[MS-SPSCLSP]:
SPSCrawl Stored Procedures Protocol Specification

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation for protocols, file formats, languages, standards as well as overviews of the interaction among each of these technologies.
- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you may make copies of it in order to develop implementations of the technologies described in the Open Specifications and may distribute portions of it in your implementations using these technologies or your documentation as necessary to properly document the implementation. You may also distribute in your implementation, with or without modification, any schema, IDL's, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications.
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.
- **Patents.** Microsoft has patents that may cover your implementations of the technologies described in the Open Specifications. Neither this notice nor Microsoft's delivery of the documentation grants any licenses under those or any other Microsoft patents. However, a given Open Specification may be covered by Microsoft's Open Specification Promise (available here: http://www.microsoft.com/interop/osp) or the Community Promise (available here: http://www.microsoft.com/interop/cp/default.mspx). If you would prefer a written license, or if the technologies described in the Open Specifications are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **Trademarks.** The names of companies and products contained in this documentation may be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights.
- **Fictitious Names.** The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

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

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

Revision Summary

Date	Revision History	Revision Class	Comments
04/04/2008	0.1		Initial Availability
06/27/2008	1.0	Major	Revised and edited the technical content
12/12/2008	1.01	Editorial	Revised and edited the technical content
07/13/2009	1.02	Major	Changes made for template compliance
08/28/2009	1.03	Editorial	Revised and edited the technical content
11/06/2009	1.04	Editorial	Revised and edited the technical content
02/19/2010	2.0	Editorial	Revised and edited the technical content
03/31/2010	2.01	Editorial	Revised and edited the technical content
04/30/2010	2.02	Editorial	Revised and edited the technical content
06/07/2010	2.03	Editorial	Revised and edited the technical content
06/29/2010	2.04	Editorial	Changed language and formatting in the technical content.
07/23/2010	2.04	No change	No changes to the meaning, language, or formatting of the technical content.
09/27/2010	2.04	No change	No changes to the meaning, language, or formatting of the technical content.
11/15/2010	2.04	No change	No changes to the meaning, language, or formatting of the technical content.
12/17/2010	2.04	No change	No changes to the meaning, language, or formatting of the technical content.
03/18/2011	2.04	No change	No changes to the meaning, language, or formatting of the technical content.
06/10/2011	2.04	No change	No changes to the meaning, language, or formatting of the technical content.

Table of Contents

1 Introduction

This document provides specific details of SPS Crawl Stored Procedures protocol. This protocol allows clients to read values of **user profile** properties for user profiles within the context of a **site**.

1.1 Glossary

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

GUID

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

bucket
crawl
front-end Web server
quick link
result set
return code
Shared Services Provider (SSP)
site
stored procedure
Transact-Structured Query Language (T-SQL)
user display name
user profile

The following terms are specific to this document:

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

1.2 References

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.

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

[MS-SQL] 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://msdn.microsoft.com/en-us/library/dd631854(v=SQL.10).aspx

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

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

[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

1.2.2 Informative References

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

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

1.3 Protocol Overview (Synopsis)

This protocol allows clients to read values of user profile properties for user profiles within the context of a site.

The following diagram shows data flow between protocol client and protocol server.

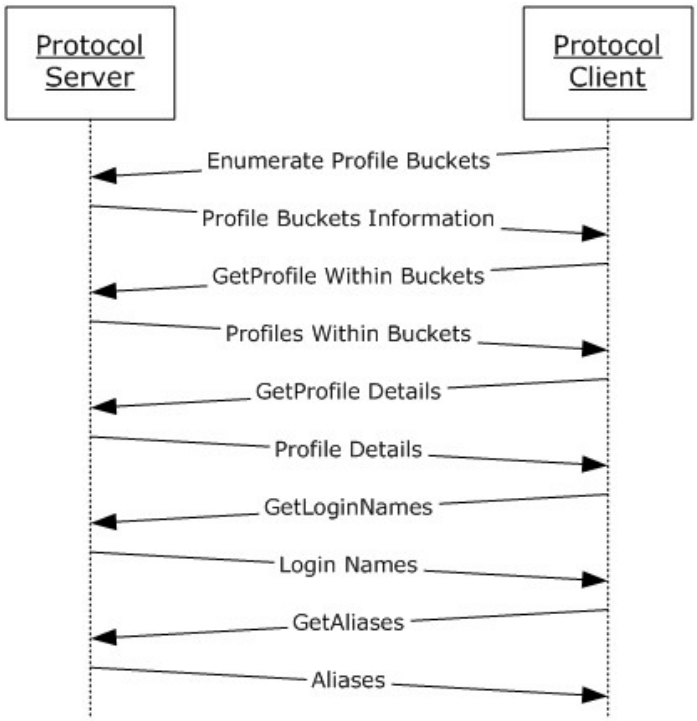

Figure 1: SPS Crawl Stored Procedure Protocol data flow between client and server

The protocol client requests the protocol server to provide a list of all **buckets**. After the protocol server provides information about all the buckets, the protocol client requests the server to enumerate the user profiles in each bucket. Once this information is provided by the protocol server, protocol client requests the protocol server to provide details of each user profile.

GetLoginNames operation requests the protocol server to provide login names of all users in the specified bucket.

The **GetAliases** operation provides the aliases of all users in the specified bucket on the protocol server.

1.4 Relationship to Other Protocols

The following diagram shows the transport stack that the protocol uses:

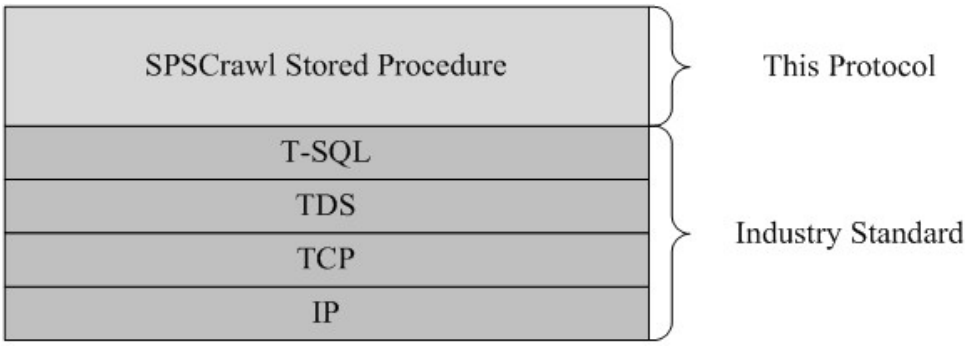

Figure 2: This protocol in relation to other protocols

1.5 Prerequisites/Preconditions

The protocol requires that a **Shared Services Provider (SSP)** is created and is configured correctly on the protocol server.

1.6 Applicability Statement

SPS Crawl Stored Procedures Protocol is well suited for a client to read up to one million user profile records.

1.7 Versioning and Capability Negotiation

Versions of the data structures or **stored procedures** in the database need to be the same as expected by the **front-end Web server**. If the stored procedures do not provide the calling parameters or return values as expected, the results of the call are indeterminate.

The version negotiation process for this protocol is identical to the process defined in [MS-WSSFO] section 1.7.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

None.

2.2 Common Data Types

This section contains common definitions used by this protocol.

2.2.1 Simple Data Types and Enumerations

None.

2.2.2 Bit Fields and Flag Structures

None.

2.2.3 Binary Structures

None.

2.2.4 Result Sets

None.

2.2.5 Tables and Views

None.

2.2.6 XML Structures

None.

2.2.6.1 Namespaces

None.

2.2.6.2 Simple Types

None.

2.2.6.3 Complex Types

None.

2.2.6.4 Elements

None.

2.2.6.5 Attributes

None.

2.2.6.6 Groups

None.

2.2.6.7 Attribute Groups

None.

3 Protocol Details

3.1 SPSCrawl Server Details

3.1.1 Abstract Data Model

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

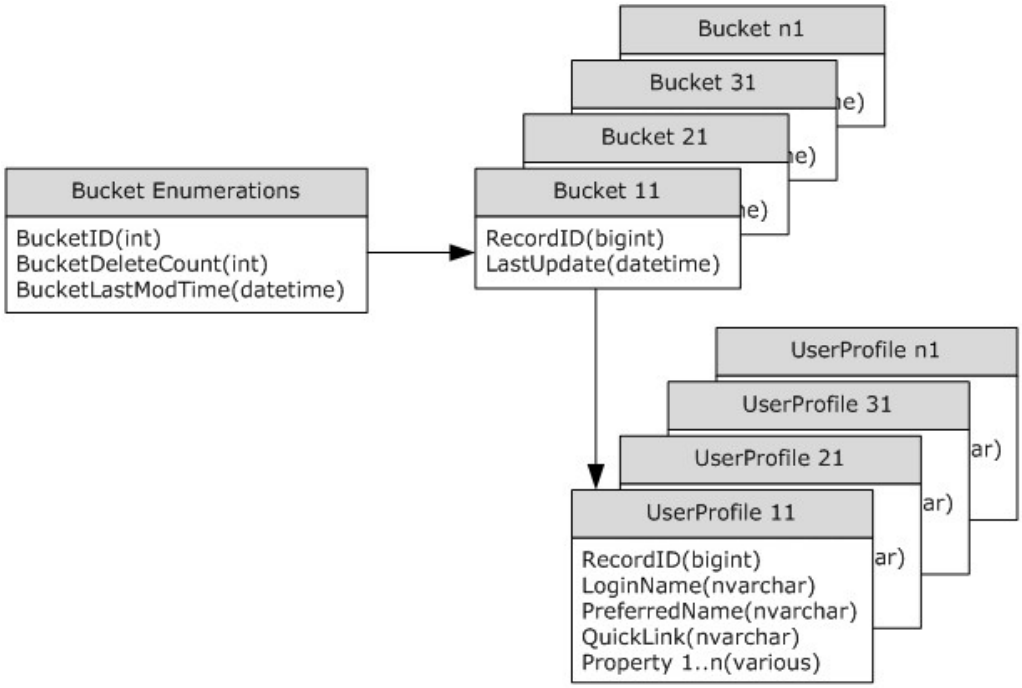

Figure 3: Abstract data model

In Figure 3, 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.

Bucket Enumerations Table: A collection of entries corresponding to the table of information about buckets in the dataset relating to user profiles. A unique **BucketID** MUST identify each entry.

BucketID: A unique identifier assigned to each user profile bucket.

- **BucketDeleteCount**: The number of user profiles deleted from the bucket identified by **BucketID**.
- **BucketLastModTime**: The date and time of the latest update to any user profile enumerated in the bucket identified by **BucketID**.

Bucket1 ... Bucket*n*: A collection of entries corresponding to the tables of user profile buckets in the dataset. A unique **RecordID** MUST identify each entry.

- **RecordID**: An identifier assigned to each user profile.
- **LastUpdate**: The date and time of the last update to the user profile identified by **RecordID**.

UserProfile1...UserProfile*n*: A collection of identifiers and user profile properties for each user profile in the dataset. A unique **RecordID** MUST identify each entry.

- **RecordID**: An identifier assigned to each user profile.
- **LoginName**: The login name for the user profile identified by **RecordID**.
- **PreferredName**: The **user display name** for the user profile identified by **RecordID**.
- **QuickLink**: One or more **quick link** values for the user profile identified by **RecordID**.
- **Property1...Property*n***: Additional entries that MAY be defined and populated for a specific dataset implementation. The entries MAY represent values for additional identifiers and user profile properties. The procedures that support **crawl** actions pass these values on to the caller as described in the following sections without modifying the values.

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Message Processing Events and Sequencing Rules

The following table summarizes the types that are defined in this specification.

Operation	Description
profile_EnumProfileBuckets	Used to request a list of user profile bucket identifiers.
profile_EnumProfileInBucket	Used to request a list of the identifiers and last update times for each user profile in a user profile bucket.
profile_EnumProfileRecords	Used to request property values for a user profile.
profile_EnumUserIDs	Used to request a list of all login names.
profile_GetAliasList	Used to request a list of user profile alias values.

3.1.4.1 profile_EnumProfileBuckets

The **profile_EnumProfileBuckets** stored procedure is called to get user profile bucket information.

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

```
PROCEDURE profile_EnumProfileBuckets ();
```

Return Code Values: profile_EnumProfileBuckets returns an integer **return code** which MUST be 0.

Result Sets: MUST return a single **result set** as follows:

3.1.4.1.1 Profile Buckets Result Set

The Profile Buckets Result Set returns multiple rows, each containing three columns. The result set will be empty if no user profile bucket was found.

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

```
BucketID                    int,
BucketDeleteCount           int,
BucketLastModTime           datetime;
```

BucketID: The identifier of the user profile bucket.

BucketDeleteCount: The number of deleted records in the corresponding user profile bucket.

BucketLastModTime: The value of the most recent update on records in the corresponding user profile bucket.

3.1.4.2 profile_EnumProfileInBucket

The **profile_EnumProfileInBucket** stored procedure is called to get identifiers for user profiles contained in the specified user profile bucket.

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

```
PROCEDURE profile_EnumProfileInBucket (
    @BucketID                int
);
```

@BucketID: The identifier of the user profile bucket.

Return Code Values: profile_EnumProfileInBucket returns an integer return code which MUST be 0 to indicate success.

Result Sets: MUST return a single result set as follows:

3.1.4.2.1 Profile In Bucket Result Set

The Profile In Bucket Result Set returns multiple rows, each containing two columns. The result set will be empty if no user profiles were found in the user profile bucket specified by the provided *@BucketID* parameter.

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

```
RecordID                bigint,
LastUpdate              datetime;
```

RecordID: The identifier of the user profile .

LastUpdate: The value of the last update on the user profile

3.1.4.3 profile_EnumProfileRecords

The **profile_EnumProfileRecords** stored procedure is called to get information for a specified user profile.

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

```
PROCEDURE profile_EnumProfileRecords (
    RecordID                    bigint
);
```

RecordID: The value of a user profile identifier.

Return Code Values: profile_EnumProfileRecords returns an integer return code which MUST be 0 to indicate success.

Result Sets: MUST return three result sets in sequence as follows:

3.1.4.3.1 User Profile Information Result Set

The User Profile Information Result Set returns multiple rows, each containing seven columns. The result set MUST be returned first, and MUST be empty if no records were found matching the provided **RecordID** parameter.

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

```
RecordID                    bigint,
UserID                      uniqueidentifier,
PropertyID                  bigint,
Privacy                     int,
PropertyVal                 nvarchar(3600),
VocValValue                 nvarchar(3600),
LastUpdate                  datetime;
```

RecordID: The identifier of the user profile.

UserID: The **GUID** of the user profile.

PropertyID: The identifier of the user profile property.

Privacy: A value that specifies the visibility option for the current user profile property.

PropertyVal: The string representation of the user profile property value for the current **RecordID** on current **PropertyID**, when user profile property is of Single Value type.

VocValValue: The string representation or the user profile property value for the current **RecordID** on current **PropertyID**, when user profile property is of Multi-Value type.

LastUpdate: The value of the last update on the corresponding user profile.

The User Profile Information Result Set will be returned, and ordered by the **PropertyID** column.

3.1.4.3.2 NT Name Result Set

The NT Name Result Set returns multiple rows, each containing a single column. The result set MUST be returned second. The result set MUST be empty if either the user profile identified by **RecordID** has only one login name, or the **RecordID** specifies an optional secondary login name for a user profile in which the administrator wishes to allow multiple login names.

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

```
NTName                    nvarchar(400);
```

NTName: A non-primary login name for the provided **RecordID** parameter, if the **RecordID** identifies the primary record for a user profile with multiple login names.

3.1.4.3.3 Quick Link Result Set

The Quick Link Result Set returns multiple rows, each containing a single column. The result set MUST be returned third, and MUST be empty if no records were found matching the provided **RecordID** parameter.

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

```
QuickLink                 nvarchar(250);
```

QuickLink: A quick link.

3.1.4.4 profile_EnumUserIDs

The **profile_EnumUserIDs** stored procedure is called to get a list of all user profile login names.

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

```
PROCEDURE profile_EnumUserIDs ();
```

Return Code Values: profile_EnumUserIDs returns an integer return code which MUST be 0 to indicate success.

Result Sets: MUST return a single result set as defined below:

3.1.4.4.1 User Identifiers Result Set

The User Identifiers Result Set contains multiple rows each containing a single column.The result set will be empty if no users were found.

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

```
NTName                    nvarchar(400);
```

NTName: An NT name.

3.1.4.5 profile_GetAliasList

The **profile_GetAliasList** stored procedure returns a list of user profile aliases.

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

```
PROCEDURE profile_GetAliasList (
    @StartTime                datetime = NULL,
    @LastUpdate               datetime OUTPUT
);
```

@StartTime: A value to be used as filter.

@LastUpdate: The most recent update date and time among all user profiles.

Return Code Values: profile_GetAliasList returns an integer return code which MUST be 0 to indicate success.

profile_GetAliasList also returns the *@LastUpdate* output parameter set to the most recent update date and time of all user profiles.

Result Sets: MUST return a single result set as follows:

3.1.4.5.1 Get Alias List Result Set

The Get Alias List Result Set contains multiple rows each containing three columns.

If *@StartTime* = NULL, the result set MUST contain a set of rows for all user profile aliases.

If *@StartTime* contains a datetime, the result set MUST contain a set of rows containing aliases for each user profile updated after *@StartTime*, and it MUST be empty if no user profiles updated after *@StartTime* were found.

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

```
RecordID                bigint,
NAME                    nvarchar(512) NOT NULL,
FLAG                    int;
```

RecordID: A user profile identifier.

NAME: A value for a user profile property marked as an alias.

FLAG: A value that specifies if the NAME column value is a user display name. Valid values are listed in the following table.

Value	Description
0	The NAME column value is not the user display name.
1	The NAME column value is the user display name.

3.1.5 Timer Events

None.

3.1.6 Other Local Events

None.

4 Protocol Examples

A caller uses the five stored procedures described in this document to crawl a dataset that contains user profiles to create one or more indices of that data. The caller may crawl subsets of the dataset based on user profile buckets or crawl the entire dataset.

4.1 Crawl Example Using User Profile Buckets

To crawl based on user profile buckets, the caller first uses **profile_EnumProfileBuckets** to determine the range of user profile bucket identifiers, called **BucketIDs** in this example. The return set from **profile_EnumProfileBuckets** also contains the most recent update date and time for all of the user profiles in each bucket, and the caller may use information cached from previous crawls to ignore buckets that contain only user profiles unchanged since the last crawl.

The caller then uses one of the **BucketIDs** as the input parameter for a call to **profile_EnumProfileInBucket**, which returns an identifier for each user profile in the user profile bucket, called the **RecordID** in this example. The procedure also returns the date and time of the most recent update for each user profile. The caller may use information cached from previous crawls to ignore user profiles unchanged since the last crawl.

The **RecordID** identifies each user profile in the dataset. The caller can use a **RecordID** as an input to **profile_EnumProfileRecords** to get several sets of user profile property values for the user profile for indexing, or to retrieve user profile property values for a user profile previously indexed.

The caller creates its indices by making multiple calls to **profile_EnumProfileRecords** for all **RecordIDs** it identifies as appropriate for indexing.

4.2 Crawl Example Using the Full Dataset

The caller may crawl the full dataset without first making calls to the stored procedures that support user profile buckets. The caller may choose to do that if it has existing indices on the dataset and needs to identify any user profiles that require re-indexing. The caller may also crawl the dataset to get alias values for one or more user profiles without the overhead required for a call to **profile_EnumProfileRecords**.

4.2.1 Crawling to Request user profile login names

The caller can use **profile_EnumUserIDs** to get a return set of all user profile login names in the dataset. The return set contains only the user profile login names and does not contain the associated **RecordIDs**. The caller must use cached index information to locate the **RecordID** associated with a specific user profile login name in the dataset.

4.2.2 Crawling to Request User Profile Alias Values

The caller can use **profile_GetAliasList** to get a return set of the user profile alias values for a subset of user profiles, or for all user profiles in the dataset. The return from **profile_GetAliasList** flags the user display name in each set of aliases for each user profile.

The caller supplies a **StartTime** as an input to **profile_GetAliasList** and the procedure returns aliases for only those user profiles updated after **StartTime**. The caller can selectively update its indices by using a **StartTime** based on the update times for recently cached indices. Using a NULL **StartTime** requests alias values for all user profiles.

The caller also provides a **LastUpdate** output parameter to **profile_GetAliasList** and the procedure uses **LastUpdate** to return the date and time of the most recently updated user profile in

the dataset. The caller can use the returned **LastUpdate** value to verify that it has all expected updates indexed.

5 Security

5.1 Security Considerations for Implementers

This protocol supports the SSPI and SQL Security Authentication Methods with the Protocol Server role. These authentication methods are defined in [MS-TDS].

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® Office SharePoint® Server 2007
- 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.

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

R

S

T

V

X