUST be ordered by PropertyName and then by PropSet columns in ascending order.

3.1.5.66 proc_MSS_GetCrawledProperty

The **proc_MSS_GetCrawledProperty** stored procedure is called to retrieve a crawled property from the metadata schema.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetCrawledProperty(
   @Propset              uniqueIdentifier,
   @PropertyName         nvarchar(440),
   @VariantType          int
);
```

@Propset: The crawled property set identifier associated with the crawled property category. This value MUST NOT be NULL.

@PropertyName: The name of the crawled property. This value MUST NOT be NULL.

@VariantType: The variant type (2) for the crawled property. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: This stored procedure MUST return a single result set as specified in Section 2.2.4.3. The result set MUST contain zero or one row.

3.1.5.67 proc_MSS_GetCrawledPropertyCategories

The **proc_MSS_GetCrawledPropertyCategories** stored procedure MUST return all of the crawled property categories from the **Crawled Property Category Set**, specified in Section 3.1.1.1.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetCrawledPropertyCategories();
```

Return Code Values: This stored procedure returns an integer value that MUST be ignored

Result Sets: MUST return the following result set:

3.1.5.67.1 GetCrawledPropertyCategories Result Set

The **GetCrawledPropertyCategories** result set contains information about all the crawled property categories. The result set will contain zero or more rows, each corresponding to a crawled property category. The result set MUST be ordered by the *CategoryName* in ascending order.

The T-SQL syntax for the result set is as follows:

```
Propset                        uniqueidentifier,
```

```
CategoryName                    nvarchar(64),
CrawledPropertyCount            int,
DiscoverNewProperties           bit,
MapToContents                   bit,
FullTextQueriable               bit,
Retrievable                     bit,
Scoped                          bit,
MatchExistingManagedProperty    bit,
MatchIgnorePrefix               nvarchar(64),
MatchIgnoreSuffix               nvarchar(64),
MaxIndexedStringLength          int,
MaxNonIndexedStringLength       int,
URINameSpace                    nvarchar(440);
```

Propset: The crawled property set identifier associated with the crawled property category. This value MUST NOT be NULL.

CategoryName: The name of the crawled property category. This value MUST NOT be NULL.

CrawledPropertyCount: The number of crawled properties with this *Propset* in the metadata schema. This value MUST NOT be NULL.

DiscoverNewProperties: This value MUST be set to 1 if the crawled properties within this crawled property category are added to the **Crawled Property Set** automatically. Otherwise, it MUST be set to 0.

MapToContents: This value MUST be set to 1 if the string data from newly discovered crawled properties within this crawled property category is put in the full-text index catalog. Otherwise, it MUST be set to 0.

FullTextQueriable: This value MUST be set to 1 if the string data from newly discovered crawled properties within this category is mapped to a new managed property which will be put in the full-text index catalog. Otherwise, it MUST be set to 0.

Retrievable: This value MUST be set to 1 if string data from newly discovered crawled properties within this category is mapped to a new managed property which will be put in the metadata index. Otherwise, it MUST be set to 0.

Scoped: This value MUST be 1 when a managed property is automatically created to map to a newly discovered crawled property within this crawled property category. Otherwise, it MUST be 0.

MatchExistingManagedProperty: This value MUST be set to 1 if the newly discovered crawled properties within this category need to be mapped to an existing managed property. Otherwise, it MUST be 0.

MatchIgnorePrefix: This value MUST be set to 1, if the managed property prefix has to be ignored when matching a new managed property to newly discovered crawled properties in this crawled property category. Otherwise, it MUST be 0.

MatchIgnoreSuffix: This value MUST be set to 1 if the managed property suffix has to be ignored when matching a new managed property to newly discovered crawled properties in this crawled property category. Otherwise, it MUST be 0.

MaxIndexedStringLength: The maximum number of characters persisted in the string value column in the *MSSDocProps* table defined in [MS-SQLPQ2] section 2.2.5.2. If the string value length

is greater than MaxIndexedStringLength, the string is truncated to MaxIndexedStringLength. This value MUST NOT be NULL.

MaxNonIndexedStringLength: If the string value length in the *MSSDocProps* table, defined in [MS-SQLPQ2] section 2.2.5.2, is greater than the size allowed by the MaxIndexedStringLength, the string is truncated to MaxIndexedStringLength and the string overflow is stored in the strVal2 column, in the *MSSDocProps* table defined in [MS-SQLPQ2] section 2.2.5.2, up to the size allowed by the MaxNonIndexedStringLength. This value MUST NOT be NULL.

URINameSpace: This parameter MUST be ignored by the protocol server.

3.1.5.68 proc_MSS_GetCrawledPropertyID

The **proc_MSS_GetCrawledPropertyID** stored procedure is called to retrieve the identifier of the specified crawled property from the metadata schema.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetCrawledPropertyID(
   @Propset          uniqueIdentifier,
   @PropertyName     nvarchar(440),
   @VariantType      int
);
```

@Propset: The crawled property set identifier associated with the crawled property category. This value MUST NOT be NULL.

@PropertyName: The name of the crawled property. This value MUST NOT be NULL.

@VariantType: The variant type (2) of the crawled property. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return the following result set:

3.1.5.68.1 Crawled Property ID Result Set

The Crawled Property ID result set returns information about the specified crawled property. The result set MUST contain zero or one row.

The T-SQL syntax for the result set is as follows:

```
PropertyId      int;
```

PropertyId: The unique identifier of the crawled property. This value MUST NOT be NULL.

3.1.5.69 proc_MSS_GetCrawledPropertySamplesByPropertyID

The **proc_MSS_GetCrawledPropertySamplesByPropertyID** stored procedure is called to retrieve the list of items associated with the specified crawled property from the Sample Crawled Property Set as specified in Section 3.1.1.1.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetCrawledPropertySamplesByPropertyID (
   @CrawledPropertyId         int,
   @SampleCount               int
);
```

@CrawledPropertyId: The unique identifier of the crawled property. This value MUST NOT be NULL.

@SampleCount: The maximum number of rows returned in Crawled Property Samples result set.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return the following result set:

3.1.5.69.1 Crawled Property Samples Result Set

The Crawled Property Samples Result set contains information about the items associated with the specified of crawled property from the Sample Crawled Property Set. The result set MUST contain zero or up to *@SampleCount* rows. The result set MUST be sorted by SampleURL field in ascending order.

The T-SQL for the result set is as follows:

```
SampleURL      nvarchar(2450);
```

SampleURL: The URL of the item which contains the crawled property. This value MUST NOT be NULL.

3.1.5.70 proc_MSS_GetCrawlHistory

The **proc_MSS_GetCrawlHistory** stored procedure is called to retrieve crawl statistics from the Crawl History Set, as specified in section 3.1.1.3.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetCrawlHistory (
   @MaxRecords          int,
   @BeginTime           datetime,
   @EndTime             datetime,
   @CrawlStatus         int,
   @ContentSourceID     int
);
```

@MaxRecords: The maximum number of rows to include in the result set, or NULL to retrieve all entries from the Crawl History Set that satisfy constraints specified by the stored procedure parameters.

@BeginTime: A date and time in UTC time that MUST be greater than the value of *StartTime* attribute for the entries included in the result set. This value MUST be set to NULL to retrieve crawl statistics for the last 7 days from the current time.

@EndTime: A date and time in UTC time that MUST be less than or equal to the value of *StartTime* attribute for the entries included in the result set. This value MUST be set to NULL if *@BeginTime* is set to NULL.

@CrawlStatus: MUST be a valid value of Crawl Status data type as specified in [MS-SQLPGAT2] section 2.2.1.4, or NULL to indicate that the entries in the result set are not limited by any **crawl status**.

@ContentSourceID: The unique identifier of the content source for which to return crawl statistics, or NULL to indicate that the entries in the result set are not limited by any content source.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return the following result set:

3.1.5.70.1 CrawlHistory Result Set

The CrawlHistory result set returns crawl statistics from the Crawl History Set (section 3.1.1.3). The result set MUST contain zero or more rows each corresponding to an entry in the Crawl History Set The rows in the result set MUST be ordered by EndTime attribute in descending order.

The T-SQL syntax for the result set is as follows:

```
CrawlID             int,
CrawlType           int,
ContentSourceID     int,
Status              int,
StartTime           datetime,
EndTime             datetime,
SuccessCount        int,
ErrorCount          int,
DeleteCount         int;
```

CrawlID: The unique identifier of the crawl.

CrawlType: The type of the crawl. The value MUST be a Crawl Type data type as specified in [MS-SQLPGAT2] section 2.2.1.2.

ContentSourceID: The unique identifier of the content source being crawled.

Status: MUST be a Crawl Status data type as specified in [MS-SQLPGAT2] section 2.2.1.4.

StartTime: A date and time in UTC time indicating when the crawl was started.

EndTime: A date and time in UTC time indicating when the crawl finished.

SuccessCount: The number of items crawled successfully.

ErrorCount: The number of items crawled with errors.

DeleteCount: The number of items removed from the search catalog.

3.1.5.71 proc_MSS_GetCrawlStoreByHostName

The **proc_MSS_GetCrawlStoreByHostName** stored procedure is called to retrieve information from Crawl Store Set about the **crawl store** ([MS-SRCHTP], section 3.1.1.3) that contains items crawled within the specified host name.

```
PROCEDURE proc_MSS_GetCrawlStoreByHostName(
   @HostName nvarchar(300)
```

```
);
```

@HostName: The host name. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: This stored procedure MUST return the following result set:

3.1.5.71.1 Crawl Store Result Set

The Crawl Store result set returns information from Crawl Store Set about the crawl store ([MS-SRCHTP], section 3.1.1.3) that contains items crawled within the specified host name. The result set MUST contain zero or one row, corresponding to a single entry in the Crawl Store Set.

The T-SQL syntax for the result set is as follows:

```
CrawlStoreID        uniqueidentifier,
HostID              int;
```

CrawlStoreId: The unique identifier of the crawl store. This value MUST NOT be NULL.

HostID: The unique identifier of the host. This value MUST NOT be NULL.

3.1.5.72 proc_MSS_GetCurrentLogData

The **proc_MSS_GetCurrentLogData** stored procedure is called to retrieve a list of items and their corresponding crawl statistics from the Crawl Url History Set ([MS-SQLPGAT2] section 3.1.1.2).

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetCurrentLogData (
   @MinDate              datetime = null,
   @MaxDate              datetime = null,
   @MessageType          int = null,
   @ContentSourceId      int = null,
   @MessageId            int = null,
   @Count                int = null,
   @Url                  nvarchar(4000) = null,
   @IsLike               bit = null,
   @HostName             nvarchar(300) = null,
   @MustUseHostName      bit = null,
   @CatalogID            int = null
);
```

@MinDate: A date and time in UTC time that MUST be less than the value of LastModifiedTime attribute for the entries included in the result set. This value MUST be set to NULL to indicate that the entries in the result set are not filtered by LastModifiedTime.

@MaxDate: A date and time in UTC time that MUST be greater than the value of LastModifiedTime attribute for the entries included in the result set. This value MUST be set to NULL to indicate that the entries in the result set are not filtered by LastModifiedTime.

@MessageType: MUST be a valid value of Crawl Error Level data type as specified in [MS-SQLPGAT2] section 2.2.1.10, or NULL to indicate that the entries in the result set are not filtered by ErrorLevel attribute.

@ContentSourceID: The unique identifier of the content source for which to retrieve items, or NULL to indicate that the entries in the result set are not filtered by the content source.

@MessageId: The unique identifier of the error occurred when crawling the item. The value MUST be one of the values listed in the Crawl Error Set (section 3.1.1.3).

@Count: The maximum number of rows to include in the result set, or NULL to retrieve all entries from the Crawl Url History Set that satisfy constraints specified by the stored procedure parameters.

@Url: The display URL of the item to include in the result set, or NULL to indicate that the entries in the result set are not limited by any display URL.

@IsLike: A bit flag specifying how *@Url* value is matched against display URLs. An integer which MUST be one of the values listed in the following table.

Value	Description
NULL	If *@Url* is not NULL, the display URL of the item to include in the result set MUST be an exact match to the *@Url* parameter value.
0	If *@Url* is not NULL, the display URL of the item to include in the result set MUST be an exact match to the *@Url* parameter value.
1	If *@Url* is not NULL, The display URL of the item to include in the result set MUST be match the pattern in the *@Url* parameter value.

@HostName: The host name of the host in Crawl Host Set (section 3.1.1.3) for which to retrieve items, or NULL to indicate that the entries in the result set are not limited by any host.

@MustUseHostName: A bit flag specifying whether the list of items and crawl statistics MUST be retrieved for the specified host name. An integer which MUST be one of the values listed in the following table.

Value	Description
NULL	*@HostName* MAY be NULL.
0	*@HostName* MAY be NULL.
1	*@HostName* MUST NOT be NULL.

@CatalogID: The unique identifier for the **catalog**, which is a Project Identifier data type as specified in [MS-SQLPGAT2] section 2.2.1.1.

Return Code Values: An integer which MUST be one of the values listed in the following table.

Value	Description
0	Successful execution.
50000	Counts are to be retrieved for a specific host name but the host name is not specified or the specified host name does not exist.

Result Sets: MUST return two result sets in the following order:

3.1.5.72.1 Count Result Set

The Count Data result set returns aggregated crawl statistics from the Crawl Url History Set ([MS-SQLPGAT2] section 3.1.1.2) for the entries that satisfy constraints specified by the stored procedure parameters. The result set MUST contain 0 to 3 rows, each row corresponding to a single error level if there is a count greater than 0 for it. The result set MUST NOT be returned when:

- No input parameters are specified.
- Only the *@Count* input parameter is specified.
- Only the *@IsLike* input parameter is specified.
- Only the *@MustUseHostName* input parameter is specified.
- *@HostName* input parameter is specified but does not exist.

The T-SQL syntax for the result set is as follows:

```
ErrorLevel       int,
Cnt              int;
```

ErrorLevel: The error level which MUST be a Crawl Log Error Level data type as specified in section 2.2.1.7.

Cnt: The number of items with the corresponding ErrorLevel value.

3.1.5.72.2 Log Data Result Set

The Log Data Result set returns crawl statistics from the Crawl Url History Set ([MS-SQLPGAT2] section 3.1.1.2) for each entry that satisfies constraints specified by the stored procedure parameters. The result set MUST contain zero or more rows, each row corresponding to an entry in the Crawl Url History Set sorted by LastTouchStart in descending order. The result set MUST be returned only if the *@Count* input parameter is specified.

The T-SQL syntax for the result set is as follows:

```
DisplayUrl           nvarchar(4000),
ErrorLevel           int,
ErrorMsg             nvarchar(2000),
HResult              int,
ErrorDesc            nvarchar(512),
ContentSourceID      int,
LastTouchStart       datetime;
```

DisplayUrl: The display URL of the item.

ErrorLevel: The error level which MUST be a Crawl Log Error Level data type as specified in section 2.2.1.7

ErrorMsg: The descriptive message for the success, warning or error.

HResult: The HRESULT value of the success, warning or error.

ErrorDesc: Additional description for the success, warning or error.

ContentSourceID: The unique identifier of the content source to which the item belongs.

LastTouchStart: The date and time the item was last crawled.

3.1.5.73 proc_MSS_GetDeletedErrorMessages

The **proc_MSS_GetDeletedErrorMessages** stored procedure is called to retrieve all entries from the Crawl Deleted Reason Set, as specified in section 3.1.1.3.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetDeletedErrorMessages();
```

Return values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return the following result set:

3.1.5.73.1 Deleted Error Result Set

The Deleted Error result set returns all entries from the Crawl Deleted Reason Set, as specified in section 3.1.1.3. The result set MUST contain zero or more rows, each row corresponding to a single entry in the Crawl Deleted Reason Set.

The T-SQL syntax for the result set is as follows:

```
DeletedID          int,
ErrorMsg           nvarchar(2000);
DeletedID: The unique identifier of the error.
```

ErrorMsg: The message describing the reason for deleting the item from the search catalog.

3.1.5.74 proc_MSS_GetErrorCrawlLogData

The **proc_MSS_GetErrorCrawlLogData** stored procedure is called to retrieve the error statistics for specified content source and host. If content source is not specified then error statistics among all content sources for a given host MUST be returned. If host is not specified then error statistics among all hosts for a given content source MUST be returned. If neither content source nor host is specified then error statistics among all content sources and all hosts MUST be returned.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetErrorCrawlLogData (
   @ContentSourceID int = null,
   @HostID int = null
);
```

@ContentSourceID: The unique identifier of the content source.

@HostID: The unique identifier of the host.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: This stored procedure MUST return the following result set

3.1.5.74.1 Crawl Log Error Statistics Result Set

The Crawl Log Error Statistics result set returns a number of times each error appears in crawl log. If an error never appeared in crawl log then result set MUST NOT contains an entry for this error.

The T-SQL syntax for the result set is as follows:

```
Count               int,
ErrorID             int
```

Count: The number of times an error appears in crawl log.

ErrorID: The unique identifier of the error.

3.1.5.75 proc_MSS_GetErrorMessages

The **proc_MSS_GetErrorMessages** stored procedure is called to retrieve a list of unique successes, warnings and errors from the Crawl Item Set, as specified in section 3.1.1.3.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetErrorMessages();
```

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return the following result set:

3.1.5.75.1 Error Result Set

The Error result set returns a list of unique successes, warnings and errors from the Crawl Used Error Set, as specified in section 3.1.1.3. The result set MUST contain zero or more rows, each row corresponding to a single entry in the Crawl Used Error Set. Each row in the result set MUST have the unique combination of ErrorID, ErrorLevel and LogLevel values within the result set.

The T-SQL syntax for the result set is as follows:

```
ErrorID             int,
ErrorLevel          int,
LogLevel            int;
```

ErrorID: The unique identifier of the error.

ErrorLevel: The type of the error. The value MUST be a Crawl Log Error Level data type as specified in section 2.2.1.7.

LogLevel: The level of the item in the site hierarchy. The value MUST be a Crawl Log Level data type as specified in section 2.2.1.8.

3.1.5.76 proc_MSS_GetHostCrawlLogData

The **proc_MSS_GetHostCrawlLogData** stored procedure is called to retrieve crawl statistics for the hosts processed by the crawler from the Crawl Host Set (section 3.1.1.3).

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetHostCrawlLogData(
   @Url          NVARCHAR(300) = null,
   @OrderField   NVARCHAR(32)  = null,
   @TopCount     int           = 6,
   @Dir          int           = 0
};
```

@Url: When this value is specified the stored procedure MUST only retrieve hosts whose prefix of the HostName attribute matches the parameter value. If the value is set to NULL the stored procedure MUST retrieve all hosts.

@OrderField: The attribute to use for ordering the result set. If the value is set to NULL the result set MUST be ordered by Total attribute (section 3.1.5.76.1).

@TopCount: The maximum number of rows to include in the result set.

@Dir: The direction of ordering the result set which MUST be one of the values listed in the following table:

Value	Description
0	The result set is sorted in ascending order.
1	The result set is sorted in descending order.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return the following result set:

3.1.5.76.1 Host Summary Result Set

The Host Summary result set returns crawl statistics for the hosts processed by the crawler from the Crawl Host Set (section 3.1.1.3). The result set MUST contain zero or more rows, each corresponding to a single entry in the Crawl Host Set. The rows in the result set MUST be ordered by the attribute specified in *@OrderField* parameter or by *Total* attribute if *@OrderField* value is NULL. The sort order is specified by *@Dir* parameter.

The T-SQL syntax for the result set is as follows:

```
HostID              int,
HostName            nvarchar(300),
Success             int,
ErrorCount          int,
WarningCount        int,
DeleteCount         int,
LevelHighErrorCount int,
Total               int;
```

HostID: The unique identifier of the host.

HostName: The host name.

Success: The number of items crawled successfully.

ErrorCount: The number of items crawled with errors.

WarningCount: The number of items crawled with warnings.

DeleteCount: The number of items removed from the search catalog.

LevelHighErrorCount: The number errors generated when crawling items with LogLevel value set to 2 as specified in section 2.2.1.8.

Total: The sum of Success, ErrorCount, WarningCount and DeleteCount values.

3.1.5.77 proc_MSS_GetHosts

The **proc_MSS_GetHosts** stored procedure is called to retrieve a list of hosts crawled under specified content source according to Crawl Used Content Source Set (section 3.1.1.3). If content source identifier is not passed then stored procedure MUST return a list of all hosts from all content sources.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetHosts(
   @ContentSourceID      int = null
);
```

@ContentSourceID: The unique identifier of the content source.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: This procedure MUST return the following result set:

3.1.5.77.1 Hosts Used In Crawl Log Result Set

The Hosts Used In Crawl Log result set returns information about crawled hosts. The result set MUST contain zero or more rows, each corresponding to the distinct host:

The T-SQL syntax for the result set is as follows:

```
HostID              int;
HostName            nvarchar(300);
```

HostID: The unique identifier of the host. This value MUST NOT be NULL.

HostName: The host name. This value MUST NOT be NULL.

3.1.5.78 proc_MSS_GetItemsCounterPerFileType

The **proc_MSS_GetItemsCounterPerFileType** stored procedure is called to select file extensions in the search index and to retrieve a count of indexed items that include that extension. These MAY be the following extensions, but any set is accceptable:

- doc, dot, obd, odt, pot, docm, docx, dotc
- ppt, pps, pptm, pptx
- xlc, xlb, xls, xlt, xlsm, xlsx, xlsb
- xml
- asp, aspx, hhc, htm, html, htw, htx, odc, stm
- pdf

It MAY be called at any time, or not at all.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetItemsCounterPerFileType();
```

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return the following result set

3.1.5.78.1 Items Per File Extension Result Set

The Transact-Structured Query Language (T-SQL) syntax for the result set is as follows:

FileExtensionnvarchar(4000);

FilesCountint;

FileExtension: The file extension of the indexed item. This value MUST NOT be NULL.

FilesCount: Any integer. This value MUST NOT be NULL.

3.1.5.79 proc_MSS_GetLanguagePhrases

The **proc_MSS_GetLanguagePhrases** stored procedure is called to retrieve information about language resources with the specified **Type** and **Locale** from the Language Resources Set**.** See section 3.1.1.8 for the definition of Language Resources Set.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetLanguagePhrases(
   @Type      smallint,
   @Locale    int
);
```

@Type: The **Type** element of the language resource. The value MUST be a valid Language Resource Type as specified in section 2.2.1.13.

@Locale: The value of the **Locale** element. This value MUST NOT be NULL.

Return Code Values: An integer that MUST be zero.

Result Sets: MUST return the result set specified in the following subsections.

3.1.5.79.1 Language Phrases Result Set

The Language Phrases result set returns information about language resources. The result set MUST contain zero or more rows, each corresponding to a single entry in the Language Resources Set as specified in section 3.1.1.8.

The T-SQL syntax for the result set is as follows:

```
Phrase      nvarchar(128),
Mapping     nvarchar(128),
Type        smallint,
Locale      int;
Author      nvarchar(256);
Created     datetime;
```

Phrase: The **Phrase** element of the language resource. This value MUST NOT be NULL.

Mapping: The **Mapping** element of the language resource.

Type: The **Type** element of the language resource. This value MUST NOT be NULL.

Locale: The **Locale** element of the language resource. This value MUST NOT be NULL.

Author: The **Author** element of the language resource. This value MUST NOT be NULL.

Created: The **Created** element of the language resource. This value MUST NOT be NULL.

3.1.5.80 proc_MSS_GetLanguageResources

The **proc_MSS_GetLanguageResources** stored procedure is called to retrieve a list of distinct values for the Locale element from the Language Resources Set**.** See section 3.1.1.8 for the definition of Language Resources Set**.**

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetLanguageResources();
```

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return the following result set:

3.1.5.80.1 Language Resources Result Set

The Language Resources result set returns a list of **Locale** values from the Language Resources Set (section 3.1.1.8). The result set MUST contain zero or more rows. The **Locale** values in this result set MUST be distinct from one another.

The T-SQL syntax for the result set is as follows:

```
Locale      int;
```

Locale: The LCID of the language resource.

3.1.5.81 proc_MSS_GetLanguageResourcesByLocaleAndType

The **proc_MSS_GetLanguageResourcesByLocaleAndType** stored procedure is called to retrieve information about language resources with the specified **Type** and **Locale** from the Language Resources Set**.** See section 3.1.1.8 for the definition of Language Resources Set.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetLanguageResourcesByLocaleAndType(
   @Locale      int,
   @Type        smallint
);
```

@Locale: The value of the **Locale** element. This value MUST NOT be NULL.

@Type: The **Type** element of the language resource. The value MUST be a valid Language Resource Type as specified in section 2.2.1.13.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return the following result set:

3.1.5.81.1 Language Resources by Locale and Type Result Set

The Language Resources by Locale and Type result set returns information about language resources. The result set MUST contain zero or more rows, each corresponding to a single entry in the Language Resources Set as specified in section 3.1.1.8.

The T-SQL syntax for the result set is as follows:

```
Phrase      nvarchar(128),
Mapping     nvarchar(128);
```

Phrase: The **Phrase** element of the language resource.

Mapping: The **Mapping** element of the language resource.

The Language Resources by Locale and Type result set MUST contain information about those language resources that have non-NULL Phrase and Mapping elements, whose Type element is equal to the @Type parameter specified in the call to proc_MSS_GetLanguageResourcesByLocaleAndType, and whose Locale element is either equal to the @Locale parameter specified in the call to proc_MSS_GetLanguageResourcesByLocaleAndType, or equal to the lower 10 bits of that parameter. The result set MUST NOT contain any information about any other language resources.

3.1.5.82 proc_MSS_GetLanguageResourceUSN

The **proc_MSS_GetLanguageResourceUSN** stored procedure is called to retrieve the USN of the Language Resources Set. See section 3.1.1.8 for the definition of Language Resources Set.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_AddLanguageResource();
```

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: The stored procedure MUST return the following result set:

3.1.5.82.1 GetLanguageResourceUSN Result Set

The GetLanguageResourceUSN result set returns the USN of the Language Resources Set. The result set MUST contain one row.

The T-SQL syntax for the result set is as follows:

```
UpdateSequenceNumber        int;
```

UpdateSequenceNumber: The USN of the Language Resources Set.

3.1.5.83 proc_MSS_GetLastLocationConfigUpdate

The **proc_MSS_GetLastLocationConfigUpdate** stored procedure is called to retrieve the LastConfigurationChange value from the Federation Set as specified in section 3.1.1.7.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetLastLocationConfigUpdate();
```

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return the following result set:

3.1.5.83.1 LastLocationConfigUpdate Result Set

The LastLocationConfigUpdate result set returns the version indicating the most recent change to the federated location configuration or visualization configuration. The result set MUST contain one row.

The T-SQL syntax for the result set is as follows:

```
LastUpdate        bigint;
```

LastUpdate: The LastConfigurationChange of the Federation Set as specified in section 3.1.1.7.

3.1.5.84 proc_MSS_GetListContentSourceCrawlLog

The **proc_MSS_GetListContentSourceCrawlLog** stored procedure is called to retrieve crawl statistics for the specified content sources from the Crawl History Set (section 3.1.1.3).

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetListContentSourceCrawlLog(
    @JoindataIds VARBINARY(MAX),
    @Count int
);
```

@JoindataIds: A list of content sources for which to retrieve crawl statistics. This parameter MUST be encoded in the format described by Id Blob [MS-SQLPQ2] section 2.2.1.2.

@Count: The maximum number of rows to include in the result set.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: This stored procedure MUST return a single Content Source Log result set as specified in section 2.2.4.2. The result set MUST contain zero or up to @Count rows. The rows MUST be sorted in descending order by StartTime.

3.1.5.85 proc_MSS_GetLocationConfigurations

The **proc_MSS_GetLocationConfigurations** stored procedure is called to retrieve configuration information for all the federated locations.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetLocationConfigurations();
```

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return three result sets in the following order:

3.1.5.85.1 Locations Result Set

The Locations result set returns information about all the federated locations. Each row in the result set contains information about a federated location and its associated Authentication Type data type, as specified in Section 2.2.1.1. The result set MUST contain zero or more rows.

The T-SQL syntax for the result set is as follows:

```
Id                          int,
InternalName                nvarchar(60),
DisplayName                 nvarchar(60),
AdminDescription            nvarchar(1024),
LocationType                tinyint,
Author                      nvarchar(60),
Version                     nvarchar(50),
IsDeletable                 bit,
IsPrefixPattern             bit,
QueryReformatPattern        nvarchar(512),
PropertySchema              nvarchar(max),
QueryRestriction            nvarchar(512),
KindsOfResults              nvarchar(max),
Languages                   ntext,
IsRestricted                bit,
AllowedSiteCollectionGuids  nvarchar(max),
CreationDate                datetime,
LastmodifiedDate            datetime,
Type                        tinyint,
Data                        nvarchar(max);
```

Id: The unique identifier of the federated location.

InternalName: The unique internal name of the federated location.

DisplayName: The display name of the federated location.

AdminDescription: The **site collection administrator**'s description of the federated location.

LocationType: Specifies the protocol used to connect to the federated location. The value MUST be a Location Type Data Type as specified in Section 2.2.1.14.

Author: The author of the federated location.

Version: The version number for the federated location. The value MUST contain at least one period (".").

IsDeletable: MUST be set to 0 when the federated location cannot be deleted. Otherwise, it MUST be set to 1.

IsPrefixPattern: A 1-bit number that indicates if the query format pattern for the federated location is a prefix pattern or a regular expression. MUST be 1 when the *@QueryReformatPattern* parameter is used as a prefix match. Otherwise, it MUST be 0, indicating that the *@QueryReformatPattern* parameter is used as a regular expression.

QueryReformatPattern: The pattern for triggering the federated location during a search.

PropertySchema: A binary large object (BLOB) that contains the property schema of the federated location.

QueryRestriction: The supplemental query restrictions to be appended to every search query.

KindsOfResults: This value MUST be ignored by the protocol client.

Languages: The languages supported by the location. It is specified as LCID for the languages, delimited by "#;#".

IsRestricted: A 1-bit number that indicates whether the permissions to modify the federated location is restricted or not. MUST be 1 when this federated location is restricted to only the allowed sites specified by the *AllowedSiteCollectionGuids* parameter. Otherwise, it MUST be 0.

AllowedSiteCollectionGuids: A semi-colon delimited string of site identifiers which are allowed to configure this federated location.

CreationDate: The UTC date and time when the federated location was created.

LastModifiedDate: The UTC date and time when the federated location was last modified.

Type: An 8 bit integer that specifies the type of authentication supported for federation. The value MUST be an Authentication Type data type, as specified in Section 2.2.1.1.

Data: The credentials used for authentication to the federated location.

3.1.5.85.2 Location Templates Result Set

The Location Templates result set contains query templates that are used by federated locations. Each row in the result set contains a query template and the corresponding location ID of the federated location that uses the template. The result set MUST contain zero or more rows.

The T-SQL syntax for the result set is as follows:

```
InternalName        nvarchar(60),
LocationId          int,
```

```
UrlTemplate         nvarchar(2048),
TemplateType        tinyint;
```

InternalName: The Internal Name of the federated location that uses the federated location query template.

LocationId: The unique identifier of the federated location.

UrlTemplate: The template for passing search queries to this federated location.

TemplateType: MUST be 0 when the UrlTemplate specifies how to pass search queries to an OpenSearch engine. Otherwise MUST be 1, indicating that the UrlTemplate the URL of the HTML page that displays results for the search query.

3.1.5.85.3 Last Location Config Update 2 Result Set

The Last Location Config Update 2 result set contains the version number of the most recent federated location configuration change. Each more recent federation configuration change MUST result in a new, larger version number. The result set MUST contain one row.

The T-SQL syntax for the result set is as follows:

```
LastUpdate          bigint,
UseCrawlProxy       bit;
```

LastUpdate: The last version.

UseCrawlProxy: a 1-bit number that indicates whether federated locations are configured to use a proxy when retrieving search results. This flag applies to all federated locations.

3.1.5.86 proc_MSS_GetLocationDescription

The **proc_MSS_GetLocationDescription** stored procedure is called to retrieve the federated location definition**.**

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetLocationDescription (
   @LocationId          int
);
```

@LocationId: The unique identifier of the federated location. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return the following result set:

3.1.5.86.1 Location Description Result Set

Location Description Result Set returns the XML that describes the specified federated location. The result set MUST contain zero or one row.

The T-SQL syntax for the result set is as follows:

```
LocationId        int,
Xml               nvarchar(max);
```

LocationId: The unique identifier of the specified federated location.

Xml: The federated location definition of the specified federated location.

3.1.5.87 proc_MSS_GetLocationVisualisations

The **proc_MSS_GetLocationVisualisations** stored procedure is called to retrieve visualizations associated with the specified federated location.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetLocationVisualisations (
   @LocationId               int,
   @VisualisationName        nvarchar(60) = NULL
);
```

@LocationId: The identifier of the federated location. This value MUST NOT be NULL.

@VisualisationName: The name of the visualization to be retrieved. If this parameter is NULL or empty string, all visualizations of the federated location MUST be retrieved.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return the following result set:

3.1.5.87.1 Location Visualisation Result Set

The Location Visualisation Result Set returns information about visualizations associated with the federated location. The result set MUST contain zero or more rows.

The T-SQL syntax for the result set is as follows:

```
LocationId         int,
VisualisationName  nvarchar(60),
Properties         nvarchar(max),
Xsl                nvarchar(max),
SampleData         nvarchar(max);
```

LocationId: The identifier of the specified federated location.

VisualisationName: The name of the visualization.

Xsl: The Xsl for this visualization. This MUST be an Xsl Data Type as specified in Section 2.2.1.22.

Properties: The properties for this visualization. This MUST be a Properties Data Type as specified in Section 2.2.1.16.

SampleData: The sample data for this visualization. This MUST be a SampleData Data Type as specified in Section 2.2.1.17.

3.1.5.88 proc_MSS_GetManagedPropertyAliasesByPid

The **proc_MSS_GetManagedPropertyAliasesByPid** stored procedure is called to list the aliases for a managed property from the metadata schema**.**

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetManagedPropertyAliasesByPid(
   @PID         int
);
```

@PID: The unique identifier of the managed property. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return the following result set:

3.1.5.88.1 Managed Property Aliases Result Set

The Managed Property Aliases result set returns information about aliases for a managed property. The result set MUST contain zero or more rows.

The T-SQL syntax for the result set is as follows:

```
alias            nvarchar(2048);
```

alias: An alternate string name which identifies a managed property. This value MUST NOT be NULL.

3.1.5.89 proc_MSS_GetManagedPropertyDocsPerPidCount

The **proc_MSS_GetManagedPropertyDocsPerPidCount** stored procedure is called to get the count of items in the metadata index that contain the specified managed property**.**

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetManagedPropertyDocsPerPidCount (
   @PID             int,
   @Limit           int,
   @FoundCount      int OUTPUT
);
```

@PID: The unique identifier of the managed property. This value MUST NOT be NULL.

@Limit: This parameter MUST be ignored by the server.

@FoundCount: Upon return from this stored procedure, this parameter MUST be set to the count of items that contain the specified managed property**.**

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: The stored procedure MUST NOT return any result sets.

3.1.5.90 proc_MSS_GetManagedPropertySamples

The **proc_MSS_GetManagedPropertySamples** stored procedure is called to list sample values for a managed property from the metadata schema**.**

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetManagedPropertySamples (
   @PID             int,
   @SampleCount     int
);
```

@PID: The unique identifier of the managed property. This value MUST NOT be NULL.

@SampleCount: The maximum number of rows allowed in the result set.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return the following result set:

3.1.5.90.1 Managed Property Samples Result Set

The Managed Property Samples result set returns information about sample values for a string type managed property ordered by the item URL. The result set MUST contain zero or more rows up to the maximum specified in *@SampleCount*.

The T-SQL syntax for the result set is as follows:

```
Url       nvarchar(4000);
```

Url: The URL of the item which contains the managed property. This value MUST NOT be NULL.

3.1.5.91 proc_MSS_GetMappedCrawledProperties

The **proc_MSS_GetMappedCrawledProperties** stored procedure is called to get a list of crawled properties mapped to the specified managed property in the metadata schema Mappings Set (section 3.1.1.1).

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetMappedCrawledProperties (
   @pid                      int,
   @resultsCount             int
);
```

@pid: The unique identifier of the managed property.

@resultsCount: The maximum number of rows allowed in the result set**.**

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return a single result set as specified in section 2.2.4.3. The number of rows in the result set MUST be limited to the value specified in @resultCount parameter. The rows in the result set MUST be ordered by ascending mapping order.

3.1.5.92 proc_MSS_GetMappingsForCrawledProperty

The **proc_MSS_GetMappingsForCrawledProperty** stored procedure is called to get a list of managed properties mapped to a crawled property in the metadata schema Mappings Set**.**

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetMappingsForCrawledProperty (
   @Propset                  uniqueidentifier,
   @PropertyName             nvarchar(440),
   @VariantType              int
);
```

@Propset: The crawled property set identifier associated with the crawled property category**.** This value MUST NOT be NULL.

@PropertyName: The name of the crawled property. This value MUST NOT be NULL.

@VariantType: The variant type (2) for the crawled property. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST NOT return any result sets if a crawled property with the specified *@Propset, @PropertyName* and *@VariantType* doesn't exist in the specified crawled property set. Otherwise MUST return the following result set specified in the following subsections.

3.1.5.92.1 Crawled Property Mappings Result Set

The Crawled Property Mappings result set returns information about the managed properties to which the crawled property is mapped. The result set MUST contain zero or more rows sorted by *FriendlyName* in ascending order.

The T-SQL syntax for the result set is as follows:

```
FriendlyName        nvarchar(64);
```

FriendlyName: The name of the managed property in a user-readable form. This value MUST NOT be NULL.

3.1.5.93 proc_MSS_GetMappingsForMangedProperty

The **proc_MSS_GetMappingsForMangedProperty** stored procedure is called to get list of crawled properties mapped to a managed property in the metadata schema Mappings Set (section 3.1.1.1).

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetMappingsForMangedProperty (
   @pid          int
);
```

@PID: The unique identifier of the managed property. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return the following result set:

3.1.5.93.1 Managed Property Mappings Result Set

The Managed Property Mappings result set returns information about the crawled properties mapped to the specified managed property. The result set MUST contain zero or more rows ordered by ascending mapping order, each corresponding to a single of crawled property.

The T-SQL syntax for the result set is as follows:

```
Propset              uniqueidentifier,
PropertyName         nvarchar(440),
VariantType          int;
```

Propset: The crawled property set identifier associated with the crawled property category**.** This value MUST NOT be NULL.

PropertyName: The name of the crawled property**.** This value MUST NOT be NULL.

VariantType: The variant type (2) for the crawled property. This value MUST NOT be NULL.

3.1.5.94 proc_MSS_GetNDayAvgCrawlHistoryStats

The **proc_MSS_GetNDayAvgCrawlHistoryStats** stored procedure is called to retrieve the average crawl statistics from the Crawl History Set (section 3.1.1.3) for the specified content source within the specified timeframe.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetNDayAvgCrawlHistoryStats (
   @ContentSourceId        int,
   @CrawlType              int,
   @NumberOfDays           int
);
```

@ContentSourceId: The unique identifier of the content source crawled. This value MUST NOT be NULL.

@CrawlType: The type of the crawl. The value MUST be a Crawl Type data type as specified in [MS-SQLPGAT2] section 2.2.1.2.

@NumberOfDays: The number of days for which the average statistics need to be calculated. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return the following result set:

3.1.5.94.1 Average Statistics Result Set

The Average Statistics result set returns the average crawl statistics from the Crawl History Set as specified in section 3.1.1.3, for the specified content source within the specified timeframe. The result set MUST contain exactly one row.

The T-SQL syntax for the result set is as follows:

```
DurationAvg             int,
AvgSuccess              int,
AvgError                int,
TotalCrawls             int;
```

DurationAvg: The average duration, in seconds, of the completed crawls.

AvgSuccess: The average number of items that were crawled successfully.

AvgError: The average number of items crawled with errors.

TotalCrawls: The total number of crawls.

3.1.5.95 proc_MSS_GetPastLogData

The **proc_MSS_GetPastLogData** stored procedure is called to retrieve the crawl information for the specified item from the MssCrawlUrlLog table ([MS-SRCHTP] section 2.2.5.8).

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetPastLogData (
   @Url    nvarchar(4000)
);
```

@Url: The display URL of the item.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return the following result set.

3.1.5.95.1 Log Data Result Set

The Log Data result set returns the crawl information for the specified item from the Crawl Item Set, as specified in section 3.1.1.3. The result set MUST contain zero or more rows, each row corresponding to a single entry in the Crawl Item Set. The rows in the result set MUST be sorted in ascending order by *LastTouchStart*.

The T-SQL syntax for the result set is as follows:

```
DisplayUrl              nvarchar(4000),
ErrorID                 int,
ErrorMsg                nvarchar(2000),
{NullValue}             null,
LastTouchStart          datetime;
```

DisplayUrl: The specified URL.

ErrorID: The unique identifier of the error occurred when the item was processed by crawler. The value MUST be one of the values listed in the Crawl Error Set, section 3.1.1.3.

ErrorMsg: The descriptive error message for the *ErrorID*.

{NullValue}: This value MUST be ignored by the client.

LastTouchStart: The date and time in UTC time the error occurred when the item was processed by crawler.

3.1.5.96 proc_MSS_GetPopularAcronyms

The **proc_MSS_GetPopularAcronyms** stored procedure is called to retrieve the information about popular acronyms, as specified in section 3.1.1.9, which are used for query suggestions.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetPopularAcronyms();
```

Return Value: This stored procedure returns an integer value that MUST be ignored

Result Sets: MUST return the following result set:

3.1.5.96.1 GetPopularAcronyms Result Set

The **GetPopularAcronyms** result set returns information about the popular acronyms. The result set MUST contain zero or more rows each corresponding to a single acronym.

The T-SQL syntax for the result set is as follows:

```
Term                nvarchar(40),
Sentence            nvarchar(255),
TermOffset          int,
TermLength          int,
Occurrence          int;
```

Term: The name of the acronym. This value MUST NOT be NULL.

Sentence: The sentence containing the term. This value MUST NOT be NULL.

TermOffset: The offset of term in the containing sentence. This value MUST NOT be NULL.

TermLength: The length of term in the containing sentence. This value MUST NOT be NULL.

Occurrence: The frequency of occurrence of the term. This value MUST NOT be NULL.

3.1.5.97 proc_MSS_GetPopularQueries

The **proc_MSS_GetPopularQueries** stored procedure is called to retrieve the list of popular queries as specified in section 3.1.1.9, which are used for query suggestions.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetPopularQueries();
```

Return Code Value: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return the following result set:

3.1.5.97.1 GetPopularQueries Result Set

The **GetPopularQueries** result set returns information about the popular queries. The result set MUST contain zero or more rows, each corresponding to a single search query.

The T-SQL syntax for the result set is as follows:

```
QueryId                 int,
QueryCount              int,
QueryString             nvarchar(1024),
LCID                    int;
```

QueryId: The identifier associated with the query. This value MUST NOT be NULL.

QueryCount: The number of times the query has been executed. This value MUST NOT be NULL.

QueryString: The query text.

LCID: The LCID of the **locale** in which this query was executed. This value MUST NOT be NULL.

3.1.5.98 proc_MSS_GetQuerySuggestionCandidates

The **proc_MSS_GetQuerySuggestionCandidates** stored procedure is called to get the list of queries from the Popular Query Set, specified in section 3.1.1.9.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetQuerySuggestionCandidates();
```

Return Code Value: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return the following result set:

3.1.5.98.1 GetQuerySuggestionCandidates Result Set

The **GetQuerySuggestionCandidates** result set returns the list of queries from the Popular Query Set, specified in section 3.1.1.9.. The result set MUST contain zero or more rows that are ordered by the queryString in ascending order.

The T-SQL syntax for the result set is as follows:

```
QueryString         nvarchar(1024);
```

QueryString: The query text.

3.1.5.99 proc_MSS_GetQuerySuggestions

The **proc_MSS_GetQuerySuggestions** stored procedure is called to retrieve the list of query suggestions, specified in section 3.1.1.9, for a given query.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetQuerySuggestions (
```

```
    @q                nvarchar(1024),
    @qHash            int,
    @nTerms           int,
    @nMaxSuggestions  int,
    @LastTerm         nvarchar(256),
    @qHash0           int,
    @qHash1           int,
    @qHash2           int,
    @qHash3           int,
    @qHash4           int,
    @qHash01          int,
    @qHash12          int,
    @qHash23          int,
    @qHash34          int
);
```

@q: The full query text.

@qHash: The identifier of the query text.

@nTerms: The number of valid tokens extracted from the query text.

@nMaxSuggestions: The maximum number of suggestions for the input query text. This value MUST NOT be NULL.

@LastTerm: The last valid token extracted from the query text.

@qHash0: The identifier of a valid query term. This value MUST NOT be NULL.

@qHash1: The identifier of a valid query term. This value MUST NOT be NULL.

@qHash2: The identifier of a valid query term. This value MUST NOT be NULL.

@qHash3: The identifier of a valid query term. This value MUST NOT be NULL.

@qHash4: The identifier of a valid query term. This value MUST NOT be NULL.

@qHash01: The identifier of the concatenation of adjacent query terms. This value MUST NOT be NULL.

@qHash12: The identifier of the concatenation of adjacent query terms. This value MUST NOT be NULL.

@qHash23: The identifier of the concatenation of adjacent query terms. This value MUST NOT be NULL.

@qHash34: The identifier of the concatenation of adjacent query terms. This value MUST NOT be NULL.

Return Code Value: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return the following result set:

3.1.5.99.1 GetQuerySuggestions Result Set

The **GetQuerySuggestions** result set returns the list of query suggestions, specified in section 3.1.1.9, for the specified query. The result set MUST contain zero or more rows, each corresponding

to a single query suggestion. The result set MUST be ordered by the number of query terms in the specified query text and the number of times the query is executed, in descending order.

The T-SQL syntax for the result set is as follows:

```
QueryString             nvarchar(1024),
tcnt                    int,
termToReplace           nvarchar(1000),
acronymToMatch          nvarchar(50),
QueryCount              int,
mtcnt                   int,
xcnt                    int;
```

QueryString: The query suggestion string.

tcnt: The number of distinct query terms associated with the query text. This value MUST NOT be NULL.

termToReplace: The query term string that is used to determine the exact term count.

acronymToMatch: The acronym string which is matched against the query suggestion string.

QueryCount: The number of times the query is executed. This value MUST NOT be NULL.

mtcnt: MUST be 0. This value MUST be ignored by the protocol client.

xcnt: The sum of the common query terms and the merged query terms. This value MUST NOT be NULL.

3.1.5.100 proc_MSS_GetRankingModels

The proc_MSS_GetRankingModels stored procedure returns all of the ranking models from the ranking model set.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetRankingModels();
```

Return values: This stored procedure returns an integer value that MUST be ignored

Result Sets: This procedure MUST return Get Ranking Models Result Set.

3.1.5.100.1 Get Ranking Models Result Set

The Get Ranking Models Result Set contains information about ranking models from the ranking model set. Each row in the result set corresponds to a ranking model. The Get Ranking Models Result Set MUST be sorted by the ranking model identifier in ascending order.

The T-SQL syntax for the result set is as follows:

```
ModelId     uniqueidentifier NOT NULL,
IsDefault   bit              NOT NULL,
ModelXml    ntext            NOT NULL;
```

ModelId: The ranking model identifier.

IsDefault: The flag indicating whether the ranking model is the default.

ModelXml: The string representation of the ranking model's weights and parameters.

3.1.5.101 proc_MSS_GetSchemaParameter

The **proc_MSS_GetSchemaParameter** stored procedure is called to retrieve the specified schema parameter from the metadata schema.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetSchemaParameter (
   @ParamName                nvarchar(40),
   @IsString                 bit OUTPUT,
   @strValue                 nvarchar(256) OUTPUT,
   @fltValue                 float OUTPUT
);
```

@ParamName: The name of the schema parameter to be retrieved.

@IsString: Upon return from this stored procedure, this parameter MUST be set to 1 if @strValue is set to the value of the schema parameter. Otherwise, if MUST be set to 0.

@strValue: Upon return from this stored procedure, this parameter MUST be set to the value of the schema parameter if @IsString is 1. Otherwise, @strValue MUST be ignored.

@fltValue: Upon return from this stored procedure, this parameter MUST be set to the value of the schema parameter if @IsString is 0. Otherwise, @fltValue MUST be ignored.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Set: SHOULD NOT<42> return a result set. The protocol client MUST ignore any result sets returned by this stored procedure

3.1.5.102 proc_MSS_GetSchemaRankingParameters

The **proc_MSS_GetSchemaRankingParameters** stored procedure is called to retrieve values for ranking parameters from the Global Ranking Parameters Set as specified in section 3.1.1.6.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetSchemaRankingParameters();
```

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return the following result set:

3.1.5.102.1 Schema Parameters Result Set

The Schema Parameters result set returns information about ranking parameters. The result set MUST contain zero or more rows, each corresponding to a single entry in the Global Ranking Parameters Set (section 3.1.1.6).

The T-SQL syntax for the result set is as follows:

```
ParamName        nvarchar(40),
IsString         bit,
strValue         nvarchar(256),
fltValue         float;
```

ParamName: Name of the ranking parameter. The value MUST be one of the values listed in the following table:

Parameter	Description
UseInterSiteAnchorText	Determines whether to consider anchorlinks to other sites.
PrecalcTopSize	Determines the number of items to use during precalculation. This parameter MUST be ignored if *fltValue* of *UsePrecalculation* is 0.
UsePrecalculation	Determines whether to use precalculated rank data.

IsString: If set to 1, the *strValue* field MUST contain the value of the parameter; otherwise the *fltValue* field MUST contain the value of the parameter.

strValue: A string value of the ranking parameter. This field MUST be ignored when *IsString* equals 0.

fltValue: A floating-point value of the ranking parameter. This field MUST be ignored when *IsString* is set to 1.

3.1.5.103 proc_MSS_GetScopeDisplayGroupIDFromName

The **proc_MSS_GetScopeDisplayGroupIDFromName** stored procedure is called to retrieve the identifier of a search scope display group with the specified search scope display group name and the search scope consumer name.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetScopeDisplayGroupIDFromName (
   @ConsumerName                nvarchar(60),
   @Name                        nvarchar(60),
   @DisplayGroupID              int OUTPUT
);
```

@ConsumerName: The name of the search scope consumer who is the owner of the search scope display group. This value MUST NOT be NULL.

@Name: The name of the search scope display group. This value MUST NOT be NULL.

@DisplayGroupID: Upon return from this stored procedure, this parameter MUST be set to the unique identifier of the search scope display group. This parameter MUST be set to NULL if the search scope display group does not exist.

Return Code Values: This stored procedure returns an integer value that MUST be ignored

Result Sets: This procedure MUST NOT return any result set.

3.1.5.104 proc_MSS_GetScopeDisplayGroupInfo

The **proc_MSS_GetScopeDisplayGroupInfo** stored procedure is called to retrieve information about a search scope display group.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetScopeDisplayGroupInfo (
   @DisplayGroupID            int,
   @Name                      nvarchar(60) OUTPUT,
   @Description               nvarchar(300)OUTPUT,
   @ConsumerName              nvarchar(60)OUTPUT,
   @DisplayInAdminUI          bit OUTPUT,
   @Undeletable               bit OUTPUT,
   @DefaultScopeID            int OUTPUT,
   @LastModifiedTime          datetime OUTPUT,
   @LastModifiedBy            nvarchar(60) OUTPUT
);
```

@DisplayGroupID: The unique identifier of the search scope display group to be retrieved. This value MUST NOT be NULL.

@Name: Upon return from this stored procedure, this parameter MUST be set to the name of the search scope display group.

@Description: Upon return from this stored procedure, this parameter MUST be set to the description of the search scope display group.

@ConsumerName: Upon return from this stored procedure, this parameter MUST be set to the name of the search scope consumer who is the owner of the search scope display group.

@DisplayInAdminUI: Upon return from this stored procedure, this parameter MUST be set to a bit flag indicating if the search scope display group is displayed in Administration user interface. The parameter value MUST be a DisplayInAdminUI data type as specified in section 2.2.1.9.

@Undeletable: Upon return from this stored procedure, this parameter MUST be set to a bit flag indicating if the search scope display group can be deleted. The parameter value MUST be an Undeletable data type as specified in section 2.2.1.20**.**

@DefaultScopeID: Upon return from this stored procedure, this parameter MUST be set to the unique identifier of the default search scope for the search scope display group.

@LastModifiedTime: Upon return from this stored procedure, this parameter MUST be set to the date and time of the last change to the search scope display group.

@LastModifiedBy: Upon return from this stored procedure, this parameter MUST be set to the name of the user who last changed the search scope display group.

Return Code Values: An integer which MUST be listed in the following table.

Value	Description
0	Successful execution.
1	The search scope display group does not exist.

Result Sets: This procedure MUST NOT return any result set.

3.1.5.105 proc_MSS_GetScopeDisplayGroupListInfo

The **proc_MSS_GetScopeDisplayGroupListInfo** stored procedure is called to retrieve entries from the Scope Display Group Membership Set associated with the specified search scope display group, as specified in section 3.1.1.4

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetScopeDisplayGroupListInfo (
   @DisplayGroupID            int
);
```

@DisplayGroupID: The unique identifier of the search scope display group. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: This procedure MUST return the following result set:

3.1.5.105.1 Scope Display Group Membership Result Set

The Scopes Display Group Membership Result set returns information about the search scopes that the specified search scope display group contains. The result set will contain zero or more rows, each corresponding to a single search scope. The rows in the result set MUST have non-negative values of Rank attribute in the Scope Display Group Membership Set, as specified in section 3.1.1.4, and MUST be sorted by Rank value in ascending order.

The T-SQL syntax for the result set is as follows:

```
ScopeID         int;
```

ScopeID: The unique identifier of the search scope.

3.1.5.106 proc_MSS_GetScopeDisplayGroupsCount

The **proc_MSS_GetScopeDisplayGroupsCount** stored procedure is called to retrieve the total number of search scope display groups.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetScopeDisplayGroupsCount (
   @Count             int OUTPUT
);
```

@Count: Upon return from this stored procedure, this parameter MUST be set to the total number of search scope display groups.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: This procedure MUST NOT return any result set.

3.1.5.107 proc_MSS_GetScopeDisplayGroupsForConsumer

The **proc_MSS_GetScopeDisplayGroupsForConsumer** stored procedure is called to retrieve a list of all search scope display groups owned by the specified search scope consumer.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetScopeDisplayGroupsForConsumer (
   @ConsumerName         nvarchar(60)
);
```

@ConsumerName: The name of the search scope consumer. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: This procedure MUST return a single result set as specified in section 2.2.4.5.

3.1.5.108 proc_MSS_GetScopeDisplayGroupsInfo

The **proc_MSS_GetScopeDisplayGroupsInfo** stored procedure is called to retrieve a list of all search scope display groups.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetScopeDisplayGroupsInfo();
```

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: This procedure MUST return a single result set as specified in section 2.2.4.5.

3.1.5.109 proc_MSS_GetScopeIDFromName

The **proc_MSS_GetScopeIDFromName** stored procedure is called to retrieve the identifier of a search scope with the specified name and search scope consumer.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetScopeIDFromName (
   @ConsumerName                 nvarchar(60),
   @Name                         nvarchar(60),
   @ScopeID                      int OUTPUT
);
```

@ConsumerName: The name of the search scope consumer that owns the search scope. This value MUST NOT be NULL.

@Name: The name of the search scope. This value MUST NOT be NULL.

@ScopeID: Upon return of this stored procedure, this parameter MUST be set to the unique identifier of the search scope if a search scope with specified name exists. Otherwise this parameter MUST be set to NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: This procedure MUST NOT return any result sets.

3.1.5.110 proc_MSS_GetScopeInfo

The **proc_MSS_GetScopeInfo** stored procedure is called to retrieve the information about the search scope with the specified identifier.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetScopeInfo(
   @ScopeID                 int,
   @Name                    nvarchar(60) OUTPUT,
   @Description             nvarchar(300) OUTPUT,
   @ConsumerName            nvarchar(60) OUTPUT,
   @DisplayInAdminUI        bit OUTPUT,
   @AlternateResultsPageURL nvarchar(2047) OUTPUT,
   @CompilationType         smallint OUTPUT,
   @CompilationState        smallint OUTPUT,
   @LastCompilationTime     datetime OUTPUT,
   @LastModifiedTime        datetime OUTPUT,
   @LastModifiedBy          nvarchar(60) OUTPUT,
   @Filter                  varchar(max) OUTPUT
);
```

@ScopeID: The unique identifier of the search scope. This value MUST NOT be NULL.

@Name: Upon return from this stored procedure, this parameter MUST be set to the name of the search scope.

@Description: Upon return from this stored procedure, this parameter MUST be set to the description of the search scope.

@ConsumerName: Upon return from this stored procedure, this parameter MUST be set to the name of the search scope consumer who owns the search scope.

@DisplayInAdminUI: Upon return from this stored procedure, this parameter MUST be set to a bit flag indicating if the search scope is displayed in the Administration UI. The value MUST be a valid DisplayInAdminUI data type as specified in section 2.2.1.9.

@AlternateResultsPageUrl: Upon return from this stored procedure, this parameter is set to the alternate URL of the web page for the search scope. This parameter can be set to NULL.

@CompilationType: Upon return from this stored procedure, this parameter MUST be set to the compilation type of the search scope. The value MUST be a valid Compilation Type data type as specified in section 2.2.1.5.

@CompilationState: Upon return from this stored procedure, this parameter MUST be set to the search scope compilation state of the search scope. The value MUST be a valid Compilation State data type as specified in section 2.2.1.4.

@LastCompilationTime: Upon return from this stored procedure, this parameter is set to the date and time in UTC time of the last search scope compilation. This parameter can be set to NULL

@LastModifiedTime: Upon return from this stored procedure, this parameter MUST be set to the date and time in UTC time of the last change to the search scope.

@LastModifiedBy: Upon return from this stored procedure, this parameter MUST be set to the name of the person who last changed the search scope.

@Filter: Reserved. This parameter MUST be ignored by the client.

Return Code Values: An integer which MUST be listed in the following table:

Value	Description
0	Successful execution.
1	search scope with the specified identifier was not found

Result Sets: This procedure MUST NOT return any result sets.

3.1.5.111 proc_MSS_GetScopeRuleInfo

The **proc_MSS_GetScopeRuleInfo** stored procedure is called to get the information about a search scope rule given its identifier.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetScopeRuleInfo(
   @RuleID              int,
   @FilterBehavior      smallint OUTPUT,
   @RuleType            smallint OUTPUT,
   @UrlRuleType         smallint OUTPUT,
   @PropertyID          int OUTPUT,
   @UserValueString     nvarchar(2047)OUTPUT
);
```

@RuleID: The unique identifier of the search scope rule. This value MUST NOT be NULL.

@FilterBehavior: Upon return from this stored procedure, this parameter MUST be set to a valid value of a ScopeFilterBehavior data type as specified in section 2.2.1.18.

@RuleType: Upon return from this stored procedure, this parameter MUST be set to a valid value of a ScopeRuleType data type as specified in section 2.2.1.19.

@UrlRuleType: Upon return from this stored procedure, this parameter MUST be set to a valid value of an UrlRuleType data type as specified in section 2.2.1.21 if the value of @RuleType parameter is set to 1. Otherwise, the value of this parameter MUST be ignored by the client. See Scope Rule Set in section 3.1.1.4 for more information.

@ PropertyID: Upon return from this stored procedure, this parameter MUST be set to the unique identifier of the managed property associated with the search scope rule if the value of @RuleType parameter is set to 2. Otherwise, the value of this parameter MUST be ignored by the client. See Scope Rule Set in section 3.1.1.4 for more information.

@UserValueString: Upon return from this stored procedure, this parameter MUST be set to the search scope rule value of the rule.

Return Code Values: An integer which MUST be listed in the following table:

Value	Description
0	Successful execution.
1	Search scope rule with the specified identifier was not found.

Result Sets: This procedure MUST NOT return any result set.

3.1.5.112 proc_MSS_GetScopeRulesCount

The **proc_MSS_GetScopeRulesCount** stored procedure is called to get the count of search scope rules defined for a search scope in the search application.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetScopeRulesCount (
   @ScopeID          int,
   @Count            int OUTPUT
);
```

@ScopeID: The unique identifier of the search scope. This value MUST NOT be NULL.

@Count: Upon return from this stored procedure, this parameter MUST be set to the count of search scope rules defined for the search scope. This value MUST NOT be NULL.

Return Code Values: An integer which MUST be listed in the following table:

Value	Description
0	Successful execution.
1	Search scope with the specified identifier was not found.

Result Sets: This procedure MUST NOT return any result set.

3.1.5.113 proc_MSS_GetScopeRulesInfo

The **proc_MSS_GetScopeRulesInfo** stored procedure is called to get the details of all the search scope rules defined for a search scope in the search application.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetScopeRulesInfo (
   @ScopeID             int
);
```

@ScopeID: The unique identifier of the search scope. This value MUST NOT be NULL.

Return Code Values: An integer which MUST be listed in the following table:

Value	Description
0	Successful execution.

Value	Description
1	Search scope with the specified identifier was not found in the search application.

Result Sets: This procedure MUST return a GetScopeRulesInfo result set.

3.1.5.113.1 GetScopeRulesInfo Result Set

The GetScopeRulesInfo result set returns information about the search scope rules. The result set MUST contain zero or more rows, each row corresponding to a single search scope rule.

The T-SQL syntax for the result set is as follows:

```
RuleID           int,
FilterBehavior   smallint,
RuleType         smallint,
UrlRuleType      smallint,
PropertyID       int,
UserValueString  nvarchar(2048);
```

RuleID: An integer that uniquely identifies the search scope rule.

FilterBehavior: The filter behavior of the search scope rule. The value MUST be a ScopeFilterBehavior data type, as specified in section 2.2.1.18.

RuleType: The type of the search scope rule. The value MUST be a ScopeRuleType data type, as specified in section 2.2.1.19.

UrlRuleType: The URI type of the search scope rule. When the value of the RuleType column is set to 1, this value MUST be a valid UrlRuleType data type, as specified in section 2.2.1.21. Otherwise, the client MUST ignore this value. For more information, see the definition of Scope Rule Set in section 3.1.1.4.

PropertyID: If the value of the RuleType column is set to 2, this value MUST be set to the unique identifier of the managed property used by the search scope rule. Otherwise, the client MUST ignore this value. For more information, see the definition of Scope Rule Set in section 3.1.1.4.

UserValueString: The **search scope rule value** used by the search scope rule.

3.1.5.114 proc_MSS_GetScopesCount

The **proc_MSS_GetScopesCount** stored procedure is called to get the count of all search scopes defined in the search application.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetScopesCount(
   @Count            int OUTPUT
);
```

@Count: Upon return from this stored procedure, this parameter MUST be set to the count of search scopes defined in the search application.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: This procedure MUST NOT return any result set.

3.1.5.115 proc_MSS_GetScopesForConsumer

The **proc_MSS_GetScopesForConsumer** stored procedure is called to retrieve a list of search scopes that are owned by the specified search scope consumer.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetScopesForConsumer (
   @ConsumerName         nvarchar(60)
);
```

@ConsumerName: The name of the search scope consumer. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: This procedure MUST return a single result set as specified in Section 2.2.4.6.

3.1.5.116 proc_MSS_GetScopesInfo

The **proc_MSS_GetScopesInfo** stored procedure is called to get the details of all the search scopes defined in the search application.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetScopesInfo ();
```

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: This procedure MUST return a single result set as specified in section 2.2.4.6.

3.1.5.117 proc_MSS_GetScopesManagerInfo

The **proc_MSS_GetScopesManagerInfo** stored procedure is called to get the details of the search scopes system in the search application.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetScopesManagerInfo (
   @AverageCompilationDuration          int OUTPUT,
   @CompilationScheduleType             smallint OUTPUT,
   @CustomCompilationSchedule           nvarchar(60) OUTPUT,
   @LastCompilationTime                 datetime OUTPUT,
   @NextCompilationTime                 datetime OUTPUT,
   @CompilationState                    int OUTPUT,
   @CompilationStartTime                datetime OUTPUT,
   @CompilationPercentComplete          smallint OUTPUT,
   @ScopesNeedingCompilation            int OUTPUT
);
```

@AverageCompilationDuration: Upon return from this stored procedure, this parameter MUST be set to the average compilation time for search scopes managed by the search scopes system. This value MUST NOT be NULL.

@CompilationScheduleType: Upon return from this stored procedure, this parameter MUST be set to the Compilation Schedule Type of the search scopes system. The value MUST be a Compilation Schedule Type data type as specified in section 2.2.1.3.

@CustomCompilationSchedule: MUST be ignored by the client.

@LastCompilationTime: Upon return from this stored procedure, this parameter is set to the date and time in UTC time of the last compilation of the search scopes system.

@ NextCompilationTime: Upon return from this stored procedure, this parameter MUST be set to the date and time in UTC time of the next compilation of the search scopes system.

@CompilationState: Upon return from this stored procedure, this parameter MUST be set to the search scope compilation state of the search scopes system. The value MUST be a Compilation State data type as specified in section 2.2.1.4.

@CompilationStartTime: Upon return from this stored procedure, this parameter is set to the date and time in UTC time of the current compilation of the search scopes system.

@CompilationPercentComplete: Upon return from this stored procedure, this parameter MUST be set to the percentage of compilation completed for the search scopes system. This value MUST NOT be NULL.

@ScopesNeedingCompilation: Upon return from this stored procedure, this parameter MUST be set to the number of search scopes belonging to the search scopes system that need compilation. This value MUST NOT be NULL.

Return values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: This procedure MUST NOT return any result set.

3.1.5.118 proc_MSS_GetSharepointLocationVisualisations

The **proc_MSS_GetSharepointLocationVisualisations** stored procedure is called to retrieve the preselected visualizations for the federated location.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetSharepointLocationVisualisations(
   @LocationId                    int,
   @FullVisualisationName         nvarchar(60) = 'full',
   @SummaryVisualisationName      nvarchar(60) = 'summary',
   @TopAnswerVisualisationName    nvarchar(60) = 'topanswer'
);
```

@LocationId: The identifier of the federated location.

@FullVisualisationName: This value MUST be "full".

@SummaryVisualisationName: This value MUST be "summary".

@TopAnswerVisualisationName: This value MUST be "topanswer".

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return zero, one, two or three result sets in the following order:

3.1.5.118.1 Core Results Visualisation Result Set

The Core Results Visualisation result set returns information about visualizations for the specified federated location. The result set MUST contain zero or one rows.

The T-SQL syntax for the result set is as follows:

```
LocationId              int,
VisualisationName       nvarchar(60),
Xsl                     nvarchar(max),
Properties              nvarchar(max),
SampleData              nvarchar(max);
```

LocationId: The identifier of the specified federated location.

VisualisationName: MUST be equal to "full".

Xsl: The Xsl for this visualization. This MUST be an Xsl Data Type as specified in Section 2.2.1.22.

Properties: The properties for this visualization. This MUST be a Properties Data Type as specified in Section 2.2.1.16.

SampleData: The sample data for this visualization. This MUST be a SampleData Data Type as specified in Section 2.2.1.17.

3.1.5.118.2 Summary Results Visualisation Result Set

The Summary Results Visualisation result set contains information about visualizations of the specified federated location. The result set MUST contain zero or one rows.

The T-SQL syntax for the result set is as follows:

```
LocationId          int,
VisualisationName   nvarchar(60),
Xsl                 nvarchar(max),
Properties          nvarchar(max),
SampleData          nvarchar(max);
```

LocationId: The identifier of the specified federated location.

VisualisationName: MUST be equal to "summary".

Xsl: The Xsl for this visualization. This MUST be an Xsl Data Type as specified in Section 2.2.1.22.

Properties: The properties for this visualization. This MUST be a Properties Data Type as specified in Section 2.2.1.16.

SampleData: The sample data for this visualization. This MUST be a SampleData Data Type as specified in Section 2.2.1.17.

3.1.5.118.3 Top Answer Visualisation Result Set

The Top Answer Visualisation result set contains information about visualizations of the specified federated location. The result set MUST contain zero or one rows.

The T-SQL syntax for the result set is as follows:

```
LocationId         int,
VisualisationName  nvarchar(60),
Xsl                nvarchar(max),
Properties         nvarchar(max),
SampleData         nvarchar(max);
```

LocationId: The identifier of the specified federated location.

VisualisationName: MUST be equal to "topanswer".

Xsl: The Xsl for this visualization. This MUST be an Xsl Data Type as specified in Section 2.2.1.22.

Properties: The properties for this visualization. This MUST be a Properties Data Type as specified in Section 2.2.1.16.

SampleData: The sample data for this visualization. This MUST be a SampleData Data Type as specified in Section 2.2.1.17.

3.1.5.119 proc_MSS_GetSpecialTerm

The **proc_MSS_GetSpecialTerm** stored procedure is called to retrieve the information about a keyword from the site collection given its keyword consumer group**,** and the keyword term.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetSpecialTerm (
   @ConsumerGpId          nvarchar(50),
   @Term                  nvarchar(100)
);
```

@ConsumerGpId: The unique identifier of the keyword consumer group.

@Term: The term for the keyword. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: This procedure MUST return a result set as specified in section 2.2.4.7.

3.1.5.120 proc_MSS_GetSpecialTerms

The **proc_MSS_GetSpecialTerms** stored procedure is called to retrieve the list of keywords from the site collection for a given keyword consumer group. The list can be filtered according to criteria specified in *@View* and *@Filter* parameters.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetSpecialTerms (
   @ConsumerGpId         nvarchar(50),
   @View                 int,
   @Filter               int,
   @Value                nvarchar(2048) = null
);
```

@ConsumerGpId: The unique identifier for the keyword consumer group.

@View: The type of keywords to include in the result set. The value MUST be a Keyword Type data type as specified in section 2.2.1.12.

@Filter: A Keyword Filter Type that specifies which keyword attributes are compared against the value given in parameter @Value. The value MUST be a Keyword Filter Type as specified in section 2.2.1.11.

@**Value**: The value that is compared against the Term or Contact elements in the Keyword Set as specified in the section 3.1.1.2, when performing the filtering. The value MUST be set to NULL to list all keywords for a given keyword consumer group.

Return Code Values: This procedure returns an integer which MUST be ignored.

Result Sets: This procedure MUST return a result set as specified in section 2.2.4.7.

3.1.5.121 proc_MSS_GetSpecialTermsCount

The **proc_MSS_GetSpecialTermsCount** stored procedure is called to calculate the number of keywords for a given keyword consumer group. The keywords included can be filtered according to criteria specified in *@View* and *@Filter* parameters.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetSpecialTermsCount (
   @ConsumerGpId  nvarchar(50),
   @View          int,
   @Filter        int,
   @Value         nvarchar(2048) = null,
   @Count         int output
);
```

@ConsumerGpId: A string that uniquely identifies the keyword consumer group.

@View: A 32 bit integer that specifies the type of keywords to include in the count. The value MUST be of type Keyword Type as specified in Section 2.2.1.12.

@Filter: A 32 bit integer that specifies which keyword attributes are compared against the value given in parameter *@Value*. The value MUST be of type Keyword Filter Type as specified in Section 2.2.1.11.

@**Value**: A string that is compared against when performing the filtering. The value MUST be set to NULL to list all keywords for a given keyword consumer group.

@Count: Upon return from this stored procedure, this parameter MUST be set to the number of keywords.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST NOT return any result set.

3.1.5.122 proc_MSS_GetSpecialTermsCountForBestBet

The **proc_MSS_GetSpecialTermsCountForBestBet** stored procedure is called to retrieve the number of keywords that are associated with the specified best bet.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetSpecialTermsCountForBestBet (
   @BestBetId          int,
   @Count              int output
);
```

@BestBetId: The unique identifier of the best bet. This value MUST NOT be NULL.

@ Count: Upon return from this stored procedure, this parameter MUST be set to the number of keywords associated with the specified best bet.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST NOT return any result set.

3.1.5.123 proc_MSS_GetSpecialTermsForBestBet

The **proc_MSS_GetSpecialTermsForBestBet** stored procedure is called to retrieve the list of keywords associated with a specified best bet.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetSpecialTermsForBestBet (
   @BestBetId           int
);
```

@BestBetId: The unique identifier of the best bet. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return a single result set as specified in Section 2.2.4.7.

3.1.5.124 proc_MSS_GetSpellingSuggestionAlwaysSuggestList

The **proc_MSS_GetSpellingSuggestionAlwaysSuggestList** stored procedure is called to retrieve information about language resources with Type 8 from the Language Resources Set**.** See section 3.1.1.8 for the definition of Language Resources Set.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetSpellingSuggestionAlwaysSuggestList();
```

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return the following result set:

3.1.5.124.1 GetSpellingSuggestionAlwaysSuggestList Result Set

The **GetSpellingSuggestionAlwaysSuggestList** result set returns information about language resources. The result set MUST contain zero or more rows, each corresponding to a single entry in the Language Resources Set as specified in section 3.1.1.8.

The T-SQL syntax for the result set is as follows:

```
Phrase      nvarchar(128),
Mapping     nvarchar(128),
```

Phrase: The **Phrase** element of the language resource.

Mapping: The **Mapping** element of the language resource.

The **GetSpellingSuggestionAlwaysSuggestList** result set MUST contain information about those language resources whose **Type** element is equal to "8". The result set MUST NOT contain any information about any other language resources.

3.1.5.125 proc_MSS_GetSpellingSuggestionBlockList

The **proc_MSS_GetSpellingSuggestionBlockList** stored procedure is called to retrieve information about language resources with **Type** "3" from the Language Resources Set**.** See section 3.1.1.8 for the definition of Language Resources Set.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetSpellingSuggestionBlockList();
```

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return the result set specified in the following subsections.

3.1.5.125.1 GetSpellingSuggestionBlockList Result Set

The **GetSpellingSuggestionBlockList** result set returns information about language resources. The result set MUST contain zero or more rows, each corresponding to a single entry in the Language Resources Set as specified in section 3.1.1.8.

The T-SQL syntax for the result set is as follows:

```
Phrase      nvarchar(128),
```

Phrase: The **Phrase** element of the language resource.

The **GetSpellingSuggestionBlockList** result set MUST contain information about those language resources whose **Type** element is equal to "3". The result set MUST NOT contain any information about any other language resources.

3.1.5.126 proc_MSS_GetSqmInfo

The **proc_MSS_GetSqmInfo** stored procedure is called to retrieve a summary of the usage of the search service application for the most recent specified interval. **proc_MSS_GetSqmInfo** MAY be called at any time, or not at all.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetSqmInfo(
   @Interval           int
);
```

@Interval: An integer representing the number of days of usage data to summarize. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return a single row of the following result set:

3.1.5.126.1 SQM Info Result Set

The T-SQL syntax for the stored procedure is as follows:

SearchCenterQueries int

SiteSearchQueries int

AdvancedSearchQueries int

DidYouMeanSearchQueries int

WebSearchQueries int

UISearchQueries int

AllSitesScopeQueries int

PeopleScopeQueries int

NonOutOfBoxScopeQueries int

BestBetClicks int

BestBetInExistance int

NumUsers int

NumQueryTokens int

NumQueryTypes int

QueriesPerDayAvg float

QueriesPerDaySdv float

QueryLengthAvg float

QueryLengthSdv float

ClicksPerQueryTokenAvg float

ClicksPerQueryTokenSdv float

SecondsPerQueryTokenAvg float

SecondsPerQueryTokenSdv float

QueriesPerUserAvg float

QueriesPerUserSdv float

SearchApplicationKItemsIndexedint

SearchCenterQueries: Any integer. This value MUST NOT be NULL.

SiteSearchQueries: Any integer. This value MUST NOT be NULL.

AdvancedSearchQueries: Any integer. This value MUST NOT be NULL.

DidYouMeanSearchQueries: Any integer. This value MUST NOT be NULL.

WebSearchQueries: Any integer. This value MUST NOT be NULL.

UISearchQueries: Any integer. This value MUST NOT be NULL.

AllSitesScopeQueries: Any integer. This value MUST NOT be NULL.

PeopleScopeQueries: Any integer. This value MUST NOT be NULL.

NonOutOfBoxScopeQueries: Any integer. This value MUST NOT be NULL.

BestBetClicks: Any integer. This value MUST NOT be NULL.

BestBetInExistance: Any integer. This value MUST NOT be NULL.

NumUsers: Any integer. This value MUST NOT be NULL.

NumQueryTokens: Any integer. This value MUST NOT be NULL.

NumQueryTypes: Any integer. This value MUST NOT be NULL.

QueriesPerDayAvg: Any float. This value MUST NOT be NULL.

QueriesPerDaySdv: Any float. This value MUST NOT be NULL.

QueryLengthAvg: Any float. This value MUST NOT be NULL.

QueryLengthSdv: Any float. This value MUST NOT be NULL.

ClicksPerQueryTokenAvg: Any float. This value MUST NOT be NULL.

ClicksPerQueryTokenSdv: Any float. This value MUST NOT be NULL.

SecondsPerQueryTokenAvg: Any float. This value MUST NOT be NULL.

SecondsPerQueryTokenSdv: Any float. This value MUST NOT be NULL.

QueriesPerUserAvg: Any float. This value MUST NOT be NULL.

QueriesPerUserSdv: Any float. This value MUST NOT be NULL.

SearchApplicationKItemsIndexed: Any integer. This value MUST NOT be NULL.

3.1.5.127 proc_MSS_GetStaticRankingFeatures

The **proc_MSS_GetStaticRankingFeatures** stored procedure is called to retrieve information about all static ranking features from the Static Ranking Features Set, specified in section 3.1.1.5.2.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetStaticRankingFeatures();
```

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return the following result set.

3.1.5.127.1 Static Ranking Features Result Set

The Static Ranking Features result set contains information about static ranking features in the Static Ranking Features Set, specified in section 3.1.1.5.2. The result set will contain zero or more rows, each corresponding to a single static ranking feature.

The T-SQL syntax for the result set is as follows:

```
Pid             int NOT NULL,
DefaultVal      int NOT NULL,
IsOccOneGreater bit NOT NULL,
IsInAnchorIndex bit NOT NULL;
```

Pid: The property identifier of the static ranking feature.

DefaultVal: The default value of the static ranking feature.

IsOccOneGreater: The value modifier flag of the static ranking feature. This value MUST be 1 if a modifier is to be applied, otherwise it MUST be 0.

IsInAnchorIndex: The flag of the static ranking feature indicating whether the feature applies to the anchor text catalog, as specified in specified in [MS-CIFO] section 2.18.2. The value of this flag MUST be 1 if the static ranking feature applies to the anchor text catalog, otherwise it MUST be 0.

3.1.5.128 proc_MSS_GetSummaryByHost

The **proc_MSS_GetSummaryByHost** stored procedure is called to retrieve crawl statistics for the hosts processed by the crawler from the Crawled Hosts Set ([MS-SQLPGAT2] section 3.1.1.7).

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetSummaryByHost (
   @Order              int,
   @Count              int,
   @Dir                bit
);
```

@Order: The ordering of the rows in the result set. An integer which MUST be one of the values listed in the following table.

Value	Description
0	The result set is ordered by host name.
1	The result set is ordered by success count.
2	The result set is ordered by warning count.

Value	Description
3	The result set is ordered by error count.
4	The result set is ordered by total count.

@Count: The number of rows to include in the result set. This value MUST NOT be NULL. *@Count* + 1 is the maximum number of rows to include in the result set.

@Dir: The direction of ordering the result set. An integer which MUST be one of the values listed in the following table.

Value	Description
0	The result set is sorted in ascending order.
1	The result set is sorted in descending order.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return the following two result sets:

3.1.5.128.1 Start At Result Set

The StartAt result set contains information about the row number for the next host if there are more hosts available than restricted by the value of *@Count* +1. Otherwise, the result set returns -1. The result set MUST contain one row.

The T-SQL syntax for the result set is as follows:

```
{StartAt}      int;
```

{StartAt}: the row number for the next host if there are more hosts available than restricted by the value of *@Count* +1. This MUST be set to -1 when crawl statistics for all hosts have been returned in the result set.

3.1.5.128.2 Host Summary Result Set

The Host Summary result set contains information about the crawl statistics for the hosts processed by the crawler from the Crawled Hosts Set ([MS-SQLPGAT2] section 3.1.1.7). The ordering of the rows depends on the combination of the input parameters *@Order* and *@Dir*. The result set MUST contain as many rows as there are hosts in Crawled Hosts Set or the value of the *@Count* + 1 as described previously, whichever is smaller.

The T-SQL syntax for the result set is as follows:

```
HostName           nvarchar(300),
SuccessCount       int,
WarningCount       int,
ErrorCount         int,
TotalCount         int;
```

HostName: The host name.

SuccessCount: The number of items crawled successfully.

WarningCount: The number of items crawled with warnings.

ErrorCount: The number of items crawled with errors.

TotalCount: The sum of *SuccessCount*, *WarningCount* and *ErrorCount.*

3.1.5.129 proc_MSS_GetSummaryLogData

The **proc_MSS_GetSummaryLogData** stored procedure is called to retrieve the aggregated crawl statistics for the specified content source from the Crawl Url History Set ([MS-SQLPGAT2] section 3.1.1.2).

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetSummaryLogData (
   @CatalogID         int = null,
   @ContentSourceID   int = null,
   @Successes         int OUTPUT,
   @Errors            int OUTPUT,
   @Warnings          int OUTPUT
);
```

@CatalogID: MUST be set to 1.

@ContentSourceID: The identifier of the content source for which to retrieve the crawl statistics**.**

@Successes: Upon return from this stored procedure, this parameter MUST be set to the total number of items crawled successfully within the specified content source. The value MUST be set to 0 if the @ContentSourceID value is set to NULL.

@Errors: Upon return from this stored procedure, this parameter MUST be set to the total number of items within the specified content source that were crawled with errors. The value MUST be set to 0 if the @ContentSourceID value is set to NULL.

@Warnings: Upon return from this stored procedure, this parameter MUST be set to the total number of items within the specified content source that were crawled with warnings. The value MUST be set to 0 if the @ContentSourceID value is set to NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST NOT return any result set.

3.1.5.130 proc_MSS_GetSynonym

The **proc_MSS_GetSynonym** stored procedure is called to retrieve the information about the specified synonym associated with a given keyword from the Synonym Set as specified in section 3.1.1.2.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetSynonym(
   @SpecialTermId             int,
   @Term                      nvarchar(100)
```

```
);
```

@SpecialTermId: The unique identifier of the keyword. This value MUST NOT be NULL.

@Term: The keyword synonym. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return a single result set as specified in Section 2.2.4.8.

3.1.5.131 proc_MSS_GetSynonyms

The **proc_MSS_GetSynonyms** stored procedure is called to retrieve the information about synonyms associated with a given keyword from the Synonym Set as specified in section 3.1.1.2.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetSynonyms(
   @SpecialTermId      int
);
```

@SpecialTermId: The unique identifier of the keyword. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return a single result set as specified in Section 2.2.4.8.

3.1.5.132 proc_MSS_GetSynonymsCount

The **proc_MSS_GetSynonymsCount** stored procedure is called to calculate the number of synonyms associated with a given keyword.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetSynonymsCount(
   @SpecialTermId             int,
   @Count                     int output
);
```

@SpecialTermId: The unique identifier of the keyword. This value MUST NOT be NULL.

@Count: Upon return from this stored procedure, this parameter MUST be set to the number of keyword synonyms.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST NOT return any result set.

3.1.5.133 proc_MSS_GetTotalSuccess

The **proc_MSS_GetTotalSuccess** stored procedure is called to retrieve the count of items crawled successfully from the Crawl Item Set as specified in section 3.1.1.3.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetTotalSuccess ();
```

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: This procedure MUST return the following result set:

3.1.5.133.1 TotalSuccess Result Set

The TotalSuccess result set returns the count of items crawled successfully from the Crawl Item Set as specified in section 3.1.1.3.. The result set MUST contain exactly one row.

The T-SQL syntax for the result set is as follows:

```
total         int;
```

Total: The count of items crawled successfully.

3.1.5.134 proc_MSS_GetUniqueAcronyms

The **proc_MSS_GetUniqueAcronyms** stored procedure is called to retrieve distinct acronyms from the Popular Acronym Set, specified in section 3.1.1.9.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetUniqueAcronyms();
```

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return the following result set:

3.1.5.134.1 GetUniqueAcronyms Result Set

The **GetUniqueAcronyms** result set returns the names of distinct acronyms. The result set MUST contain zero or more rows.

The T-SQL syntax for the result set is as follows:

```
Acronym      nvarchar(50);
```

Acronym: The acronym string.

3.1.5.135 proc_MSS_GetUnusedScopesForConsumer

The **proc_MSS_GetUnusedScopesForConsumer** stored procedure is called to retrieve the information about all search scopes owned by the specified search scope consumer that are not associated with any search scope display group.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetUnusedScopesForConsumer (
   @ConsumerName         nvarchar(60)
```

```
);
```

@**ConsumerName**: The name of the search scope consumer. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: This procedure MUST return a single result set as specified in section 2.2.4.6.

3.1.5.136 proc_MSS_GetUrlCrawlLogData

The **proc_MSS_GetUrlCrawlLog** stored procedure is called to retrieve entries from the Crawl Item Set (section 3.1.1.3).

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetUrlCrawlLogData(
   @Url NVARCHAR(450)= null,
   @IsLike bit = null,
   @ContentSourceId int = null,
   @ErrorLevel int = null,
   @ErrorId int = null,
   @HostId int = null,
   @LogLevel int = null,
   @MaximumTimeStamp datetime = null,
   @MinimumTimeStamp datetime = null,
   @MaximumDocId int,
   @MinimumDocId int,
   @MaximumRows int
);
```

@Url: The display URL of the item to include in the result set, or NULL to indicate that the entries in the result set are not limited by any display URL.

@IsLike: It MUST be set to 1 to retrieve the items whose prefix of the *DisplayUrl* attribute matches the *@Url* parameter value. Otherwise, MUST be set to 0 or NULL.

@ContentSourceId: The integer identifier of the content source, or NULL to indicate that the entries in the result set are not limited by any content source.

@ErrorLevel: The error type which MUST be a Crawl Log Error Level data type as specified in section 2.2.1.7, or NULL to indicate that the entries in the result set are not limited by any error type.

@ErrorId: The unique identifier of the error, or NULL to indicate that the entries in the result set are not limited by ErrorId.

@HostId: The unique identifier of the host, or NULL to indicate that the entries in the result set are not limited by any host.

@LogLevel: The level of the item in the site hierarchy which MUST be a Crawl Log Level data type as specified in section 2.2.1.8, or NULL to indicate that the entries in the result set are not limited by LogLevel.

@MaximumTimeStamp: A date and time in UTC time that MUST be greater than the value of *TimeStamp* attribute for the entries included in the result set. This value MUST be set to NULL to indicate that the entries in the result set are not filtered by *TimeStamp*.

@MinimumTimeStamp: A date and time in UTC time that MUST be less than the value of *TimeStamp* attribute for the entries included in the result set. This value MUST be set to NULL to indicate that the entries in the result set are not filtered by *TimeStamp*.

@MaximumDocId: The integer that MUST be greater that the value of *DocId* attribute for the entries included in the result set.

@MinimumDocId: The integer that MUST be less the value of *DocID* attribute for the entries included in the result set.

@MaximumRows: The maximum number of rows to include in the result set.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: This procedure MUST return the following result set:

3.1.5.136.1 Url Crawl Log Result Set

The Url Crawl Log result set returns information about items from the Crawl Item Set (section 3.1.1.3). The result set MUST contain zero or more rows, each row corresponding to a single item.

The T-SQL syntax for the result set is as follows:

```
FullUrl             nvarchar(1500),
StatusTypeId        int,
StatusMessageId     int,
ContentSourceId     int,
DocID               int,
TimeStamp           datetime,
DeleteReason        int,
ErrorDesc           nvarchar(512);
```

FullUrl: The display URL of the item. This value MUST NOT be NULL.

StatusTypeId: The type of the error. The value MUST be a Crawl Log Error Level data type as specified in section 2.2.1.7.

StatusMessageId: The unique identifier of the error. The value MUST be one of the values listed in the Crawl Error Set (section 3.1.1.3).

ContentSourceId: The unique identifier of the content source. This value MUST NOT be NULL.

DocID: The unique identifier of the item. This value MUST NOT be NULL.

TimeStamp: The UTC date and time when the item was last crawled. This value MUST NOT be NULL.

DeleteReason: The unique identifier of the reason for removing the item from the search catalog. The value MUST be one of the values listed in the Crawl Deleted Reason Set (section 3.1.1.3)**.**

ErrorDesc: The description of the error occurred when crawling the item. This value can be NULL.

3.1.5.137 proc_MSS_GetUrlCrawlLogSummary

The **proc_MSS_GetUrlCrawlLogSummary** stored procedure is called to retrieve the number of entries from the Crawl Item Set (section 3.1.1.3) matching stored procedure input parameters.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetUrlCrawlLogSummary(
   @Url NVARCHAR(450) = null,
   @IsLike bit = null,
   @ContentSourceId in = null,
   @ErrorLevel int = null,
   @ErrorId int = null,
   @HostId int = null,
   @LogLevel int = null,
   @MaximumTimeStamp datetime = null,
   @MinimumTimeStamp datetime = null
);
```

@Url: The display URL of the item to include in the result set, or NULL to indicate that the entries in the result set are not limited by any display URL.

@IsLike: It MUST be set to 1 to retrieve the items whose prefix of the *DisplayUrl* attribute matches the *@Url* parameter value. Otherwise, MUST be set to 0 or NULL.

@ContentSourceId: The unique identifier of the content source, or NULL to indicate that the entries in the result set are not limited by any content source.

@ErrorLevel: The error type which MUST be a Crawl Log Error Level data type as specified in section 2.2.1.7, or NULL to indicate that the entries in the result set are not limited by any error type.

@ErrorId: The unique identifier of the error, or NULL to indicate that the entries in the result set are not limited by any error.

@HostId: The unique identifier of the host, or NULL to indicate that the entries in the result set are not limited by any host.

@LogLevel: The level of the item in the site hierarchy which MUST be a Crawl Log Level data type as specified in section 2.2.1.8, or NULL to indicate that the entries in the result set are not limited by LogLevel.

@MaximumTimeStamp: A date and time in UTC time that MUST be greater than the value of *TimeStamp* attribute for the entries included in the result set. This value MUST be set to NULL to indicate that the entries in the result set are not filtered by *TimeStamp*.

@MinimumTimeStamp: A date and time in UTC time that MUST be less than the value of *TimeStamp* attribute for the entries included in the result set. This value MUST be set to NULL to indicate that the entries in the result set are not filtered by *TimeStamp*.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: This procedure MUST return the following result set:

3.1.5.137.1 Url Crawl Log Summary Result Set

The Url Crawl Log Summary Groups result set returns the number of entries from the Crawl Item Set (section 3.1.1.3) corresponding given input parameters. The result set MUST contain exactly one row.

The T-SQL syntax for the result set is as follows:

Total int**Total:** The number of items corresponding to given input parameters. This value MUST NOT be NULL.

3.1.5.138 proc_MSS_GetUsedMessages

The **proc_MSS_GetUsedMessages** stored procedure is called to retrieve entries from the Crawl Error Set, as specified in section 3.1.1.3, corresponding to the entries in the Crawl Url History set ([MS-SQLPGAT2] section 3.1.1.2).

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetUsedMessages();
```

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return a single result set as specified in section 2.2.4.4.

3.1.5.139 proc_MSS_GetUserPreference

The **proc_MSS_GetUserPreference** stored procedure is called to retrieve search preference information, as specified in section 3.1.1.10, for a given user.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetUserPreference (
   @UserID         varbinary(512)
);
```

@UserID: Unique identifier, as defined by tp_SystemID of [MS-WSSFO2] section 2.2.7.10, of the user for whom to retrieve the search preference information. This value MUST NOT be NULL and MUST NOT be empty.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return the following result set:

3.1.5.139.1 User Preference Result Set

The user preference result set returns the search preference, specified in section 3.1.1.10, for the specified user. The result set MUST contain one row if the specified user has configured search preference, otherwise the result set MUST contain zero row. If the client stores a copy of the user preference instance locally, it MUST update the *v* attribute, as specified in section 2.2.6.3.22, of the copy with the value of the *Version* element in the result set.

The T-SQL syntax for the result set is as follows:

```
Preference              nvarchar(max),
Version                 int;
```

Preference: Contains an **XML fragment** representing the search user preference of the user specified by *@UserID*. The XML Schema for this structure is defined in section 2.2.6.4.2. This value MUST NOT be NULL and MUST NOT be empty.

Version: The server configuration version of the search user preference instance. This value MUST NOT be NULL and MUST be greater or equal to 0.

3.1.5.140 proc_MSS_GetVisibleScopesCount

The **proc_MSS_GetVisibleScopesCount** stored procedure is called to get the count of all visible scopes defined in the search application that have DisplayInAdminUI = 1. For the specification of DisplayInAdminUI, see section 2.2.1.9.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetVisibleScopesCount (
   @Count          int OUTPUT
);
```

@Count: Upon return from this stored procedure, this parameter MUST be set to the count of visible scopes defined in the search application. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: This procedure MUST NOT return any result set.

3.1.5.141 proc_MSS_GetVolatileScopeInfo

The **proc_MSS_GetVolatileScopeInfo** stored procedure is called to get the details likely to change for the specified search scope.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetVolatileScopeInfo (
   @ScopeID                      int,
   @CompilationState             smallint OUTPUT,
   @LastCompilationTime          datetime OUTPUT
);
```

@ScopeID: The unique identifier of the search scope. This value MUST NOT be NULL.

@CompilationState: Upon return from this stored procedure, this parameter MUST be set to the search scope compilation state of the search scope. The value MUST be a Compilation State data type as specified in section 2.2.1.4.

@LastCompilationTime: Upon return from this stored procedure, this parameter is set to the date and time in UTC time of the last compilation of the search scope.

Return Code Values: An integer which MUST be listed in the following table:

Value	Description
0	Successful execution.
1	Search scope with the specified identifier was not found.

Result Sets: This procedure MUST NOT return any result set.

3.1.5.142 proc_MSS_GetVolatileScopesManagerInfo

The **proc_MSS_GetVolatileScopesManagerInfo** stored procedure is called to get the details likely to change for a search scopes system in the search application.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_GetVolatileScopesManagerInfo (
   @AverageCompilationDuration   int OUTPUT,
   @LastCompilationTime          datetime OUTPUT,
   @NextCompilationTime          datetime OUTPUT,
   @CompilationState             smallint OUTPUT,
   @CompilationStartTime         datetime OUTPUT,
   @CompilationPercentComplete   smallint  OUTPUT,
   @ScopesNeedingCompilation     int OUTPUT
);
```

@AverageCompilationDuration: Upon return from this stored procedure, this parameter MUST be set to the average compilation time for search scopes in the search scopes system. This value MUST NOT be NULL.

@LastCompilationTime: Upon return from this stored procedure, this parameter is set to the date and time of the last compilation of the search scopes system.

@NextCompilationTime: Upon return from this stored procedure, this parameter MUST be set to the date and time of the next compilation of the search scopes system.

@CompilationState: Upon return from this stored procedure, this parameter MUST be set to the search scope compilation state of the search scopes system. The value MUST be a Compilation State data type as specified in section 2.2.1.4.

@CompilationStartTime: Upon return from this stored procedure, this parameter is set to the date and time of the current compilation of the search scopes system.

@CompilationPercentComplete: Upon return from this stored procedure, this parameter MUST be set to the percentage of compilation completed for the search scopes system. This value MUST NOT be NULL.

@ScopesNeedingCompilation: Upon return from this stored procedure, this parameter MUST be set to the number of search scopes belonging to the search scopes system that need compilation. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: This procedure MUST NOT return any result set.

3.1.5.143 proc_MSS_PurgePastCrawlLog

The **proc_MSS_PurgePastCrawlLog** stored procedure is called to delete all crawl information from the MssCrawlUrlLog table ([MS-SRCHTP] section 2.2.5.8).

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_PurgePastCrawlLog ();
```

Return values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: This procedure SHOULD NOT<43> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.144 proc_MSS_QLog_GetClickFrequenciesForUrl

The **proc_MSS_QLog_GetClickFrequenciesForUrl** stored procedure is called to get the number of times a given URL has been clicked in the past week and the past month from the specified date.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_QLog_GetClickFrequenciesForUrl {
   @url                nvarchar(1024),
   @referenceDate      datetime
};
```

@url: The URL whose click frequencies are computed.

@referenceDate: The date and time against which the click frequencies are computed. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored

Result Sets: MUST return the following result set:

3.1.5.144.1 GetClickFrequenciesForUrl Result Set

The **GetClickFrequenciesForUrl** result set returns the number of times a given URL has been clicked in the past week and the past month from the specified date. The result set MUST contain two rows, the sum of the number of times the specified URL has been clicked in the past week, and the past month, from the date and time specified by @referenceDate parameter.

The T-SQL syntax for the result set is as follows:

```
numClicks         int;
```

numClicks: The click frequency for the specified URL.

3.1.5.145 proc_MSS_QLog_GetTopQueryStringsForUrl

The **proc_MSS_QLog_GetTopQueryStringsForUrl** stored procedure is called to get the queries that result in the most number of clicks for a given URL.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_QLog_GetClickFrequenciesForUrl {
   @url                nvarchar(1024),
   @topCount           int
};
```

@url: The input URL

@topCount: The number of queries to return. This value MUST NOT be NULL

Return Code Values: This stored procedure returns an integer value that MUST be ignored

Result Sets: MUST return the following result set:

3.1.5.145.1 GetTopQueryStringsForUrl Result Set

The **GetTopQueryStringsForUrl** result set returns the top queries, and the number of clicks on the input URL that are attributed to them. The result set MUST contain zero or more rows. The result set MUST be ordered by the number of clicks, in descending order.

The T-SQL syntax for the result set is as follows:

```
queryString            nvarchar(1024),
frequency              int;
```

queryString: The query text.

frequency: The number of clicks on the specified URL

3.1.5.146 proc_MSS_RemoveFilenameFromResults

The **proc_MSS_RemoveFilenameFromResults** stored procedure is called to remove an item from the metadata index so it will not appear in the search result.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_RemoveFilenameFromResults (
   @DocID          int
);
```

@DocID: The unique identifier of the item. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<44> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.147 proc_MSS_SetConfigurationProperty

The proc_MSS_SetConfigurationProperty stored procedure is specified in [MS-SQLPGAT2] section 3.1.5.71.

3.1.5.148 proc_MSS_SetCrawledCategoryPropertiesAllOM

The **proc_MSS_SetCrawledCategoryPropertiesAllOM** stored procedure is called to updated data for the specified crawled property category in the metadata schema with the values provided by the stored procedure parameters. This procedure also updates value of the CrawledPropertyCategoryTimestamp in the Metadata Timestamp Set as specified in Section 3.1.1.1 with the current date and time in local time of the server.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_SetCrawledCategoryPropertiesAllOM (
   @Name                          nvarchar(64),
```

```
    @NewName                        nvarchar(64),
    @DiscoverNewProperties          bit,
    @MapToContents                  bit,
    @FullTextQueriable              bit,
    @Retrievable                    bit,
    @Scoped                         bit,
    @MatchExistingManagedProperty   bit,
    @MatchIgnorePrefix              nvarchar(64),
    @MatchIgnoreSuffix              nvarchar(64),
    @MaxIndexedStringLength         int,
    @MaxNonIndexedStringLength      int
);
```

@Name: The name of the crawled property category to be updated.

@NewName: If not equal to *@Name* and if the *@NewName* does not already exist in the Crawled Property Category Set, specified in Section 3.1.1.1, then the crawled property name is updated with *@NewName*.

@DiscoverNewProperties: MUST be 1 if the crawled properties within this crawled property category are added automatically to the Crawled Properties Set, specified in Section 3.1.1.1. Otherwise, it MUST be 0.

@MapToContents: MUST be 1 if string data from newly discovered crawled properties within this crawled property is put in the full-text index catalog. Otherwise, it MUST be 0.

@FullTextQueriable: This value MUST be 1 if the data for the crawled property category is kept in the full-text index catalog. Otherwise, it MUST be 0.

@Retrievable: This value MUST be 1 if the data for the crawled property category is kept in the metadata index. Otherwise, it MUST be 0.

@Scoped: This value MUST be 1 if the data for the crawled property category is kept in the search scope index. Otherwise, it MUST be 0.

@MatchExistingManagedProperty: This value MUST be set to 1, if the crawled properties in this crawled property category have to be mapped to an existing managed property before auto generating the corresponding managed properties. Otherwise, it MUST be 0.

@MatchIgnorePrefix: This value MUST be set to 1, if the managed property prefix has to be ignored when matching the managed property to the crawled properties in this crawled property category. Otherwise, it MUST be 0.

@MatchIgnoreSuffix: This value MUST be set to 1 if the managed property suffix has to be ignored when matching the managed property to the crawled properties in this crawled property category. Otherwise, it MUST be 0.

@MaxIndexedStringLength: If the string value length in the *MSSDocProps* table, defined in [MS-SQLPQ2] section 2.2.5.2, is greater than the size allowed by the MaxIndexedStringLength, the string is truncated to MaxIndexedStringLength and the string overflow is stored in the strVal2 column, in the *MSSDocProps* table defined in [MS-SQLPQ2] section 2.2.5.2, up to the size allowed by the MaxNonIndexedStringLength. This value MUST NOT be NULL.

@MaxNonIndexedStringLength: If the string value length in the *MSSDocProps* table, defined in [MS-SQLPQ2] section 2.2.5.2, is greater than the size allowed by the MaxIndexedStringLength, the string is truncated to MaxIndexedStringLength and the string overflow is stored in the strVal2

column, in the *MSSDocProps* table defined in [MS-SQLPQ2] section 2.2.5.2, up to the size allowed by the MaxNonIndexedStringLength. This value MUST NOT be NULL.

Return Code Values: An integer which MUST be one of the values listed in the following table.

Value	Description
0	Successful execution.
1	New name for the crawled property already exists.

Result Sets: SHOULD NOT<45> return any result sets. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.149 proc_MSS_SetCrawledPropertyMapToContent

The **proc_MSS_SetCrawledPropertyMapToContent** stored procedure is called to update the value of IsMappedToContent attribute in the Crawled Property set for the specified crawled property. Upon successful execution the stored procedure updates the value of CrawledPropertyTimestamp in the Metadata Timestamp Set, and the value of LastModifiedTime attribute in the Crawled Property Set for the corresponding crawled property with the local time of the server. See section 3.1.1.1 for the definition of Crawled Property set and Metadata Timestamp set.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_SetCrawledPropertyMapToContent (
   @crawledPropId             int,
   @IsMappedToContent         bit
);
```

@crawledPropId: Unique identifier of the crawled property. This value MUST NOT be NULL.

@IsMappedToContent: When the value is set to 1 and the variant type (2) of the specified crawled property is a string, the data from this crawled property will be written to the full-text index catalog. Otherwise, it MUST be 0.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<46> return any result sets. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.150 proc_MSS_SetDefaultRankingModelOM

The **proc_MSS_SetDefaultRankingModel** stored procedure marks a specific ranking model as the default ranking model within the Ranking Model Set, specified in section 3.1.1.5.1, and updates the Global Ranking Parameters Timestamp variable with the current date and time.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_SetDefaultRankingModelOM (
   @ModelId      uniqueidentifier,
);
```

@ModelId: The identifier of the ranking model which is to be marked as the default. This value MUST NOT be NULL.

Return Code Values:

Value	Description
0	Successful execution.
1	The specified ranking model identifier was not found within the Ranking Model Set, specified in section 3.1.1.5.1, and no changes were made by the stored procedure.

Result Sets: MUST NOT return any result set.

3.1.5.151 proc_MSS_SetManagedPropertyAllOM

The **proc_MSS_SetManagedPropertyAllOM** stored procedure is called to update all attributes for a managed property in the metadata schema with the values provided by the stored procedure parameters . This procedure also updates the LastModifyTime attribute in Managed Property Set, specified in Section 3.1.1.1, and the value of the ManagedPropertyAddModifyTimestamp in the Metadata Timestamp Set with the current date and time in local time of the server.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_SetManagedPropertyAllOM (
   @PID                                int,
   @Name                               nvarchar(64),
   @Description                        nvarchar(2048),
   @ManagedType                        int,
   @FullTextQueriable                  bit,
   @Retrievable                        bit,
   @EnabledForScoping                  bit,
   @RespectPriority                    bit,
   @RemoveDuplicates                   bit,
   @HasMultipleValues                  bit,
   @OverrideHasMultipleValues          bit,
   @NoWordBreaker                      bit,
   @IsNameNormalized                   bit,
   @IncludeInMd5                       bit,
   @Mapped                             bit,
   @QueryIndependentRank               bit,
   @UserFlags                          smallInt,
   @WordBreakerOverrride               int,
   @Weight                             float,
   @LengthNormalization                float,
   @MaxIndexedStringLength             int,
   @MaxNonIndexedStringLength          int,
   @MaxRetrievalLength                 int,
   @DecimalPlaces                      tinyint,
   @IsInDocProps                       bit,
   @IsInPropertyBlobOptimizedResults   bit,
   @QueryPropertyBlob                  bit,
   @SuppressStringNormalizer           bit,
   @Pronunciation                      bit,
   @DefaultForQIR                      int,
   @SplitStringCharacters              nvarchar(64)
);
```

@PID: The unique identifier of the managed property. This value MUST NOT be NULL.

@Name: The new name for the managed property. This value MUST NOT be NULL.

@Description: Description of the managed property.

@ManagedType: The type of the managed property as specified in Section 2.2.1.15**.**

@FullTextQueriable: MUST be 1 if the data for the managed property is kept in the full-text index catalog, otherwise, it MUST be 0.

@Retrievable: MUST be 1 if the data for the managed property is kept in the metadata index. Otherwise, it MUST be 0.

@EnabledForScoping: MUST be 1 if the data for the managed property is kept in the search scope index. Otherwise, it MUST be 0.

@RespectPriority: MUST be 1 if only data from the crawled property mapped to this managed property with highest priority mapping order is used. It MUST be 0 if values from all crawled properties mapped to this managed property are used.

@RemoveDuplicates: This value MUST be 1 if the duplicate entries are removed. Otherwise, MUST be 0.

@HasMultipleValues: A bit which MUST be 1 if the value of the managed property can contain multiple values. Otherwise, it MUST be 0.

@OverrideHasMultipleValues: A flag indicating whether the @HasMultipleValues parameter is the sole mechanism for the protocol client to use for determining whether the managed property is multi-valued. If the value of the @OverrideHasMultipleValues parameter is 1 then the protocol client MUST NOT ignore the value of the @HasMultipleValues parameter.

@NoWordBreaker: MUST be ignored by the server.

@IsNameNormalized: MUST be 1 if the value of this managed property is normalized by the crawler. Otherwise, it MUST be 0.

@IncludeInMD5: MUST be 1 if values mapped to this managed property are used to determine if the item has changed. Otherwise, it MUST be 0.

@Mapped: MUST be 1 when the property is an URL that is subject to **alternate access mappings**. Otherwise, it MUST be 0..

@QueryIndependentRank: This value MUST be 1 when the property participates in **query independent rank**. Otherwise, it MUST be 0.

@UserFlags: A 16 bit field that can be retrieved and set by an administrator that is open to custom applications that use the public schema object model to get and set these. This value MUST NOT be NULL.

@WordBreakerOverrride: MUST be ignored by the server. This value MUST NOT be NULL.

@Weight: A decimal value used to adjust property oriented rank. This value MUST NOT be NULL.

@LengthNormalization: A decimal value used to adjust property oriented rank. This value MUST NOT be NULL.

@MaxIndexedStringLength: The maximum number of characters persisted in the string value column in the *MSSDocProps* table defined in [MS-SQLPQ2] section 2.2.5.2. If the string value length is greater than MaxIndexedStringLength, the string is truncated to MaxIndexedStringLength. This value MUST NOT be NULL.

@MaxNonIndexedStringLength: If the string value length in the *MSSDocProps* table, defined in [MS-SQLPQ2] section 2.2.5.2, is greater than the size allowed by the MaxIndexedStringLength, the string is truncated to MaxIndexedStringLength and the string overflow is stored in the strVal2 column, in the *MSSDocProps* table defined in [MS-SQLPQ2] section 2.2.5.2, up to the size allowed by the MaxNonIndexedStringLength. This value MUST NOT be NULL.

@MaxRetrievalLength: The maximum number of characters persisted for a fixed-length string property in the *MSSDocResults* table defined in [MS-SQLPQ2] section 2.2.5.3. This MUST NOT be NULL

@DecimalPlaces: The number of floating point decimal places that must be honored in the metadata index. This value MUST NOT be NULL.

@IsInDocProps: This value MUST be 1 if the managed property is stored in the *MSSDocProps* table defined in [MS-SQLPQ2] section 2.2.5.2. Otherwise, MUST be 0.

@IsInPropertyBlobOptimizedResults: This value MUST be 1 if the managed property is persisted in the *PropertyBlob* column of the *MSSDocResults* table defined in [MS-SQLPQ2] section 2.2.5.3. Otherwise, MUST be 0.

@QueryPropertyBlob: MUST contain the same value as the IsInPropertyBlobOptimizedResults parameter.

@SuppressStringNormalizer: This value MUST be 1 if the string normalization for the managed property is to be skipped. Otherwise, MUST be 0.

@Pronunciation: This value MUST be 1 if the managed property needs a pronunciation string. Otherwise, MUST be 0.

@DefaultForQIR: The value of this parameter MUST contain the static ranking default value for this managed property as specified in Section 3.1.1.5.2. This value MUST NOT be NULL.

@SplitStringCharacters: This value MUST contain a list of characters used to split the string data for the managed property into separate strings which do not contain the **SplitStringCharacters**. Otherwise, it MUST be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<47> return any result sets. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.152 proc_MSS_SetManagedPropertyHasMultipleValues

The **proc_MSS_SetManagedPropertyHasMultipleValues** stored procedure is called to specify that the given managed property can contain multiple values for a single item. This procedure also updates the LastModifyTime attribute in Managed Property Set and the value of the ManagedPropertyAddModifyTimestamp in the Metadata Timestamp Set with the current date and time in local time of the server.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_SetManagedPropertyHasMultipleValues (
```

```
    @PID                          int,
    @HasMultipleValues            bit,
    @OverrideHasMultipleValues    bit = 1
);
```

@PID: The unique identifier of the managed property. This value MUST NOT be NULL.

@HasMultipleValues: 1 indicates that the managed property can have multiple values for a single item; otherwise it MUST be 0. Upon successful execution, the HasMultipleValues attribute of the managed property MUST be set to this value.

@OverrideHasMultipleValues: A flag indicating whether the @HasMultipleValues parameter is the sole mechanism for the protocol client to use for determining whether the managed property is multi-valued. If the value of the @OverrideHasMultipleValues parameter is 1 then the protocol client MUST NOT ignore the value of the @HasMultipleValues parameter, otherwise the protocol client MUST use logic to determine whether the managed property is multi-valued.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<48> return any result sets. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.153 proc_MSS_SetPendingMappings

The **proc_MSS_SetPendingMappings** stored procedure is called to copy mappings between the specified managed property and crawled properties from the Pending Mappings Set to the Mappings Set as specified in section 3.1.1.1. This procedure updates the LastModifyTime attribute in Crawled Property Set for the corresponding crawled properties entries and the value of the CrawledPropertyTimestamp in the Metadata Timestamp Set with the current date and time in local time of the server.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_SetPendingMappings (
    @PID            int
);
```

@PID: The unique identifier of the managed property. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<49> return any result sets. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.154 proc_MSS_SetRankingModel

The **proc_MSS_SetRankingModel** stored procedure is called to set the details of a ranking model inside the Ranking Model Set, specified in section 3.1.1.5.1. If the specified ranking model exists it is updated. If the specified ranking model does not exist a new ranking model is created. Immediately after updating or creating the specified ranking model the stored procedure changes the Global Ranking Parameters Timestamp variable, specified in section 3.1.1.1, with the current local server date and time.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_SetRankingModel (
   @ModelId     uniqueidentifier,
   @IsDefault   bit,
   @ModelXml    nvarchar(max)
);
```

@ModelId: The identifier of the ranking model. This value MUST NOT be NULL.

@IsDefault: The flag indicating whether the ranking model is the default ranking model. This value MUST be set to 1 to indicate this is the default ranking model, otherwise it MUST be set to 0.

@ModelXml: The string representation of the weights and parameters of the ranking model to be created or updated. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<50> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.155 proc_MSS_SetRecrawl

The **proc_MSS_SetRecrawl** stored procedure is called to update the values of ErrorId and ErrorLevel attributes in the Crawl Url History Set, as specified in [MS-SQLPGAT2] section 3.1.1.3.Transaction scope for the given crawl history item MUST be set to 2 ([MS-SQLPGAT2] section 2.2.1.15)

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_SetRecrawl (
   @DocID      int,
   @ErrorID    int,
   @ErrorLevel int
);
```

@DocID: The unique identifier of the item. This value MUST NOT be NULL.

@ErrorID: The unique identifier of the error occurred when crawling the item. The value MUST be one of the values listed in the Crawl Error Set (section 3.1.1.3).

@ErrorLevel: The type of error occurred when crawling the item. The value MUST be a Crawl Log Error Level data type as specified in section 2.2.1.7

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<51> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.156 proc_MSS_SetSchemaParameter

The **proc_MSS_SetSchemaParameter** stored procedure is called to update values for the specified ranking parameter in the Global Ranking Parameters Set as specified in section 3.1.1.6. This procedure also changes the Global Ranking Parameters Timestamp attribute in the Metadata Timestamp Set as specified in Section 3.1.1.1 to the current date and time in local time of the server.

The T-SQL syntax for the result set is as follows:

```
PROCEDURE proc_MSS_SetSchemaParameter (
   @ParamName                nvarchar(40),
   @IsString                 bit,
   @strValue                 nvarchar(256),
   @fltValue                 float
);
```

@ParamName: The name of the ranking parameter.

@IsString: If set to 1, the *@strValue* parameter is used and the *@fltvalue* parameter is not used. Otherwise, the *@fltValue* parameter is used and the *@strValue* parameter is not used.

@strValue: A string value of the ranking parameter.

@fltValue: A floating-point value of the parameter.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Set: SHOULD NOT<52> return a result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.157 proc_MSS_SetScopeDisplayGroupInfo

The **proc_MSS_SetScopeDisplayGroupInfo** stored procedure is called to set the configuration information for the specified search scope display group. Upon successful execution the stored procedure increments the LastConsumerChangeID value in the Scope System Set and sets the LastUpdate value for the corresponding consumer in the Consumer Set to the resulting LastConsumerChangeID value. See section 3.1.1.4 for details.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_SetScopeDisplayGroupInfo (
   @DisplayGroupID        int,
   @Name                  nvarchar(60),
   @Description           nvarchar(300),
   @DisplayInAdminUI      bit,
   @Undeletable           bit,
   @DefaultScopeID        int,
   @ModifierName          nvarchar(60)
);
```

@DisplayGroupID: The unique identifier of the search scope display group. This value MUST NOT be NULL.

@Name: The name of the search scope display group. This value MUST NOT be NULL.

@Description: The description of the search scope display group. This value MUST NOT be NULL.

@DisplayInAdminUI: The bit flag indicating whether the search scope display group is displayed in the Administration user interface. The value MUST be a DisplayInAdminUI data type as specified in section 2.2.1.9**.**

@Undeletable: The bit flag indicating whether the search scope display group can be deleted. The value MUST be an Undeletable data type as specified in section 2.2.1.20**.**

@DefaultScopeID: The default search scope of the search scope display group. This value MUST NOT be NULL.

@ModifierName: Name of the user who last modified the search scope display group. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<53> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.158 proc_MSS_SetScopeDisplayGroupListItem

The **proc_MSS_SetScopeDisplayGroupListItem** stored procedure is called to add the search scope to the specified search scope display group with the specified order. The stored procedure MUST be called after the call to proc_MSS_BeginScopeDisplayGroupList stored procedure and before the call to proc_MSS_EndScopeDisplayGroupList stored procedure. See section 3.1.5.17 for the specification of proc_MSS_BeginScopeDisplayGroupList stored procedure and section 3.1.5.45 for the specification of proc_MSS_EndScopeDisplayGroupList stored procedure.

Upon successful execution the stored procedure increments the LastConsumerChangeID value in the Scope System Set and sets the LastUpdate value for the corresponding consumer in the Consumer Set to the resulting LastConsumerChangeID value. See section 3.1.1.4 for details.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_SetScopeDisplayGroupListItem (
   @DisplayGroupID         int,
   @ScopeID                int,
   @Rank                   int
);
```

@DisplayGroupID: The unique identifier of the search scope display group. This value MUST NOT be NULL.

@ScopeID: The unique identifier of the search scope. This value MUST NOT be NULL.

@Rank: The order of the search scope within the search scope display group. This parameter MUST be set to a negative value minus 1.

Return values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<54> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.159 proc_MSS_SetScopeInfo

The **proc_MSS_SetScopeInfo** stored procedure is called to set the details of the specified search scope in the search application. Upon successful execution the stored procedure modifies the CompilationState value in the Scope Set. The stored procedure also increments the LastConsumerChangeID value in the Scope System Set and sets the LastUpdate value for the corresponding consumer in the Consumer Set to the resulting LastConsumerChangeID value. See section 3.1.1.4 for details.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_SetScopeInfo (
   @ScopeID                  int,
   @Name                     nvarchar(60),
   @Description              nvarchar(300),
   @ConsumerName             nvarchar(60),
   @DisplayInAdminUI         bit,
   @AlternateResultsPageURL  nvarchar(2047) = NULL,
   @CompilationType          smallint,
   @ModifierName             nvarchar(60),
   @Filter                   nvarchar(2047) = NULL
   @LargeFilter              nvarchar(max) = NULL
);
```

@ScopeID: The unique identifier of the search scope. This value MUST NOT be NULL.

@Name: The name of the search scope. Upon successful execution, the name of the search scope MUST be set to this value. This value MUST NOT be NULL.

@Description: The description of the search scope. Upon successful execution, the description of the search scope MUST be set to this value. This value MUST NOT be NULL.

@ConsumerName: The name of the search scope consumer of the search scope. Upon successful execution, the search scope consumer name of the search scope MUST be set to this value. This value MUST NOT be NULL.

@DisplayInAdminUI: The bit flag indicating whether the search scope display group is displayed in the Administration user interface. The value MUST be a DisplayInAdminUI data type as specified in Section 2.2.1.9.

@AlternateResultsPageUrl: The URL of an alternate web page to display the results of a search performed on this search scope.

@CompilationType: The compilation type of the search scope. The value MUST be a Compilation Type data type, as specified in section 2.2.1.5**.**

@ModifierName: The name of the person modifying the search scope. This value MUST NOT be NULL.

@Filter: Reserved. This parameter MUST be ignored by the client.

@LargeFilter: Reserved. This parameter MUST be ignored by the client.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<55> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.160 proc_MSS_SetScopeRuleInfo

The **proc_MSS_SetScopeRuleInfo** stored procedure is called to set the details of the specified search scope rule in the search application. Upon successful execution the stored procedure modifies the value of CompilationState attribute in the Scope Set as follows:

If the Compilation State value is set to 0 or 2, as specified in section 2.2.1.4, the stored procedure MUST set the value 3. If the Compilation State value is set to 4, the stored procedure MUST set the value to 5. The stored procedure also increments the LastConsumerChangeID value in the Scope

System Set and sets the LastUpdate value for the corresponding consumer in the Consumer Set to the resulting LastConsumerChangeID value. See section 3.1.1.4 for details.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_SetScopeRuleInfo (
   @RuleID              int,
   @FilterBehavior      smallint,
   @UrlRuleType         smallint,
   @PropertyID          int,
   @UserValueString     nvarchar(2047),
   @ModifierName        nvarchar(60)
);
```

@RuleID: The unique identifier of the search scope rule**.** This value MUST NOT be NULL.

@FilterBehavior: The filter behavior of the search scope rule. The value MUST be a ScopeFilterBehavior data type as specified in section 2.2.1.18. Upon successful execution, the filter behavior of the search scope rule MUST be set to this value.

@UrlRuleType: The URL type of the search scope rule. The value MUST be a UrlRuleType Data type as specified in section 2.2.1.21. Upon successful execution, the URL type of the search scope rule MUST be set to this value.

@PropertyID: The unique identifier of the managed property that is associated with the search scope rule. Upon successful execution, the property identifier of the search scope rule MUST be set to this value.

@UserValueString: A user value string for the search scope rule. Upon successful execution, the user value string of the search scope rule MUST be set to this value.

@ModifierName: The name of the person modifying the search scope rule. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<56> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.161 proc_MSS_SetScopesManagerInfo

The **proc_MSS_SetScopesManagerInfo** stored procedure is called to set the details of the search scopes system in the search application.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_SetScopesManagerInfo (
   @CompilationScheduleType             smallint,
   @CustomCompilationSchedule           nvarchar(60)
);
```

@CompilationScheduleType: The search scope compilation schedule type for the search scopes system. The value MUST be a Compilation Schedule Type data type as specified in section 2.2.1.3. Upon successful execution, the compilation schedule type of the search scopes system MUST be set to this value.

@CustomCompilationSchedule: MUST be ignored by the client.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<57> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.162 proc_MSS_SetUserPreference

The **proc_MSS_SetUserPreference** stored procedure is called to set search preference information, as specified in section 3.1.1.10, for a given user. If the search preference record of the specified user does not exist on the server, the SERVER MUST create a new record, set the server configuration version to *@Version* + 1, and return the server configuration version to signal a successful update. Otherwise, the server MUST check to see if @Version matches the server configuration version. If the versions match, the server MUST update the existing record, set the server configuration version to *@Version* + 1, and return the server configuration version to signal a successful update. If the versions do not match, the server MUST return the value of *@Version* to signal a failed update.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_ SetUserPreference (
   @UserID             varbinary(512),
   @Preference         nvarchar(max),
   @Version            int
);
```

@UserID: Unique identifier, as defined by tp_SystemID of [MS-WSSFO2] section 2.2.7.10, of the user for whom to set the search preference information. This value MUST NOT be NULL and MUST NOT be empty.

@Preference: Contains an XML fragment representing the search user preference of the user specified by *@UserID*. The XML Schema for this structure is defined in section 2.2.6.4.2. This value MUST NOT be NULL and MUST NOT be empty.

@Version: The server configuration version of the search user preference instance. This value MUST NOT be NULL and MUST be greater or equal to 0. The client MUST pass in the value of the *v* attribute, as specified in section 2.2.6.3.22, of *@Preference* as the value.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: MUST return the following result set:

3.1.5.162.1 Server Configuration Version Result Set

The server configuration version result set returns the server configuration version of the user preference instance for the specified user. The client MUST update the *v* attribute, as specified in section 2.2.6.3.22, of its local user preference instance with the value of the *Version* element in the result set.

The T-SQL syntax for the result set is as follows:

```
Version               int;
```

Version: The server configuration version of the search user preference instance. This value MUST NOT be NULL and MUST be greater or equal to 0.

3.1.5.163 proc_MSS_StartScopesCompilation

The **proc_MSS_StartScopesCompilation** stored procedure is called to start search scope compilation as soon as possible.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_StartScopesCompilation();
```

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<58> return any result sets. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.164 proc_MSS_UpdateAuthorityPageUrl

The **proc_MSS_UpdateAuthorityPageUrl** stored procedure is called to modify URL of an existing authority page.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_UpdateAuthorityPageUrl(
   @OldUrl              nvarchar(2048),
   @NewUrl              nvarchar(2048),
   @NewHash             int
);
```

@OldUrl: The URL of the authority page to modify.

@NewUrl: The new URL of the authority page.

@NewHash: The identifier of the new URL. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<59> return any result sets. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.165 proc_MSS_UpdateBestBet

The **proc_MSS_UpdateBestBet** stored procedure is called to change the information about the specified best bet.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_UpdateBestBet(
   @ConsumerGpId         nvarchar(50),
   @BestBetId            int,
   @Title                nvarchar(100),
   @Url                  nvarchar(2048),
   @Description           nvarchar(500)
```

```
);
```

@ConsumerGpId: A string that uniquely identifies the keyword consumer group for the best bet.

@BestBetId: The unique identifier of the best bet to update. This value MUST NOT be NULL.

@Title: A string that contains the title for the best bet. This value MUST NOT be NULL.

@Url: A string that contains the URL for the best bet. This value MUST NOT be NULL.

@Description: A string that contains the description for the best bet.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<60> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.166 proc_MSS_UpdateBestBetOrder

The **proc_MSS_UpdateBestBetOrder** stored procedure is called to update the order of the specified best bet within all best bets associated with the specified keyword.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_UpdateBestBetOrder(
   @SpecialTermId    int,
   @BestBetId        int,
   @Order            int
);
```

@SpecialTermId: The unique identifier of the keyword. This value MUST NOT be NULL.

@BestBetId: The unique identifier of the best bet. This value MUST NOT be NULL.

@Order: The new order of the specified best bet within all best bets associated with the specified keyword. This value MUST NOT be NULL.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<61> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.5.167 proc_MSS_UpdateProxy

The **proc_MSS_UpdateProxy** stored procedure is called to updates the value of UseProxySettings attribute in Federation Set, specified in section 3.1.1.7, which indicates whether or not to use the proxy settings for federated locations. The proxy settings MUST be used by all federated locations.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_UpdateProxy (
   @UseCrawlProxy         bit
);
```

@UseCrawlProxy: A 1-bit number that indicates whether all federated locations use the proxy settings or not. MUST be 1 if all federated locations use the proxy settings. Otherwise, it MUST be 0.

Return Values: An integer which MUST be 0 on successful execution. Any other value indicates error.

Result Sets: MUST NOT return any result set.

3.1.5.168 proc_MSS_UpdateRankingModelOM

The **proc_MSS_UpdateRankingModelOM** stored procedure modifies a ranking model in the Ranking Model Set, specified in section 3.1.1.5.1, with the specified string representation of weights and parameters. Immediately after updating the specified ranking model the stored procedure changes the Global Ranking Parameters Timestamp variable, specified in section 3.1.1.1, with the current date and time.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_UpdateRankingModelOM (
   @ModelId       uniqueidentifier,
   @ModelXml      nvarchar(max)
);
```

@ModelId: The unique identifier of the ranking model to be updated. This value MUST NOT be NULL.

@ModelXml: The string representation of weights and parameters with which to update the given ranking model in the Ranking Model Set, specified in section 3.1.1.5.1.

Return Code Values:

Value	Description
0	Successful execution.
1	The ranking model specified by @ModelId was not found and the stored procedure did not make any modifications.

Result Sets: MUST NOT return any result set.

3.1.5.169 proc_MSS_UpdateSpecialTerm

The **proc_MSS_UpdateSpecialTerm** stored procedure is called to update the information about the specified keyword.

The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE proc_MSS_UpdateSpecialTerm(
   @ConsumerGpId       nvarchar(50),
   @SpecialTermId      int,
   @Term               nvarchar(100),
   @Definition         nvarchar(500) = null,
   @Contact            nvarchar(50) = null,
   @StartDate          datetime,
   @EndDate            datetime = null,
```

```
    @ReviewDate         datetime = null
);
```

@ConsumerGpId: A string that uniquely identifies the keyword consumer group. This value MUST be ignored by the server.

@SpecialTermId: The unique identifier of the newly added keyword. This value MUST NOT be NULL.

@Term: The term for the keyword. This value MUST NOT be NULL.

@Definition: The description of the keyword.

@Contact: The contact name for the keyword.

@StartDate: A datetime value that specifies when the keyword begins to appear in search result. This value MUST NOT be NULL.

@EndDate: A datetime value that specifies when the keyword stops appearing in search result.

@ReviewDate: A datetime value that specifies when the keyword is expected to be reviewed.

Return Code Values: This stored procedure returns an integer value that MUST be ignored.

Result Sets: SHOULD NOT<62> return any result set. The protocol client MUST ignore any result sets returned by this stored procedure.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Client Details

None

3.2.1 Abstract Data Model

None.

3.2.2 Timers

None.

3.2.3 Initialization

None.

3.2.4 Higher-Layer Triggered Events

None.

3.2.5 Message Processing Events and Sequencing Rules

None.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

4 Protocol Examples

This section provides specific example scenarios for end-to-end administration tasks.

4.1 Crawled Properties Administration

This section illustrates the protocol operations required to add, change, and retrieve crawled property categories and their associated crawled properties from the database on the back-end database server**.**

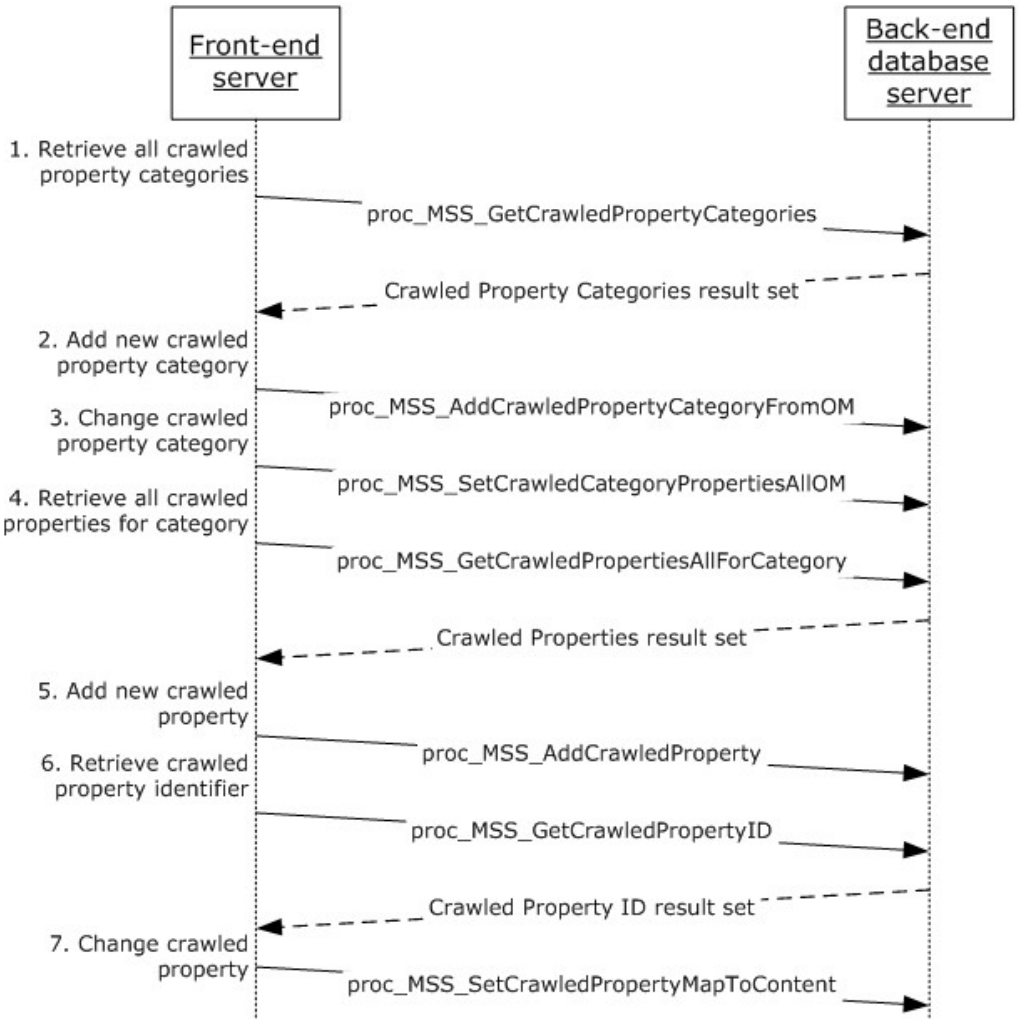

Figure 8: Crawled Properties Administration

The steps in the preceding diagram are explained in the following list.

1. This example begins with the protocol client retrieving the list of all crawled property categories from the metadata schema to determine if a crawled property category already exists.

2. If the crawled property category does not exist in the result set returned from the call to the proc_MSS_GetCrawledPropertyCategories stored procedure, the protocol client calls the

proc_MSS_AddCrawledPropertyCategoryFromOM stored procedure to add the new crawled property category to the metadata schema.

3. To change data for the existing crawled property category, the protocol client calls the proc_MSS_SetCrawledCategoryPropertiesAllOM stored procedure.

4. To add a new crawled property to the metadata schema, the protocol client first retrieves a list of crawled properties associated with the specified crawled property category to determine if the crawled property already exists.

5. If the crawled property is not returned as part of the Crawled Properties result set, the client calls the proc_MSS_AddCrawledProperty stored procedure to add it to the metadata schema and associate it with the crawled property category.

6. To change the IsMappedToContent attribute for the existing crawled property, the protocol client first calls the proc_MSS_GetCrawledPropertyID stored procedure to retrieve the unique identifier for the specified crawled property.

7. The client then calls the proc_MSS_SetCrawledPropertyMapToContent stored procedure, passing crawled property and the new value for the IsMappedToContent attribute.

4.2 Managed Properties Administration

This section illustrates the protocol operations required to add, change, and retrieve managed properties and associate them with the crawled properties in the database on the back-end database server**.**

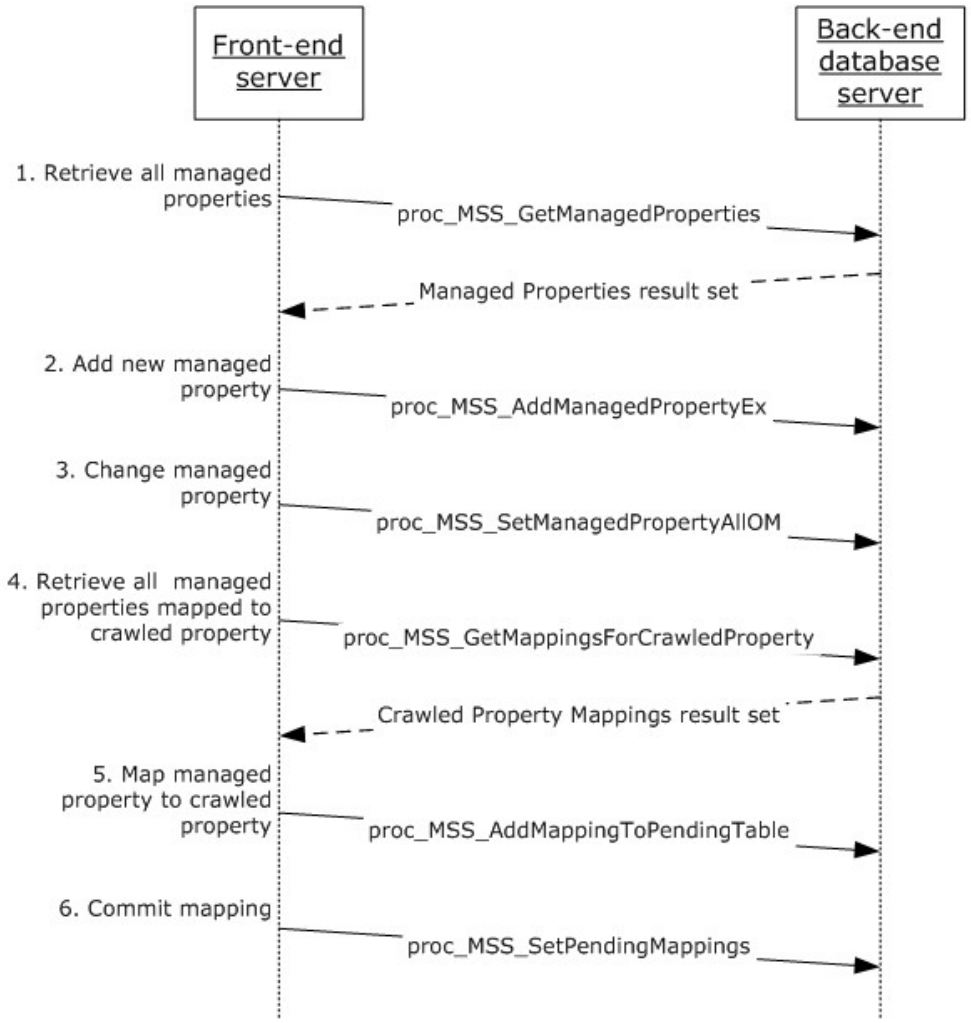

Figure 9: Managed Properties Administration

The steps in the preceding diagram are explained in the following list.

1. This example begins with the protocol client retrieving the list of all managed properties from the metadata schema to determine if a managed property already exists.

2. If the managed property does not exist in the result set returned from the call to proc_MSS_GetManagedProperties, as specified in [MS-SQLPQ2] section 3.1.5.3, the protocol client calls the proc_MSS_AddManagedPropertyEx stored procedure to add the new managed property to the metadata schema.

3. To change data for the existing managed property, the protocol client calls theproc_MSS_SetManagedPropertyAllOM stored procedure.

4. To map the managed property to the crawled property, the protocol client first retrieves a list of all managed properties associated with the specified crawled property to determine if the managed property is already mapped.

5. If the managed property is not returned as part of the Crawled Property Mappings result set, the client calls the proc_MSS_AddMappingToPendingTable stored procedure to set the mapping between the managed property and the crawled property.

6. The mapping between the managed property and the crawled property is saved with the call to proc_MSS_SetPendingMappings stored procedure.

5 Security

5.1 Security Considerations for Implementers

Security for this protocol is controlled by the access rights to the databases on the back-end database server, which is negotiated as part of the Tabular Data Stream [MS-TDS] protocol.

5.2 Index of Security Parameters

None.

6 Appendix A: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include released service packs:

- Microsoft® FAST™ Search Server 2010
- Microsoft® SharePoint® Foundation 2010
- Microsoft® SQL Server® 2005
- Microsoft® SQL Server® 2008
- Microsoft® SQL Server® 2008 R2

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 3.1.5.1: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<2> Section 3.1.5.2: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<3> Section 3.1.5.3: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<4> Section 3.1.5.4: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<5> Section 3.1.5.5: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<6> Section 3.1.5.6: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<7> Section 3.1.5.7: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<8> Section 3.1.5.8: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<9> Section 3.1.5.9: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<10> Section 3.1.5.10: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<11> Section 3.1.5.12: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<12> Section 3.1.5.13: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<13> Section 3.1.5.14: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<14> Section 3.1.5.15: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<15> Section 3.1.5.16: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<16> Section 3.1.5.17: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<17> Section 3.1.5.18: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<18> Section 3.1.5.21: [1] If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<19> Section 3.1.5.22: [1] If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<20> Section 3.1.5.23: [1] If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<21> Section 3.1.5.24: [1] If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<22> Section 3.1.5.25: [1] If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<23> Section 3.1.5.26: [1] If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<24> Section 3.1.5.27: [1] If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<25> Section 3.1.5.28: [1] If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<26> Section 3.1.5.29: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<27> Section 3.1.5.30: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<28> Section 3.1.5.31: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<29> Section 3.1.5.32: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<30> Section 3.1.5.33: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<31> Section 3.1.5.35: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<32> Section 3.1.5.36: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<33> Section 3.1.5.37: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<34> Section 3.1.5.39: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<35> Section 3.1.5.40: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<36> Section 3.1.5.41: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<37> Section 3.1.5.42: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<38> Section 3.1.5.43: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<39> Section 3.1.5.44: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<40> Section 3.1.5.45: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<41> Section 3.1.5.51: [1] If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<42> Section 3.1.5.101: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<43> Section 3.1.5.143: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<44> Section 3.1.5.146: [1] If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<45> Section 3.1.5.148: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<46> Section 3.1.5.149: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<47> Section 3.1.5.151: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<48> Section 3.1.5.152: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<49> Section 3.1.5.153: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<50> Section 3.1.5.154: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<51> Section 3.1.5.155: [1] If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<52> Section 3.1.5.156: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<53> Section 3.1.5.157: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<54> Section 3.1.5.158: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<55> Section 3.1.5.159: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<56> Section 3.1.5.160: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<57> Section 3.1.5.161: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<58> Section 3.1.5.163: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<59> Section 3.1.5.164: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<60> Section 3.1.5.165: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<61> Section 3.1.5.166: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

<62> Section 3.1.5.169: If a given stored procedure does an INSERT, UPDATE, or DELETE SQL operation in the database, the stored procedure returns one or more extra result sets that contain the number of records affected by the operation.

7 Change Tracking

No table of changes is available. The document is either new or has had no changes since its last release.

8 Index

E

F

G

H

I

K

L

M

N

O

P

Q

T