

[MS-WSSCADM2]: Windows SharePoint Services Content Database Administrative Communications Version 2 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
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	Minor	Updated 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	1.04	No change	No changes to the meaning, language, or formatting of the technical content.
09/27/2010	1.04	No change	No changes to the meaning, language, or formatting of the technical content.
11/15/2010	1.05	Major	Significantly changed the technical content.
12/17/2010	1.05	No change	No changes to the meaning, language, or formatting of the technical content.
03/18/2011	1.05	No change	No changes to the meaning, language, or formatting of the technical content.
06/10/2011	1.05	No change	No changes to the meaning, language, or formatting of the technical content.

Table of Contents

1	Introduction	8
1.1	Glossary	8
1.2	References.....	10
1.2.1	Normative References.....	10
1.2.2	Informative References	11
1.3	Protocol Overview (Synopsis)	11
1.3.1	Auditing Operations	11
1.3.2	Quota Management Operations	11
1.3.2.1	Query and Update Quota Operations	11
1.3.2.2	Query and Update Usage Operations	12
1.3.2.3	Query Warn Operations	12
1.3.3	Recycle Bin Operations.....	12
1.3.3.1	Query Operations	12
1.3.3.2	Administration Operations	12
1.3.3.2.1	Delete Operations	12
1.3.3.2.2	Restore Operations.....	13
1.3.4	Security Operations	13
1.3.4.1	Operations Related to External Security Provider	13
1.3.4.2	Operations Related to ACL	13
1.3.4.3	User and Group Operations	13
1.3.5	Database Integrity and Maintenance Operations.....	14
1.3.5.1	Orphaned Objects Management.....	14
1.3.5.2	Dead Web Management.....	14
1.3.5.3	Maintenance Operations	14
1.3.6	Query Operations	14
1.3.6.1	Filtered Query Operations	14
1.3.6.2	Paged Query Operations	14
1.4	Relationship to Other Protocols.....	14
1.5	Prerequisites/Preconditions	15
1.6	Applicability Statement.....	15
1.7	Versioning and Capability Negotiation.....	15
1.8	Vendor-Extensible Fields.....	15
1.9	Standards Assignments	15
2	Messages.....	16
2.1	Transport.....	16
2.2	Common Data Types	16
2.2.1	Simple Data Types and Enumerations	16
2.2.2	Simple Data Types	16
2.2.2.1	Audit Event Source	16
2.2.2.2	Audit Event Type	16
2.2.2.3	Delete Item Type.....	17
2.2.2.4	Recycle Bin Stage	18
2.2.3	Bit Fields and Flag Structures.....	18
2.2.4	Enumerations	18
2.2.5	Binary Structures	18
2.2.6	Common Result Sets	18
2.2.6.1	Site Collection with No Sites Result Set	18
2.2.6.2	Sites with No Site Collection Result Set	18
2.2.6.3	Sites with No Parent Site Result Set.....	19

2.2.6.4	Folders with No Site Result Set.....	19
2.2.6.5	Orphaned Lists Result Set.....	19
2.2.6.6	User Storage Info Result Set.....	20
2.2.7	SQL Structures	20
2.2.8	Tables and Views	20
2.2.8.1	RecycleBin Table	21
2.2.9	XML Structures	22
3	Protocol Details.....	23
3.1	Back-End Database Server Details	23
3.1.1	Abstract Data Model	23
3.1.1.1	Audit Operations.....	25
3.1.1.2	Quota Management Operations	25
3.1.1.3	Recycle Bin Operations.....	27
3.1.1.3.1	Recycle Bin.....	27
3.1.1.3.2	Query Operations	28
3.1.1.3.3	Delete Operations	28
3.1.1.3.4	Restore Operations.....	29
3.1.1.4	Security Operations	29
3.1.1.4.1	Operations Related to External Security Provider.....	29
3.1.1.4.2	Operations Related to ACL.....	30
3.1.1.4.3	Operations Related to User and Group	31
3.1.2	Timers	31
3.1.3	Initialization	31
3.1.4	Higher-Layer Triggered Events.....	31
3.1.5	Message Processing Events and Sequencing Rules.....	31
3.1.5.1	fn_CompareTZTransitionDate	31
3.1.5.2	fn_EscapeForLike.....	32
3.1.5.3	fn_GetRootFolder.....	33
3.1.5.4	fn_HtmlEncode.....	33
3.1.5.5	fn_IsOverQuotaOrWriteLocked	34
3.1.5.6	fn_LocalDayFromUTCDate	34
3.1.5.7	proc_AddAuditEntry	35
3.1.5.8	proc_AddAuditEntryUrl	36
3.1.5.9	proc_CalculateAndUpdateSiteDiskUsed	38
3.1.5.10	proc_ConfirmSiteUsage	38
3.1.5.11	proc_ConvertStringToDate	39
3.1.5.12	proc_CountAuditEntries	40
3.1.5.13	proc_DefragmentIndices	40
3.1.5.14	proc_DeleteRecycleBinItem	41
3.1.5.15	proc_DetectOrphans.....	42
3.1.5.15.1	Site Collection with No Sites Result Set	42
3.1.5.15.2	Sites with No Site Collection Result Set	42
3.1.5.15.3	Sites with No Parent Site Result Set.....	42
3.1.5.15.4	Folders with No Site Result Set	42
3.1.5.15.5	Orphaned Lists Result Set.....	42
3.1.5.16	proc_DetectOrphansFix.....	43
3.1.5.16.1	Site Collection with No Sites Result Set	43
3.1.5.16.2	Sites with No Site Collection Result Set	43
3.1.5.16.3	Sites with No Parent Site Result Set.....	43
3.1.5.16.4	Folders with No Site Result Set	43
3.1.5.16.5	Orphaned Lists Result Set.....	43
3.1.5.17	proc_DTSetRelationship	44

3.1.5.18	proc_EnumRecycleBinItemsForCleanup.....	44
3.1.5.18.1	Recycle Bin Items For Cleanup Result Set.....	45
3.1.5.19	proc_EnumRecycleBinToFreeSecondStageQuota	45
3.1.5.19.1	Item Metadata Result Set.....	46
3.1.5.20	proc_EnumSitesForDeadWebCheck	46
3.1.5.20.1	Site Information Result Set.....	46
3.1.5.21	proc_ForceDeleteList.....	47
3.1.5.22	proc_GetAdminRecycleBinInfo	47
3.1.5.23	proc_GetAdminRecycleBinItems.....	48
3.1.5.23.1	Admin Recycle Bin Items Result Set.....	48
3.1.5.24	proc_GetAllSPWebIdentifiersGivenSiteGuid	50
3.1.5.24.1	All SPWeb Identifiers Given Site Guid Result Set	50
3.1.5.25	proc_GetAuditEntries.....	50
3.1.5.25.1	Get Audit Entries Result Set.....	51
3.1.5.26	proc_GetCustomizedDocumentsInWeb	52
3.1.5.26.1	DocumentID Result Set	52
3.1.5.27	proc_GetDeadWebInfo.....	53
3.1.5.27.1	Dead Web Information Result Set.....	53
3.1.5.28	proc_GetDocLibrarySizes	53
3.1.5.28.1	Document Library Size Result Set.....	54
3.1.5.29	proc_GetDocSizeInfo.....	55
3.1.5.29.1	Document Size Result Set	55
3.1.5.30	proc_GetFirstUniqueAncestorWebUrl	56
3.1.5.30.1	First Ancestor Site Url Result Set.....	56
3.1.5.31	proc_GetListBestMatch	57
3.1.5.32	proc_GetListSizes	57
3.1.5.32.1	Lists Size Result Set	57
3.1.5.33	proc_GetListSubset	58
3.1.5.33.1	Get List Subset Result Set	59
3.1.5.34	proc_GetRecycleBinItemInfo	59
3.1.5.35	proc_GetRecycleBinItems	60
3.1.5.35.1	Recycle Bin Items Result Set.....	61
3.1.5.36	proc_GetSitecollectionBestMatch.....	62
3.1.5.37	proc_GetSiteCollectionSubset.....	63
3.1.5.37.1	Get Site Collection Subset Result Set	63
3.1.5.38	proc_GetSiteQuota.....	64
3.1.5.38.1	Quota Information Result Set.....	64
3.1.5.39	proc_GetSiteUsage.....	65
3.1.5.39.1	Usage Totals Result Set.....	65
3.1.5.40	proc_GetSizeOfWebPartsOnPage.....	65
3.1.5.40.1	Webparts Size Result Set	66
3.1.5.40.2	AllFileFragmentsBlob Size Result Set.....	66
3.1.5.41	proc_GetSPSiteGuidsGivenHostHeaderPattern	67
3.1.5.41.1	Site Guids With Host Header Result Set.....	67
3.1.5.42	proc_GetSPSiteGuidsGivenIdentity.....	67
3.1.5.42.1	Site Guids With Identity Result Set	67
3.1.5.43	proc_GetSPSiteGuidsGivenLockState	68
3.1.5.43.1	Site Guids With Lock State Result Set	68
3.1.5.44	proc_GetSPSiteGuidsGivenOwner.....	68
3.1.5.44.1	Site Guids Given Owner Result Set	69
3.1.5.45	proc_GetSPSiteGuidsGivenSecondaryOwner	69
3.1.5.45.1	Site Guids Given Secondary Owner Result Set.....	70
3.1.5.46	proc_GetSPWebIdentifiersGivenTitle.....	71

3.1.5.46.1	Get SPWeb Identifiers Given Title Result Set.....	71
3.1.5.47	proc_GetTimerLock	71
3.1.5.48	proc_GetTotalDiscussionsSize.....	72
3.1.5.48.1	Discussions Size Result Set.....	73
3.1.5.49	proc_GetUniqueScopesInWeb.....	73
3.1.5.49.1	Unique Scopes under Site Result Set	74
3.1.5.50	proc_GetUserStorageInfo.....	74
3.1.5.50.1	User Storage Info Result Set.....	75
3.1.5.51	proc_GetWebBestMatch	75
3.1.5.52	proc_GetWebSubset.....	75
3.1.5.52.1	Get Web Subset Result Set	76
3.1.5.53	proc_MakeExceptionForThrottle	76
3.1.5.54	proc_MoveRecycleBinItemToSecondStage	77
3.1.5.55	proc_GetStorageMetrics.....	79
3.1.5.55.1	Individual URL Security Result Set.....	80
3.1.5.55.2	Storage Metrics Result Set.....	80
3.1.5.55.3	Title Result Set	81
3.1.5.56	proc_ProcessStorageMetricsChanges	81
3.1.5.57	proc_QMChangeSiteDiskUsedAndContentTimestamp	82
3.1.5.58	proc_QMGetDiskWarning	82
3.1.5.58.1	Disk Warning Result Set.....	83
3.1.5.59	proc_QMMarkDiskWarning	83
3.1.5.60	proc_RestoreRecycleBinItem	83
3.1.5.61	proc_RevertDocContentStreams	86
3.1.5.62	proc_ScorchList	87
3.1.5.63	proc_ScorchWeb.....	87
3.1.5.63.1	Audit Mask Result Set	88
3.1.5.64	proc_SecBackupAllWebMembers	88
3.1.5.64.1	UserInfo Result Set.....	88
3.1.5.65	proc_SecGetListItemSecurity.....	89
3.1.5.65.1	Access Control List Result Set	89
3.1.5.66	proc_SecRemoveExternalSecurityProvider.....	90
3.1.5.67	proc_SetAuditMask	90
3.1.5.68	proc_SetDeadWebNotificationCount	91
3.1.5.69	proc_SetListRequestAccess	91
3.1.5.70	proc_SetSiteQuota	92
3.1.5.71	proc_SetSubscription	93
3.1.5.72	proc_SiteCollectionExists	93
3.1.5.73	proc_SizeOfPersonalizationsPerUser	94
3.1.5.73.1	User Storage Info Result Set.....	94
3.1.5.74	proc_TrimAuditEntries	94
3.1.5.75	proc_UpdateDiskUsed.....	95
3.1.5.76	proc_UpdateStatistics.....	95
3.1.5.77	proc_GetDatabaseInformation	95
3.1.5.77.1	Database Information Result Set	96
3.1.5.78	proc_SetDatabaseInformation	96
3.1.6	Timer Events	96
3.1.7	Other Local Events	96
3.2	Front-end Web Server Client Details.....	96
3.2.1	Abstract Data Model	96
3.2.2	Timers	97
3.2.3	Initialization	97
3.2.4	Message Processing Events and Sequencing Rules.....	97

3.2.5	Timer Events	97
3.2.6	Other Local Events	98
4	Protocol Examples	99
4.1	Auditing Operations	99
4.2	Quota Management Operations	99
4.2.1	Querying Quota.....	99
4.2.2	Updating Quota.....	100
4.2.2.1	Setting Quota from a Quota Template	100
4.2.3	Get Usage Information for a Site Collection	101
4.2.4	Warning Site Collections Which Are Near the Allowed Disk Space	101
4.3	Recycle Bin Operations.....	102
4.3.1	Query Items in First-Stage Recycle Bin	102
4.3.2	Delete a First-Stage Recycle Bin Item to Second-Stage Recycle Bin	103
4.3.3	Restore a First-Stage Recycle Bin Item	103
4.3.4	Delete a Second-Stage Recycle Bin Item	104
4.4	Security Operations	104
4.4.1	Remove External Security Provider.....	104
4.4.2	Get the ACL of a Specific SPLListItem	105
4.4.3	Retrieve All Site Members.....	106
4.5	Database Integrity and Maintenance Operations	106
4.5.1	Find Orphaned Objects for Repair	106
4.6	Query Operations	108
4.6.1	Filtered Query Operations by Owner	108
4.6.2	Filtered Query Operations by Best Match	108
4.6.3	Pages Query Operations.....	109
5	Security.....	110
5.1	Security Considerations for Implementers.....	110
5.2	Index of Security Parameters	110
6	Appendix A: Product Behavior.....	111
7	Change Tracking.....	112
8	Index	113

1 Introduction

The Windows SharePoint Services Content Database Administrative Communications protocol specifies the communication sequences used by front-end Web servers and application servers to perform administrative operations on a back-end database server related to content databases. This includes quota, recycle bin, security, database integrity and maintenance, auditing and query operations.

1.1 Glossary

The following terms are defined in [\[MS-GLOS\]](#):

- access control list (ACL)**
- anonymous user**
- ASCII**
- Coordinated Universal Time (UTC)**
- language code identifier (LCID)**

The following terms are defined in [\[MS-OFCGLOS\]](#):

- absolute URL**
- Active Directory account creation mode**
- All Site Members**
- ancestor**
- attachment**
- audit entry**
- audit event**
- audit log**
- author**
- back-end database server**
- configuration database**
- content database**
- content database lock**
- content type**
- current user**
- current version**
- datetime**
- delete transaction**
- delete transaction identifier**
- deleted**
- directory name**
- display name**
- document**
- document identifier**
- document library**
- document store type**
- document version**
- domain account mode**
- e-mail address**
- empty GUID**
- external security provider**
- farm**
- field**
- file**
- file fragment**

**first-stage Recycle Bin
folder
front-end Web server
full URL
group
host header
item
item identifier
leaf name
list
list identifier
list item
list schema
list server template
list template identifier
locked
login name
notify count
object model
parent list
parent site
permission
permission level
publishing level
quota template identifier
quota warning
quota warning level
Recycle Bin
Recycle Bin item
relationship lookup field
request identifier
restrict behavior
result set
return code
role
root folder
row
scope identifier
second-stage Recycle Bin
securable object
security group
security principal
security role
security scope
setup path
site
site certification
site collection
site collection administrator
site collection identifier
site collection quota
site identifier
site subscription
site template
stored procedure**

store-relative form
store-relative URL
subsite
survey list
template
top-level site
transaction application lock
Transact-Structured Query Language (T-SQL)
uncustomized
Uniform Resource Locator (URL)
user identifier
Web Part
Web Part Page
workflow
write lock
zero-based index

The following terms are specific to this document:

list flag: An 8-byte unsigned integer bit mask that provides metadata about a SharePoint list.

orphaned object: A content database object that lacks a requisite relationship to a corresponding object.

quota template: A group of default quotas that can be applied to a site collection. It is stored in the configuration database.

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

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

[MS-WSSCCSP2] Microsoft Corporation, "[Windows SharePoint Services Content Database Core List Schema and Site Provisioning Communications Version 2 Protocol Specification](#)"

[MS-WSSDLIM2] Microsoft Corporation, "[Windows SharePoint Services: Content Database Document and List Item Management Communications Version 2 Protocol Specification](#)"

[MS-WSSFO2] Microsoft Corporation, "[Windows SharePoint Services \(WSS\): File Operations Database Communications Version 2 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 specifies the communication between the **front-end Web server** and the **back-end database server** used to satisfy requests involving management and administration of **content databases**.

Content Database Administrative Communications Protocol is composed of six major operation areas as follows.

1.3.1 Auditing Operations

A common business requirement is that a protocol server be able to provide a recorded history of operations performed on various objects stored on the protocol server, such as when and by whom an object was viewed or modified. This recorded history can be used for the purposes of forensic monitoring ("auditing") to verify conformance of the operations performed on the objects to business requirements. This recorded history of **audit entries** is generally persisted during and potentially after the end-of-life of the objects, and as such is generally stored in an **audit log**.

The auditing operations allow protocol clients to determine and configure settings on objects that specify which operations performed on those objects are to be recorded. The protocol further provides methods for protocol clients to record, retrieve, and trim audit entries in an audit log.

1.3.2 Quota Management Operations

Quota management allows the administrators to set a quota for a **site collection**, create **quota templates** for use on many site collections, and get a list of site collections that are near their quota limits. A **site collection quota** is set to block any updates of existing content or additions of new content or users to a site collection that has reached its quota limit.

1.3.2.1 Query and Update Quota Operations

The back-end database server provides methods for the client to query and update quota information for site collections. Quota information consists of data about disk space allowed in Bytes, and the number of users allowed per site collection. When client requests for quota information are sent to the front-end Web server, the front-end Web server sends a series of **stored procedure** calls to the back-end database server for the requested information. The stored procedures return data which in turn can be used for further calls to other stored procedures. The front-end Web server turns the values in the **return codes** and **result sets** of the stored procedures into objects which contain the data requested by the client and uses the objects according to implementation-specific procedures.

1.3.2.2 Query and Update Usage Operations

The back-end database server provides methods for the client to query and update usage information for a site collection. Usage information consists of data about actual disk space used, in Bytes, by various types of content in a site collection including **documents**, **document libraries**, **lists**, and the **Recycle Bin**. When client requests for usage information are sent to the front-end Web server, the front-end Web server sends a series of stored procedure calls to the back-end database server for the requested information. The stored procedures return data which in turn can be used for further calls to other stored procedures. The front-end Web server turns the values in the return codes and result sets of the stored procedures into objects which contain the data requested by the client and uses the objects according to implementation-specific procedures.

1.3.2.3 Query Warn Operations

The back-end database server provides methods for the client to query site collections whose actual disk space used has crossed the warning limits set in the quota. The front-end Web server sends a series of stored procedure calls to the back-end database server to get a list of site collections which have crossed the warning limits for actual disk space used. The stored procedures return data which in turn can be used for further calls to other stored procedures. The front-end Web server turns the values in the return codes and result sets of the stored procedures into objects which contain the data requested by the client and uses the objects according to implementation-specific procedures to send notifications to the **site collection administrators** of those site collections. Next, the front-end Web server sends a series of stored procedure calls to mark the notified site collections as already been notified.

1.3.3 Recycle Bin Operations

Recycle Bin Operations allow protocol clients to perform maintenance activities on the Recycle Bins such as querying, deleting and restoring their contents.

1.3.3.1 Query Operations

The back-end database server provides methods to query for all **Recycle Bin items** in the first-stage Recycle Bin for a specific user in a specific **site** or for all Recycle Bin items in the second-stage Recycle Bin in a site collection when the user is a site collection administrator. The front-end Web server sends a series of stored procedure calls to the back-end database server for the requested information. The stored procedures return data which in turn are used for further calls to other stored procedures. The front-end Web server turns the values in the return codes and result sets of the stored procedures into objects which contain the data requested by the client and uses the objects according to implementation-specific procedures.

1.3.3.2 Administration Operations

There are two sets of administration operations. The first set is used to move items from a site collection through the different Recycle bins. The second set is used to restore items from the Recycle Bins back to their original location in the site collection.

1.3.3.2.1 Delete Operations

The back-end database server provides methods to delete items from a site collection. These operations move the **deleted** items into the first-stage Recycle Bin. The back-end database server also provides methods to delete items from the Recycle Bins. Deleting items in the first-stage Recycle Bin moves the items to the second-stage Recycle Bin. Deleting items in the second-stage Recycle Bin removes them in a way that they no longer exist on disk on the back-end database server.

Each individual **delete transaction** is identified by a **delete transaction identifier**, which is unique in a content database. All items in the same delete transaction are assigned the same delete transaction identifier value.

The front-end Web server sends a series of stored procedure calls to the back-end database server to delete items from a site collection or the Recycle Bins. The stored procedures return data which in turn are used for further calls to other stored procedures. The front-end Web server turns the values in the return codes and result sets of the stored procedures into objects which contain the data requested by the client and uses the objects according to implementation-specific procedures.

1.3.3.2.2 Restore Operations

The Recycle Bin allows users and site collection administrators to restore **items** back to their original location. The **first-stage Recycle Bin** allows users to restore their items, and the **second-stage Recycle Bin** allows site collection administrators in case the items can no longer be restored from the first-stage Recycle Bin. The back-end database server provides methods to restore all Recycle Bin items from a single delete transaction. Items are restored from the first-stage Recycle Bin or from the **second-stage Recycle Bin** to their original locations in the site collection. Items in the **second-stage Recycle Bin** can only be restored by a site collection administrator.

The front-end Web server sends a series of stored procedure calls to the back-end database server to restore items from the Recycle Bins. The stored procedures return data which in turn are used for further calls to other stored procedures. The front-end Web server turns the values in the return codes and result sets of the stored procedures into objects which contain the data requested by the client and uses the objects according to implementation-specific procedures.

1.3.4 Security Operations

Security operations satisfy requests involving file access and administration of users and **Groups** within the system.

1.3.4.1 Operations Related to External Security Provider

This protocol allows **permissions** on a site to be enforced by an **external security provider**. The external security provider can also be removed to allow the client manage permissions directly.

1.3.4.2 Operations Related to ACL

This protocol provides methods for retrieving information about **access control lists (ACLs)** and Windows SharePoint Services Rights Masks, as defined in [\[MS-WSSFO2\]](#) section 2.2.3.14, for **anonymous users** on **securable objects** such as sites, lists, and **list items**.

1.3.4.3 User and Group Operations

This protocol provides methods for retrieving information about individual users and groups. When the **object model** on the front-end Web server operates on requests to query or update users or groups, the front-end Web server confirms if the data is already populated in the local objects that represent the specific user or group, and if it does not exist, it sends a series of stored procedure calls to the back-end database server for the requested information. The stored procedures return data which in turn are used for further calls to other stored procedures. The front-end Web server turns the values in the return codes and result sets of the stored procedures into objects that contain the data and metadata for the requested users or groups and uses the objects according to implementation specific procedures.

1.3.5 Database Integrity and Maintenance Operations

The database integrity and maintenance operations allow protocol clients to perform maintenance activities on the content database such as detecting and fixing **orphaned objects**.

1.3.5.1 Orphaned Objects Management

The back-end database server provides methods for the client to look for objects in the content database that are orphaned and, optionally, fix them.

1.3.5.2 Dead Web Management

The back-end database server provides methods for the client to enumerate through **top-level sites** and collect information such as the number of days that have elapsed between the last **site certification** and the current date. This information can be used to determine which site collections have been inactive so that they can be deleted if desired.

1.3.5.3 Maintenance Operations

The back-end database server provides methods for dealing with database integrity issues such as deleting a corrupted list, a list with no **parent site**, or a list with items without a **parent list**.

1.3.6 Query Operations

The back-end database server provides methods for obtaining a list of site collections and sites that match specific criteria.

The back-end database server also provides methods for obtaining subsets of site collections, sites and lists. These methods can be used by a front-end Web server to obtain a list of objects in smaller subsets.

1.3.6.1 Filtered Query Operations

These methods can be used by the front-end Web server to identify subsets of site collections and sites to perform operations on or to provide users of the front-end Web server with a list of objects stored in the back-end database server filtered by the criteria provided by them.

Examples of the criteria supported for filtering are **site identifier**, **host header**, lock state, name of the owner or secondary owner and title.

1.3.6.2 Paged Query Operations

These methods can be used by the front-end Web server to provide users of the front-end Web server with a subset of all the site collections, sites and lists stored in the back-end database server. This can be used, for example, in a paged display model, where a subset of the site collections, sites and lists are displayed on each page of a display and the user can scroll or page between multiple pages of information.

1.4 Relationship to Other Protocols

This client-to-server protocol uses the Tabular Data Stream Protocol, as described in [\[MS-TDS\]](#), as its transport between the front-end Web server, acting as a client, and the back-end database server, acting as a server.

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

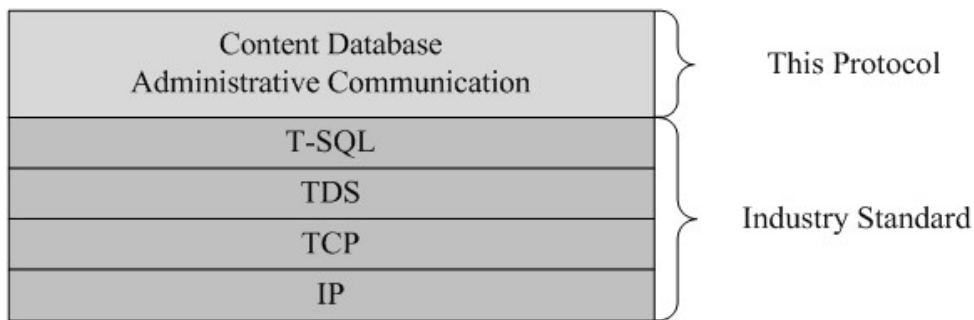


Figure 1: This protocol in relation to other protocols

1.5 Prerequisites/Preconditions

The operations described by this protocol operate between a client and a back-end database server. The client is expected to know the location and connection information for the databases.

This protocol requires that the protocol client has appropriate permissions to call the stored procedures stored on the back-end database server.

1.6 Applicability Statement

This protocol is intended for use by protocol clients and protocol servers that are both connected by high-bandwidth, low latency network connections.

1.7 Versioning and Capability Negotiation

- **Security and Authentication Methods:** This protocol supports the SSPI and SQL Authentication with the Protocol Server role described in [\[MS-TDS\]](#).

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

The Tabular Data Stream Protocol [\[MS-TDS\]](#) is the transport protocol used to call the stored procedures, query SQL views or SQL tables, return result sets and return codes.

2.2 Common Data Types

The following sections define the common data types that are used in this protocol.

2.2.1 Simple Data Types and Enumerations

None.

2.2.2 Simple Data Types

2.2.2.1 Audit Event Source

A 1-byte unsigned integer enumeration specifying the source of the audit entry. This MUST be a value specified as follows:

Value	Meaning
0x00	The audit entry is generated by the server code internally.
0x01	The audit entry is generated by object model code.

2.2.2.2 Audit Event Type

A 4-byte unsigned integer enumeration specifying the type of operation that generated the **audit event**. This MUST be a value specified as follows:

Value	Meaning
0x00000001	The audit event was generated when a document was checked out.
0x00000002	The audit event was generated when a document was checked in.
0x00000003	The audit event was generated when an object was viewed.
0x00000004	The audit event was generated when an object was deleted.
0x00000005	The audit event was generated when an object was updated.
0x00000006	The audit event was generated when a content type was updated.
0x00000007	The audit event was generated when a child object was deleted.
0x00000008	The audit event was generated when a list schema changed.
0x0000000A	The audit event was generated when an object was undeleted.
0x0000000B	The audit event was generated by workflow .
0x0000000C	The audit event was generated when an object was copied.

Value	Meaning
0x0000000D	The audit event was generated when an object was moved.
0x0000000E	The audit event was generated when the audit flags, as specified in [MS-WSSFO2] , Section 2.2.3.1 , of an object were updated.
0x0000000F	The audit event was generated when a search operation was performed.
0x00000010	The audit event was generated when a child object was moved.
0x00000011	The audit event was generated when a file fragment was written for the object.
0x0000001E	The audit event was generated when a security group was created.
0x0000001F	The audit event was generated when a security group was deleted.
0x00000020	The audit event was generated when a security principal was added to a security group.
0x00000021	The audit event was generated when a security principal was removed from a security group.
0x00000022	The audit event was generated when a security role was created.
0x00000023	The audit event was generated when a security role was deleted.
0x00000024	The audit event was generated when a security role was updated.
0x00000025	The audit event was generated when a security role breaks inheritance.
0x00000026	The audit event was generated when a security scope was updated.
0x00000027	The audit event was generated when a security scope restores inheritance.
0x00000028	The audit event was generated when a security scope breaks inheritance.
0x00000032	The audit event was generated when audit events were deleted.
0x00000064	The audit event was generated by a custom operation.

2.2.2.3 Delete Item Type

A 1-byte signed integer value indicating the type of the Recycle Bin item. It MUST be one of the following values:

Value	Description
1	The item is a document.
2	The item is a document version of a document.
3	The item is a list item.
4	The item is a list.
5	The item is a folder .
6	The item is a folder which contains lists.
7	The item is an attachment .

Value	Description
8	The item is a version of a list item.
9	The item is the parent item in a cascading delete operation.

2.2.2.4 Recycle Bin Stage

A 1-byte signed integer value indicating the stage of the Recycle Bin. It MUST be one of the following values:

Value	Description
1	First-stage Recycle Bin
2	Second-stage Recycle Bin

2.2.3 Bit Fields and Flag Structures

None.

2.2.4 Enumerations

None.

2.2.5 Binary Structures

None.

2.2.6 Common Result Sets

The following common result sets are used by this protocol.

2.2.6.1 Site Collection with No Sites Result Set

The Site Collection with no Sites result set returns site collections without sites. The Site Collection with no Sites result set MUST return a number of rows equal to the number of site collections without sites. If there are no such site collections, it returns zero rows. The T-SQL (Transact-Structured Query Language) syntax for the result set is as follows:

```
Id    uniqueidentifier;
```

Id: The **site collection identifier** of the site collection with no sites. **Id** MUST NOT be NULL.

2.2.6.2 Sites with No Site Collection Result Set

The Sites with no Site Collection result set returns sites with no site collection and sites with no **root folder**. The Sites with no Site Collection result set MUST return a number of rows equal to the number of sites with no site collection plus the number of sites with no root folder. If there are no such sites, it returns zero rows. The **T-SQL** syntax for the result set is as follows:

```
Id          uniqueidentifier,
Title       nvarchar(255),
```

```
SiteId      uniqueidentifier,  
FullUrl     nvarchar(256);
```

Id: The site identifier of the site with no site collection or the site with no root folder. **Id** MUST NOT be NULL.

Title: The **display name** of the site.

SiteId: The site collection identifier of the site collection which contains the site. **SiteId** MUST be NULL for sites with no site collection and MUST NOT be NULL otherwise.

FullUrl: The **store-relative form** of the site. **FullUrl** MUST NOT be NULL.

2.2.6.3 Sites with No Parent Site Result Set

The Sites with no Parent Site result set returns those sites that have no parent site. The Sites with no Parent Site result set MUST return a number of rows equal to the number of sites with no parent site. If there are no such sites, it returns zero rows. The T-SQL syntax for the result set is as follows:

```
Id          uniqueidentifier,  
Title       nvarchar(255),  
SiteId      uniqueidentifier,  
FullUrl     nvarchar(256);
```

Id: The site identifier of the site with no parent site. **Id** MUST NOT be NULL.

Title: The display name of the site.

SiteId: The site collection identifier of the site collection which contains the site. **SiteId** MUST NOT be NULL.

FullUrl: The store-relative form of the site. **FullUrl** MUST NOT be NULL.

2.2.6.4 Folders with No Site Result Set

The Folders with no Site result set returns orphaned folders at the root of the site that have no associated site. The Folders with no Site result set MUST return a number of rows equal to the number of folders with no site. If there are no such folders, it returns zero rows. The T-SQL syntax for the result set is as follows:

```
WebId       uniqueidentifier,  
SiteId      uniqueidentifier;
```

WebId: The site identifier of the site which contains the orphaned folder. **WebId** MUST NOT be NULL.

SiteId: The site collection identifier of the site collection which contains the orphaned folder. **SiteId** MUST NOT be NULL.

2.2.6.5 Orphaned Lists Result Set

The Orphaned Lists result set returns orphaned lists in the following three categories:

- Lists with no parent site
- Lists with documents that have no parent list
- Lists with items that have no parent list

The Orphaned Lists result set MUST return a result set having a number of rows equal to the number of orphaned lists. If there are no such lists, the result set returns zero rows. The T-SQL syntax for the result set is as follows:

```
ListId      uniqueidentifier,
Title       nvarchar(255),
WebId       uniqueidentifier,
SiteId      uniqueidentifier;
```

ListId: The **list identifier** of an orphaned list.

Title: The display name of the list. **Title** MUST be NULL if the result is in either "Lists with documents with no Parent List" or "Lists with items with no Parent List" categories.

WebId: The site identifier of the site which contains the list. **WebId** MUST be NULL if the result is in either "Lists with documents with no Parent List" or "Lists with items with no Parent List" categories.

SiteId: The site collection identifier of the site collection that contains the list. **SiteId** MUST be NULL if the result is in "Lists with no Parent Site" category.

2.2.6.6 User Storage Info Result Set

The User Storage Info result set returns the **user identifier**, **login name**, and the total size, in bytes, of the personalizations and **Web Parts** on a particular **Web Part Page**. There is one row returned for each user that customizes the Web Part Page.

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

```
tp_Id      int,
tp_Login    nvarchar(255),
Size        bigint;
```

tp_Id: The user identifier.

tp_Login: The login name of the user.

Size: The size, in bytes, of both the personalization and Web Parts on a particular Web Part Page.

2.2.7 SQL Structures

None.

2.2.8 Tables and Views

This section describes the tables and views used in this protocol.

2.2.8.1 RecycleBin Table

The RecycleBin table stores the descriptions and properties of items in the Recycle Bin for all site collections in the current database. The table is defined using T-SQL syntax as follows:

```
TABLE [dbo].[RecycleBin]
(
    SiteId                uniqueidentifier NOT NULL,
    WebId                 uniqueidentifier NOT NULL,
    BinId                 tinyint NOT NULL,
    DeleteUserId          int NOT NULL,
    DeleteTransactionId   varbinary(16) NOT NULL,
    DeleteDate            datetime NOT NULL,
    ItemType              tinyint NOT NULL,
    ListId               uniqueidentifier NULL,
    DocId                uniqueidentifier NULL,
    DocVersionId         int NULL,
    ListItemId           int NULL,
    Title                nvarchar(260) NOT NULL,
    DirName              nvarchar(256) NOT NULL,
    LeafName            nvarchar(128) NOT NULL,
    AuthorId             int NULL,
    Size                bigint NOT NULL,
    ListDirName          nvarchar(256) NULL,
    ScopeId              uniqueidentifier NULL,
    ProgId               nvarchar(255) NULL,
    ChildDeleteTransactionId varbinary(16) NOT NULL DEFAULT 0x,
    OriginalItemType      tinyint NULL,
    EffectiveDeleteTransactionId AS CASE
        WHEN ChildDeleteTransactionId = 0x
        THEN DeleteTransactionId
        ELSE ChildDeleteTransactionId
    END PERSISTED
)
```

SiteId: The site collection identifier of the site collection in which this item belongs.

WebId: The site identifier of the site in which this item belongs.

BinId: The [Recycle Bin stage](#) for this item.

DeleteUserId: The user identifier of the user who placed the specified item in the Recycle Bin.

DeleteTransactionId: The delete transaction identifier of the delete transaction to which this item belongs.

DeleteDate: The date on which this item was placed in the Recycle Bin.

ItemType: The type of the Recycle Bin item. **ItemType** MUST NOT be NULL and MUST be one of the values in [Delete Item Type](#).

ListId: The list identifier of the list to which this item belongs.

DocId: The **document identifier** of the document that corresponds to this item. DocId MUST be NULL if **ItemType** is either 4 or 8.

DocVersionId: The document version of the document that corresponds to this item.

DocVersionId MUST be NULL for the following values of ItemType: 1, 3, 4, 5, 6, or 7.

ListItemId: The identifier of the list item that corresponds to this item. **ListItemId** MUST be NULL for the following values of **ItemType**: 2, 4, or 6.

Title: The display name of the item.

DirName: The **directory name** of the document which corresponds to this item.

LeafName: The leaf name of the document which corresponds to this item.

AuthorId: The user identifier of the user who originally created the item. If the value of **ItemType** is 7, **AuthorId** value MUST be NULL.

Size: The size of the item in Bytes.

ListDirName: The directory name of the list to which this item belongs. If the value of **ItemType** is 4, **ListDirName** value MUST be NULL.

ScopeId: The **scope identifier** of the security scope of the document that corresponds to this item.

ProgId: The file type which specifies the preferred application used to open the document that corresponds to this item.

ChildDeleteTransactionId: If the deleted item was a child item of a cascading delete operation, it specifies the delete transaction identifier of the delete transaction for deleting the individual item. If the deleted item was not a child item of a cascading delete operation, it MUST be 0x.

OriginalItemType: If the deleted item was a parent item of a cascading delete operation, it specifies the original type of the Recycle Bin item and MUST be one of the values 1,2,3,4,5,6,7 or 8 in Delete Item Type. If the deleted item was not a parent item of a cascading delete operation, it MUST be NULL.

EffectiveDeleteTransactionId: It MUST be equal to the **DeleteTransactionId** value if **ChildDeleteTransactionId** is equal to 0x, otherwise it MUST be equal to the **ChildDeleteTransactionId**.

2.2.9 XML Structures

None.

3 Protocol Details

This section provides detailed information about front-end and back-end server processes for this protocol.

3.1 Back-End Database Server Details

This section provides detailed information about the back-end database server.

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.

The back-end database server maintains the following sets of data for this protocol within both a **configuration database** and one or more content databases. Data within the appropriate databases are maintained until updated or removed.

Documents: A set of information about all documents in a content database. Document entries are identified by document identifiers, and are also represented by store-relative form.

Lists: A set of information about all lists in a content database. List entries are identified by list identifiers, and are also represented by store-relative form.

List Items: A set of information about all list items in a content database. List item entries are identified by list item identifiers.

Quota Templates: Objects that represent the quota limits that can be set on a site collection. Quota templates are stored in the configuration database.

Roles: A set of information about all **roles** in a content database. Role entries are identified by role identifiers.

Site Collections: A set of information about all site collections in a content database. Site collection entries are identified by site collection identifiers and are also represented by either **absolute URLs** or store-relative form.

Permission levels: A set of information about all **permission levels** in a content database. Permission level entries are identified by permission level identifiers.

Sites: A set of information about all sites in a content database. Site entries are identified by site identifiers and are also represented by store-relative form.

Users: A set of information about all users in a content database. User entries are identified by user identifiers.

Versions: A set of information indicating the **current version** information for various components in the **farm**.

The following stored procedures have multiple uses:

fn_CompareTZTransitionDate

fn_EscapeForLike

fn_GetRootFolder
fn_HtmlEncode
fn_LocalDayFromUTCDate
proc_ConvertStringToDate
proc_DefragmentIndices
proc_GetDatabaseInformation
proc_SetDatabaseInformation
proc_DTSetRelationship
proc_GetAllSPWebIdentifiersGivenSiteGuid
proc_GetCustomizedDocumentsInWeb
proc_GetFirstUniqueAncestorWebUrl
proc_GetListBestMatch
proc_GetSitecollectionBestMatch
proc_GetSiteCollectionSubset
proc_GetSPSiteGuidsGivenHostHeaderPattern
proc_GetSPSiteGuidsGivenIdentity
proc_GetSPSiteGuidsGivenLockState
proc_GetSPSiteGuidsGivenOwner
proc_GetSPSiteGuidsGivenSecondaryOwner
proc_GetSPWebIdentifiersGivenTitle
proc_GetTimerLock
proc_GetUniqueScopesInWeb
proc_GetWebBestMatch
proc_GetWebSubset
proc_MakeExceptionForThrottle
proc_SetListRequestAccess
proc_SetSubscription
proc_UpdateStatistics

3.1.1.1 Audit Operations

The protocol server stores a hierarchy of objects. Operations that can be performed against those objects are divided into categories (for example updates, deletes, copies), and the protocol server maintains three sets of bit masks, as specified in [MS-WSSFO2], Section 2.2.3.1, that specify which categories of operations will be recorded ("audited"). An audit entry is recorded for an operation performed against the object if the corresponding category is set in any of the three sets of audit flags as specified in [MS-WSSFO2], Section 2.2.3.1. The three sets of audit flags are as follows:

1. **Direct Audit Flags:** These audit flags indicate operations that are audited when performed directly on an object.
2. **Inherited Audit Flags:** These audit flags indicate operations that are audited for an object because the direct audit flags being set on an object contained within the object. In this manner, the inherited audit flags indicate that an audit entry is recorded for operations that indirectly affect an object even if the action is not performed directly on that object. For example, if the direct audit flags on a document in a folder specify that the "Delete" event will be audited on the document, then the inherited audit flags of the folder that contains the document will indicate that the "Delete" event will be audited for the folder.
3. **Global Audit Flags:** These audit flags indicate operations that are audited for all objects.

The protocol server stores the list of audit entries to an "Audit Log".

Audit operations use the following stored procedures:

```
proc_AddAuditEntry  
proc_AddAuditEntryUrl  
proc_CountAuditEntries  
proc_GetAuditEntries  
proc_SetAuditMask  
proc_TrimAuditEntries
```

3.1.1.2 Quota Management Operations

Data is stored in hierarchical objects on the protocol server. A site collection is at the root of the logical partitioning of this hierarchy. Site collections can contain sites, which in turn can contain lists and items. A site collection can be considered as an independent unit and can be managed for security and size limits. Quota management allows for setting the maximum allowed disk space on the protocol server for a site collection and the maximum number of users allowed in a site collection if **Active Directory account creation mode** is used. The actual disk space used refers to the total size of all the content such as documents, lists, and list items in a site collection stored on the protocol server. Quota management also allows for setting warning limits for actual disk space used and the number of users in a site collection. Protocol clients can use the warning limits to warn the site collection administrators that their site crossed the warning limits but is still below the maximum allowed limits.

Protocol clients can set the maximum allowed and the warning limits on the actual disk space used and the number of users in a site collection using a call to the stored procedure `proc_SetSiteQuota`. Protocol clients can request the quota information for a site collection using the stored procedure `proc_GetSiteQuota`.

Protocol clients can get a list of site collections that have crossed the warning limits set in the quota using the stored procedure `proc_QMGetDiskWarning` and can send out notifications to the site collection administrators of those site collections from the result set of the stored procedure. A call to the stored procedure `proc_QMGetDiskWarning` MUST be followed by a call to the stored procedure `proc_QMMarkDiskWarning`, which indicates that the site collections have been warned so that those site collections are not returned in the next call to `proc_QMGetDiskWarning` if there are no further changes in them.

The actual disk space used by a site collection is updated when stored procedures such as `proc_AddDocument` and `proc_AddListItem`, as specified in [\[MS-WSSFO2\]](#), Sections 3.1.4.3 and 3.1.4.4, are used to add documents or list items to the site collection. The actual disk space used by a site collection on the protocol server can also be updated using the stored procedure `proc_CalculateAndUpdateSiteDiskUsed`. Stored procedures such as `proc_AddDocument` and `proc_AddListItem` also prevent adding of content to the protocol server if the site collection to which data is being added has crossed the allowed disk space limits set by a quota.

The number of users in a site collection is updated when the stored procedures such as `proc_SecAddUser`, as specified in [\[MS-WSSFO2\]](#), Section 3.1.4.52, is used to add a user to a site collection. `proc_SecAddUser` also prevents adding users to a site collection if the number of users exceeds its quota limit set for the maximum number of users allowed.

Quota management operations use the following stored procedures:

- `fn_IsOverQuotaOrWriteLocked`
- `proc_CalculateAndUpdateSiteDiskUsed`
- `proc_GetSiteQuota`
- `proc_QMMarkDiskWarning`
- `proc_QMGetDiskWarning`
- `proc_QMChangeSiteDiskUsedAndContentTimestamp`
- `proc_SetSiteQuota`
- `proc_GetTotalDiscussionsSize`
- `proc_GetDocSizeInfo`
- `proc_SizeOfPersonalizationsPerUser`
- `proc_GetListSizes`
- `proc_GetSiteUsage`
- `proc_UpdateDiskUsed`
- `proc_GetDocLibrarySizes`
- `proc_GetSizeOfWebPartsOnPage`
- `proc_GetUserStorageInfo`

3.1.1.3 Recycle Bin Operations

3.1.1.3.1 Recycle Bin

The protocol server stores a collection of objects that are intended to be removed from a site collection in a logical container called a Recycle Bin, which is split into two parts. The first-stage Recycle Bin is visible to site collection administrators and Users, and the second-stage Recycle Bin is visible to site collection administrators.

A protocol client can move a Recycle Bin item from a site collection into the first-stage Recycle Bin or from the first-stage Recycle Bin to the second-stage Recycle Bin dependent on permission checks. A protocol client can also move an item from both of the Recycle Bins back to its original location in a site collection dependent on permission checks. A protocol client can also delete a Recycle Bin item from the second-stage Recycle Bin, which permanently deletes the data on the back-end database server physical disk.

The act of moving a Recycle Bin item into a Recycle Bin is called delete, and the inverse action is called restore. These actions are explained based on the diagrams in the following three subsections. In the diagrams, the first-stage Recycle Bin and the second-stage Recycle Bin are abbreviated as FSRB and SSRB, respectively.

Recycle bin operations use the following stored procedures:

- proc_ConfirmSiteUsage
- proc_DeleteRecycleBinItem
- proc_EnumRecycleBinItemsForCleanup
- proc_EnumRecycleBinToFreeSecondStageQuota
- proc_GetAdminRecycleBinInfo
- proc_GetAdminRecycleBinItems
- proc_GetRecycleBinItemInfo
- proc_GetRecycleBinItems
- proc_RestoreRecycleBinItem
- proc_MoveRecycleBinItemToSecondStage
- proc_ForceDeleteList
- proc_GetListSubset
- proc_EnumSitesForDeadWebCheck
- proc_DetectOrphans
- proc_DetectOrphansFix
- proc_ScorchList
- proc_ScorchWeb
- proc_GetDeadWebInfo

```

proc_SiteCollectionExists
proc_RevertDocContentStreams
proc_SetDeadWebNotificationCount

```

3.1.1.3.2 Query Operations

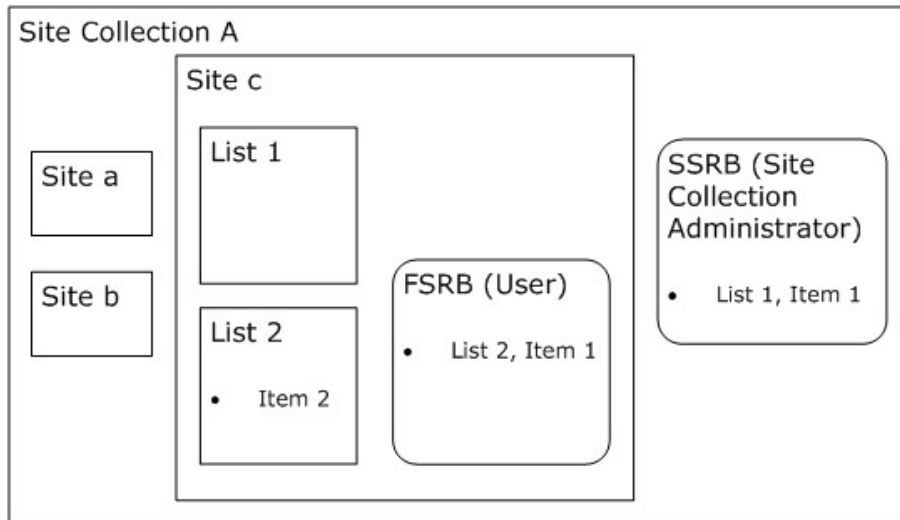


Figure 2: Query Operations on Recycle Bins

The protocol server can query the contents of the first-stage Recycle Bin if it has the appropriate permissions for Site c, which would return List 2, Item 1. The protocol server can query the contents of the second-stage Recycle Bin if it has site collection administrator permission on Site Collection A, which would return List 1, Item 1 and List 2, Item 1.

3.1.1.3.3 Delete Operations

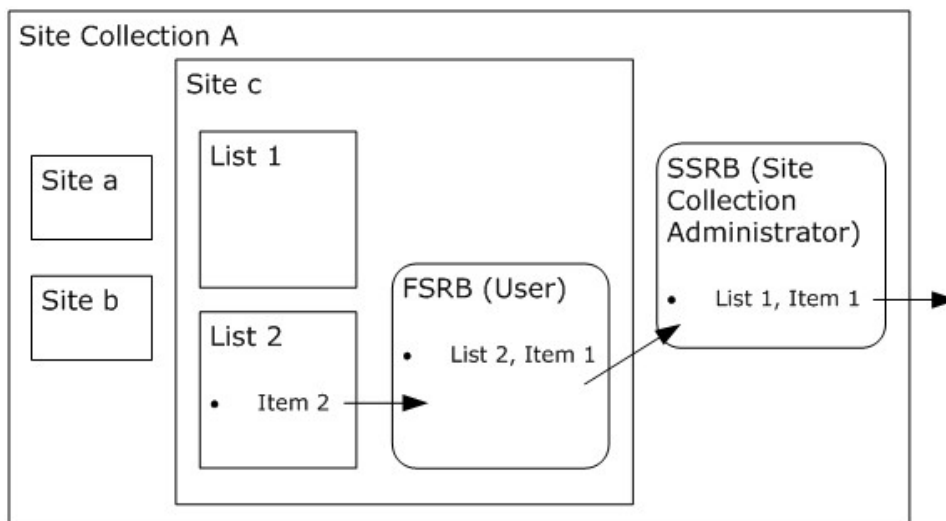


Figure 3: Delete Operations on Recycle Bins

The protocol server can delete List1, Item1 from the second-stage Recycle Bin and remove it from the back-end database server physical disk if it has site collection administrator permissions on Site Collection A. The protocol server can delete List 2, Item 1 from the first-stage Recycle Bin into the second-stage Recycle Bin if it has the appropriate permissions for Site c. The protocol server can delete List 2, Item 2 into the first-stage Recycle Bin if it has appropriate permission for Site c.

3.1.1.3.4 Restore Operations

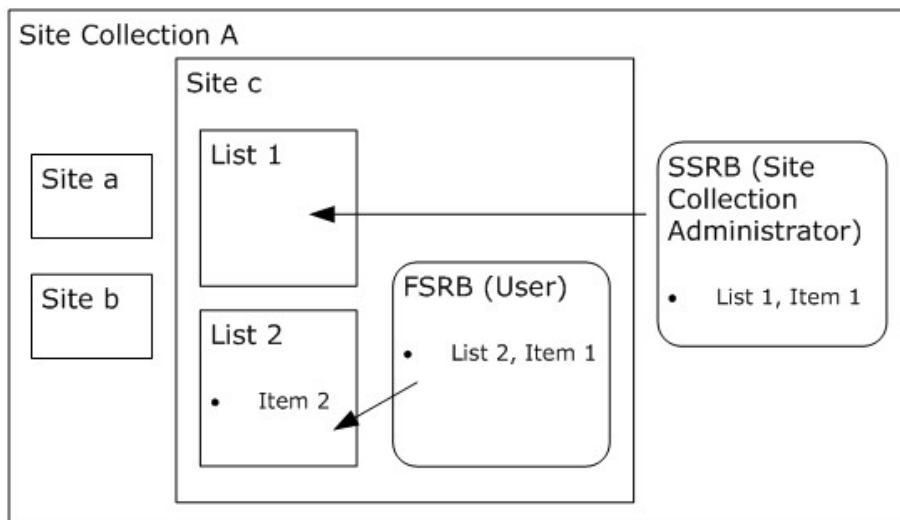


Figure 4: Restore Operations on Recycle Bins

The protocol server can restore List 2, Item1 from the first-stage Recycle Bin to its original location in List 2 if it has the appropriate permissions on Site c. The protocol server can restore List1, Item1 from the second-stage Recycle Bin to its original location in List 1 if it has the appropriate permissions on Site Collection A.

3.1.1.4 Security Operations

The protocol server stores a collection of sites with a globally unique identifier (GUID) indicating whether there is an external security provider associated with the site, and the scope identifier associated with the site.

Security operations use the following stored procedures:

proc_SecRemoveExternalSecurityProvider

proc_SecGetListItemSecurity

proc_SecBackupAllWebMembers

3.1.1.4.1 Operations Related to External Security Provider

When the protocol server is called to remove external security provider for a site, the GUID associated with that site is removed.



Figure 5: Removal of External Security Provider

3.1.1.4.2 Operations Related to ACL

The protocol server stores a collection of the binary serialization of access control list (ACL) that is indexed by the scope identifier.

The protocol server stores a collection of list items that are indexed by site identifier, list identifier, and **item identifier**. The scope identifier for every list item is also stored.

When the protocol server is called to retrieve access control lists (ACLs) about a specific list item, a data structure containing the list items is joined with a data structure containing access control lists (ACLs) and returns a result set that contains access control list (ACL) information for the list items.

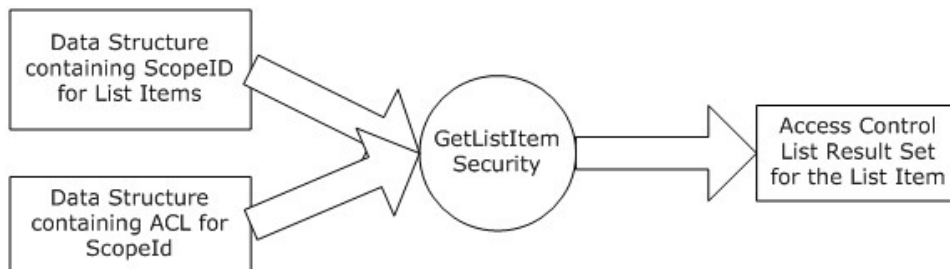


Figure 6: Retrieval of List Item Security

When the protocol server is called to retrieve access control list (ACL) for all the unique security scopes of subsites contained in a specified site, a data structure that contains scope identifiers for all the subsites is joined with a data structure that contains access control lists (ACLs) with the scope identifier and returns a result set that contains access control list (ACL) information for all the unique security scopes of the subsites.

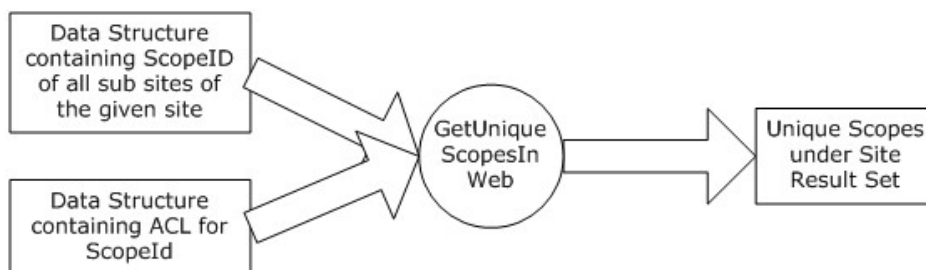


Figure 7: Retrieval of Unique Scopes for Subsites of a Specified Site

3.1.1.4.3 Operations Related to User and Group

The protocol server stores a data structure that contains user identifier of **All Site Members**, and a data structure that contains information about all users within the site collection.

When the protocol server is called to return information about All Site Members, the two data structures are joined to return a result set that contains user information for All Site Members.

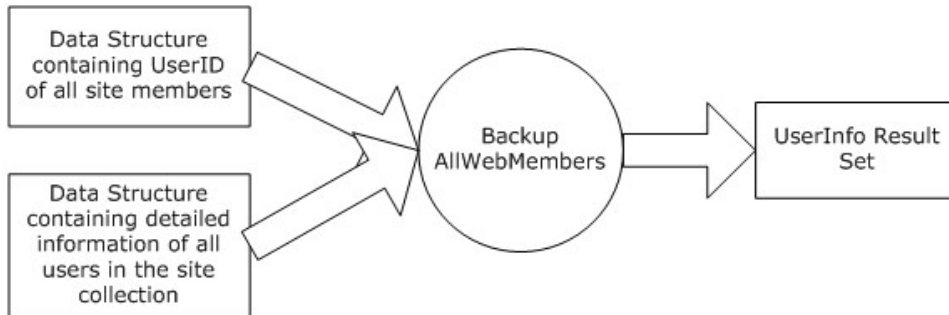


Figure 8: Retrieval of Information for All Site Members

3.1.2 Timers

An execution timeout timer on the protocol server governs the execution time for any requests. The amount of time is specified by a timeout value that is configured on the protocol server for all connections.

3.1.3 Initialization

A connection that uses the underlying protocol layers that are specified in [Relationship to Other Protocols](#) MUST be established before using this protocol as specified in [\[MS-TDS\]](#).

3.1.4 Higher-Layer Triggered Events

None

3.1.5 Message Processing Events and Sequencing Rules

3.1.5.1 fn_CompareTZTransitionDate

The fn_CompareTZTransitionDate function is called to compare a date and time value to a second date and time value described by its month, week, day of the week, and hour. The second date and time value has an implicit year equal to the first date and time value.

The T-SQL syntax for the function is as follows.

```
FUNCTION fn_CompareTZTransitionDate (
    @dtLocal      datetime,
    @_m           int,
    @_nwd         int,
    @_wd          int,
    @_h           int
)
RETURNS         bit;
```

@dtLocal: The date and time value to compare.

@_m: The month of the second date and time value.

@_nwd: The week of the month of the second date and time value.

@_wd: The weekday of the second date and time value.

@_h: The hour of the day of the second date and time value.

The values for the hour, weekday, week, and month parameters **MUST** be within the following ranges:

Parameter	Valid Range
Hour	0 – 23
Weekday	0 (Sunday) – 6 (Saturday)
Week	1 – 5, where 5 indicates the final occurrence during the month if that day of the week does not occur 5 times.
Month	1 – 12

Return Value: The function **MUST** return 1 if @dtLocal is greater than or equal to the second date and time value. Otherwise, it **MUST** return 0.

3.1.5.2 fn_EscapeForLike

The fn_EscapeForLike function is called to prepare a string for use in a T-SQL LIKE search pattern. The characters '%', '_', and '[' are used as wildcard characters in a LIKE search pattern. To use them as literal characters rather than wildcard characters, each of these wildcard characters **MUST** be surrounded by square brackets('[' and ']'). This process is known as escaping the string. The T-SQL syntax for the function is as follows.

```
FUNCTION dbo.fn_EscapeForLike (
    @Source          nvarchar(260),
    @AddTerminalWildcard bit = 1
)
RETURNS nvarchar(1024);
```

@Source: The string to be escaped.

@AddTerminalWildcard: A bit flag specifying whether a wildcard search pattern should be added to the escaped @Source string.

Return Value: If @Source is NULL, then fn_EscapeForLike **MUST** return NULL. Otherwise, fn_EscapeForLike **MUST** escape the @Source string by replacing wildcard characters according to the following table.

Character in Source	Escaped Character
%	[%]
_	[_]

Character in Source	Escaped Character
[[[

If @AddTerminalWildcard is set to 1 and @Source is not NULL, the literal string "/" MUST be appended to the escaped @Source string.

3.1.5.3 fn_GetRootFolder

The fn_GetRootFolder function is called to retrieve the **full URL** for the root folder of a list.

The T-SQL syntax for the function is as follows.

```
FUNCTION fn_GetRootFolder(
    @tp_RootFolder    uniqueidentifier
)
RETURNS              nvarchar(260);
```

@tp_RootFolder: The identifier of a root folder.

Return Value: If @tp_RootFolder does not reference a valid root folder, then fn_GetRootFolder MUST return NULL. Otherwise, fn_GetRootFolder MUST return the full URL of the specified root folder.

3.1.5.4 fn_HtmlEncode

The fn_HtmlEncode function is called to replace ampersand, less-than, greater-than, single-quote, double-quote, and line-feed characters in input strings with the appropriate entity references. The T-SQL syntax for the function is as follows.

```
FUNCTION fn_HtmlEncode (
    @Value            nvarchar(1023),
    @PreserveNewLine  bit
)
RETURNS              nvarchar(4000);
```

@Value: The string to be encoded.

@PreserveNewLine: A bit value that specifies whether to append
 after the line-feed character. If the value is 1,
 MUST be appended to every line-feed character.

Return Value: The string that is converted according to the following table.

Character in input string	HTML character-entity in output string
& (ampersand)	&
< (less-than)	<
> (greater-than)	>
' (single-quote)	'
" (double-quote)	"

Character in input string	HTML character-entity in output string
"\n"(ASCII line feed)	"\n" (if @PreserveNewLine is 1)

If the converted string has more than 4000 characters, only the first 4000 characters MUST be returned.

3.1.5.5 fn_IsOverQuotaOrWriteLocked

The fn_IsOverQuotaOrWriteLocked function is called to get the Disk Quota and Write Lock status of a site collection. The T-SQL syntax for the function is as follows:

```

FUNCTION fn_IsOverQuotaOrWriteLocked (
    @SiteId    uniqueidentifier
)
RETURNS      int;

```

@SiteId: The site collection identifier of the site collection.

Return Value: An integer value MUST be one of the following values:

Value	Description
0	Site collection disk space used does not exceed Disk Quota and the site collection is not locked by Write lock.
1	Site collection disk space used exceeds Disk Quota and the site collection is not locked by Write lock.
2	Site collection disk space used does not exceed Disk Quota and the site collection is locked by Write lock.
3	Site collection disk space used exceeds Disk Quota and the site collection is locked by Write lock.

3.1.5.6 fn_LocalDayFromUTCDate

The fn_LocalDayFromUTCDate function is called to calculate the day in local time from the specified UTC (Coordinated Universal Time) date and time value and the specified time zone information. The T-SQL (Transact-Structured Query Language) syntax for the function is as follows:

```

FUNCTION fn_LocalDayFromUTCDate (
    @dtUTC      datetime,
    @BiasMinutes int,
    @Dlt_m      int,
    @Dlt_nwd    int,
    @Dlt_wd     int,
    @Dlt_h      int,
    @Std_m      int,
    @Std_nwd    int,
    @Std_wd     int,
    @Std_h      int
)

```

RETURNS datetime;

@dtUTC: The date and time value, expressed in UTC, for which the local day is desired.

@BiasMinutes: The difference, in minutes, between UTC and local time for the time zone.

@Dlt_m: The month in which Daylight Saving Time begins for the time zone.

@Dlt_nwd: The week of the month in which Daylight Saving Time begins for the time zone.

@Dlt_wd: The weekday on which Daylight Saving Time begins for the time zone.

@Dlt_h: The hour of the day when Daylight Saving Time begins for the time zone.

@Std_m: The month in which Daylight Saving Time ends for the time zone.

@Std_nwd: The week of the month in which Daylight Saving Time ends for the time zone.

@Std_wd: The weekday on which Daylight Saving Time ends for the time zone.

@Std_h: The hour of the day when Daylight Saving Time ends for the time zone.

The values for the hour, weekday, week, and month parameters **MUST** be within the following ranges:

Parameter	Valid Range
Hour	0 – 23
Weekday	0 (Sunday) – 6 (Saturday)
Week	1 – 5, where 5 indicates the final occurrence during the month if that day of the week does not occur 5 times.
Month	1 – 12

Return Value: The function **MUST** return a date and time value expressed in local time that occurs on the same day as the specified UTC date and time value.

3.1.5.7 proc_AddAuditEntry

The proc_AddAuditEntry stored procedure is called to add an audit entry about an object identified by its identifier to the audit log.

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

```
PROCEDURE proc_AddAuditEntry(  
    @SiteId          uniqueidentifier,  
    @ItemId           uniqueidentifier,  
    @ItemType        smallint,  
    @UserId          int,  
    @MachineName     nvarchar(128) = NULL,  
    @MachineIp       nvarchar(20)  = NULL,  
    @Location        nvarchar(260) = NULL,  
    @LocationType    tinyint = NULL,  
    @Occurred        datetime,  
    @Event           int,
```

```

@EventName      nvarchar(128) = NULL,
@EventSource    tinyint,
@SourceName     nvarchar(256) = NULL,
@EventData      nvarchar(max) = NULL,
@RequestGuid    uniqueidentifier = NULL OUTPUT
);

```

@SiteId: The site collection identifier which contains the object for which the audit entry is being added.

@ItemId: The identifier of the object.

@ItemType: The type of object, which MUST be a value from the Audit Item Type enumeration as specified in [\[MS-WSSFO2\]](#), Section [2.2.4.2](#).

@UserId: The user identifier of the user who performed the operation that generated the audit entry.

@MachineName: Reserved. MUST be NULL.

@MachineIp: Reserved. MUST be NULL.

@Location: The **URL** of the object. MUST be a **store-relative URL**.

@LocationType: Reserved. MUST be 0x00.

@Occurred: The date and time, in UTC, when the operation occurred.

@Event: The type of Audit Event, which MUST be a value from the [Audit Event Type](#) enumeration.

@EventName: If @Event is 0x00000064 (Custom), then this is the name of the event, which MUST be a non-empty string. Otherwise, this MUST be NULL.

@EventSource: The source of the Audit Event, which MUST be a value from the [Audit Event Source](#) enumeration.

@SourceName: The display name of the Audit Event source.

@EventData: The event data of the Audit Event. This SHOULD [<1>](#) be less than or equal to 4000 characters.

@RequestGuid: The optional **request identifier** for the current request.

Return Code Values An integer which MUST be one of the following values.

Value	Description
0	Successful completion.
2	The object with the specified type or its parent cannot be found.

Result Sets: MUST NOT return any result sets.

3.1.5.8 **proc_AddAuditEntryUrl**

The `proc_AddAuditEntryUrl` stored procedure is called to add an audit entry about an object identified by its URL (as specified by its directory name and **leaf name**) to the audit log.

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

```
PROCEDURE proc_AddAuditEntryUrl(
    @SiteId            uniqueidentifier,
    @DirName            nvarchar(256),
    @LeafName           nvarchar(128),
    @ItemType           smallint,
    @UserId             int,
    @MachineName         nvarchar(128) = NULL,
    @MachineIp          nvarchar(20) = NULL,
    @Location            nvarchar(260) = NULL,
    @LocationType        tinyint = NULL,
    @Occurred           datetime,
    @Event              int,
    @EventName          nvarchar(128) = NULL,
    @EventSource         tinyint,
    @SourceName         nvarchar(256) = NULL,
    @EventData          nvarchar(max) = NULL
    @EventFlag          int = 0,
    @RequestGuid         uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The site collection identifier which contains the object for which the audit entry is being added.

@DirName: The directory name of the object.

@LeafName: The leaf name of the object.

@ItemType: The type of object, which MUST be a value from the Audit Item Type enumeration as specified in [\[MS-WSSFO2\]](#), Section [2.2.4.2](#).

@UserId: The user identifier of the user who performed the operation that generated the audit entry.

@MachineName: Reserved. MUST be NULL.

@MachineIp: Reserved. MUST be NULL.

@Location: The URL of the object. MUST be a store-relative URL.

@LocationType: Reserved. MUST be 0x00.

@Occurred: The date and time, in **UTC**, when the operation occurred.

@Event: The type of Audit Event, which MUST be a value from the [Audit Event Type](#) enumeration.

@EventName: If @Event is 0x00000064 (Custom), then this is the name of the event, which MUST be a non-empty string. Otherwise, this MUST be NULL.

@EventSource: The source of the Audit Event, which MUST be a value from the [Audit Event Source](#) enumeration.

@SourceName: The display name of the Audit Event source.

@EventData: The event data of the Audit Event. This SHOULD [<2>](#) be less than or equal to 4000 characters.

@EventFlag: The audit flags, as specified in [MS-WSSFO2], Section [2.2.3.1](#), for which the @Event applies, which MUST be an audit flags bit mask. The protocol server determines the direct, inherited, and global audit flags applicable to the object and adds the audit entry if @EventFlag is zero or (@EventFlag & ({AuditFlags} | {InheritAuditFlags} | {GlobalAuditMask})) is nonzero.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be one of the following values.

Value	Description
0	Successful completion.
2	The object with the specified type or its parent cannot be found.

Result Sets: MUST NOT return any result sets.

3.1.5.9 proc_CalculateAndUpdateSiteDiskUsed

The proc_CalculateAndUpdateSiteDiskUsed stored procedure is called to calculate and update the size of disk, in Bytes, used by a site collection.

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

```
PROCEDURE proc_CalculateAndUpdateSiteDiskUsed (  
    @SiteId            uniqueidentifier,  
    @RequestGuid       uniqueidentifier = NULL OUTPUT  
  
);
```

@SiteId: The site collection identifier of the site collection of which disk size is calculated and updated.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.5.10 proc_ConfirmSiteUsage

The proc_ConfirmSiteUsage stored procedure is called to confirm that the specified site collection is in use and to specify whether it can be deleted automatically when not in use.

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

```
PROCEDURE proc_ConfirmSiteUsage(  
    @SiteId            uniqueidentifier,  
    @Disable           bit  
    @RequestGuid       uniqueidentifier = NULL OUTPUT  
  
);
```

@SiteId: The site collection identifier of the site collection which is being confirmed.

@Disable: A bit value that specifies if the site collection can be deleted automatically when not in use. @Disable MUST be one of the following values:

Value	Description
0	Delete site collection automatically when not in use.
1	Do not delete site collection automatically when not in use.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.5.11 **proc_ConvertStringToDate**

The `proc_ConvertStringToDate` stored procedure is called to transform text stored in a **field** into a **datetime** string. The conversion logic only preserves the date part and sets the time part to 1:00 AM in the site local time of the site which contains the specified list.

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

```
PROCEDURE proc_ConvertStringToDate (  
    @ListId          uniqueidentifier,  
    @OldColName      nvarchar(255),  
    @OldRowOrdinal   int,  
    @NewColName      nvarchar(255),  
    @NewRowOrdinal   int,  
    @WebTimeBias     int,  
    @DateOrder       nvarchar(3),  
    @bFromNtext      bit,  
    @RequestGuid     uniqueidentifier = NULL OUTPUT  
);
```

@ListId: The list identifier of the list where this data transformation takes place.

@OldColName: The database column name in which the text data is stored.

@OldRowOrdinal: A **zero-based index** number that specifies which row the text field data is stored in the UserData View (as defined in [\[MS-WSSFO2\]](#), Section [2.2.8.8](#)).

@NewColName: The database column name in which to store the converted datetime string field data.

@NewRowOrdinal: A zero-based index number that specifies which row the datetime string field data will be stored in the UserData View (as defined in [\[MS-WSSFO2\]](#), Section [2.2.8.8](#)).

@WebTimeBias: An integer that specifies the difference in minutes between the current time zone and **Coordinated Universal Time (UTC)**. This number MUST take daylight savings time into consideration where applicable. The stored procedure then uses this number to convert the text field into UTC times. It assumes the text string contains a local time.

@DateOrder: A string of three characters. It will be passed to T-SQL statement "SET DATEFORMAT". This setting is used by the back-end database server in the interpretation of

character strings as they are converted to date values. It has no effect on the display of date values. @DateOrder MUST be one of the following values.

Value	Description
'mdy'	The date string will be interpreted as "Month, Day, Year"
'dmy'	The date string will be interpreted as "Day, Month, Year"
'ymd'	The date string will be interpreted as "Year, Month, Day"

@bFromNText: A bit that specifies how the stored procedure considers the database column specified by @OldColName. @bFromNText MUST be either 0 or 1. If @bFromNText is 1, the stored procedure will assume the specified database column as an ntext type column. Otherwise, it will assume the database column as a nvarchar(255) column.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: This stored procedure MUST return 0 upon completion.

Result Sets: MUST NOT return any result sets.

3.1.5.12 proc_CountAuditEntries

The proc_CountAuditEntries stored procedure is called to count the number of audit entries in the audit log for a site collection.

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

```
PROCEDURE proc_CountAuditEntries(  
    @SiteId    uniqueidentifier,  
    @Entries   bigint          OUTPUT  
)  
;
```

@SiteId: The site collection identifier of the site collection.

@Entries: The number of audit entries for the site collection.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.5.13 proc_DefragmentIndices

The proc_DefragmentIndices stored procedure is called to rebuild indices that are used internally by the back-end database server on user tables in the content database. Indices are used by the database server to optimize query performance.

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

```
PROCEDURE proc_DefragmentIndices ();
```

proc_DefragmentIndices MUST NOT take any parameters.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.5.14 **proc_DeleteRecycleBinItem**

The `proc_DeleteRecycleBinItem` stored procedure is called to permanently delete a Recycle Bin item.

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

```
PROCEDURE proc_DeleteRecycleBinItem(
    @SiteId                uniqueidentifier,
    @WebId                  uniqueidentifier,
    @UserId                  int,
    @ThresholdRowCount      int,
    @DeleteTransactionId    varbinary(16),
    @RequestGuid            uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The site collection identifier of the site collection which contains the Recycle Bin item. This parameter MUST NOT be NULL.

@WebId: The site identifier of the site which contains the Recycle Bin item. If the @UserId parameter is nonzero, then this parameter MUST NOT be NULL.

@UserId: An Integer that MUST either be 0 or the user identifier of the User who deleted the Recycle Bin item. If the value is 0, the store procedure can delete the recycle bin item of any user. Otherwise, it can only delete the item in the recycle bin of the user specified by @UserId. This parameter MUST NOT be NULL.

@ThresholdRowCount: If value is not 0, it specifies the Maximum number of items the container is allowed to have when the recycle item being deleted is a container such as a list, folder or folder within a list). If the container has more items than this value, the delete operation will fail. If value is 0, there is no limit. The value MUST be 0 or the same as the query size threshold on the specific web application.

@DeleteTransactionId: The delete transaction identifier of the delete transaction. DeleteTransactionId is a varbinary(16) type. It MUST contain a valid GUID and MUST NOT be NULL.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be one of the following values.

Value	Description
0	Successful completion.
36	The number of items in the container is greater than @ThresholdRowCount
1168	Either the Recycle Bin item could not be found or the site which contains the Recycle Bin item could not be found.
1359	An internal error occurred.

Result Sets: MUST NOT return any result sets.

3.1.5.15 proc_DetectOrphans

The proc_DetectOrphans stored procedure is called to detect orphaned objects in the content database for reporting purposes.

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

```
PROCEDURE proc_DetectOrphans(  
    @RequestGuid          uniqueidentifier = NULL OUTPUT  
)  
;
```

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST return five result sets in the following order:

3.1.5.15.1 Site Collection with No Sites Result Set

The Site Collection with no Sites result set returns site collections without sites.

The T-SQL syntax for the result set is defined in the [Site Collection with no Sites Result Set](#).

3.1.5.15.2 Sites with No Site Collection Result Set

The Sites with no Site Collection result set returns sites with no site collection and sites with no root folder.

The T-SQL syntax for the result set is defined in the [Sites with no Site Collection Result Set](#).

3.1.5.15.3 Sites with No Parent Site Result Set

The Sites with no Parent Site result set returns those sites that have no parent site.

The T-SQL syntax for the result set is defined in the [Sites with no Parent Site Result Set](#).

3.1.5.15.4 Folders with No Site Result Set

The Folders with no Site result set returns folders at the root of the site that have no associated site.

The T-SQL syntax for the result set is defined in the [Folders with no Site Result Set](#).

3.1.5.15.5 Orphaned Lists Result Set

The Orphaned Lists result set returns lists in the following three categories:

- Lists with no parent site
- Lists with documents with no parent list
- Lists with items with no parent list

The Orphaned Lists result set MUST return a result set having a number of rows equal to the number of orphaned lists. If there are no such lists, the result set returns zero rows.

The T-SQL syntax for the result set is defined in the [Orphaned Lists Result Set](#).

3.1.5.16 proc_DetectOrphansFix

The proc_DetectOrphansFix stored procedure is called to detect orphaned objects in the content database upon a database repair attempt.

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

```
PROCEDURE proc_DetectOrphansFix(  
    @RequestGuid          uniqueidentifier = NULL OUTPUT  
)  
;
```

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST return five result sets in the following order:

3.1.5.16.1 Site Collection with No Sites Result Set

The Site Collection with no Sites result set returns site collections without sites.

The T-SQL syntax for the result set is defined in the [Site Collection with no Sites Result Set](#).

3.1.5.16.2 Sites with No Site Collection Result Set

The Sites with no Site Collection result set returns sites with no site collection and sites with no root folder.

The T-SQL syntax for the result set is defined in the [Sites with no Site Collection Result Set](#).

3.1.5.16.3 Sites with No Parent Site Result Set

The Sites with No Parent Site result set returns sites with no parent site.

The T-SQL syntax for the result set is defined in the [Sites with no Parent Site Result Set](#).

3.1.5.16.4 Folders with No Site Result Set

The Folders with no Site result set returns folders at the root of the site that have no associated site.

The T-SQL syntax for the result set is defined in the [Folders with no Site Result Set](#).

3.1.5.16.5 Orphaned Lists Result Set

Orphaned Lists result set returns lists in the following three categories:

- Lists with no parent site
- Lists with documents with no parent list
- Lists with items with no parent list

The Orphaned Lists result set MUST return a result set having a number of rows equal to the number of orphaned lists. If there are no such lists, the result set returns 0 rows.

The T-SQL syntax for the result set is defined in the [Orphaned Lists Result Set](#).

3.1.5.17 proc_DTSetRelationship

The proc_DTSetRelationship stored procedure is called to set the parent/child relationship between two documents. The documents are an original document (the parent) and a document created by transforming the original document (the child).

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

```
PROCEDURE proc_DTSetRelationship (
    @SiteId            uniqueidentifier,
    @DirName           nvarchar(256),
    @ChildDocLevel     tinyint,
    @ChildLeafName     nvarchar(128),
    @ParentDocLevel    tinyint,
    @ParentLeafName    nvarchar(128),
    @TransformerId     uniqueidentifier,
    @ParentVersion     int,
    @RequestGuid       uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The site collection identifier of the site collection which contains the parent and child documents.

@DirName: The directory name of the documents.

@ChildDocLevel: The **publishing level** for the child document.

@ChildLeafName: The leaf name for the child document. This value MUST NOT be NULL.

@ParentDocLevel: The publishing level for the parent document.

@ParentLeafName: The leaf name for the parent document. This value MUST NOT be NULL.

@TransformerId: The identifier of the agent that performed the transformation. If the child document is a transformed version of the parent document, this MUST be the GUID of the agent that performed the transformation. Otherwise, @TransformerId MUST be NULL.

@ParentVersion: The document version of the parent document.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: This stored procedure MUST return 0 upon completion.

Result Sets: MUST NOT return any result sets.

3.1.5.18 proc_EnumRecycleBinItemsForCleanup

The proc_EnumRecycleBinItemsForCleanup stored procedure is called to get the Recycle Bin items that have been in the Recycle Bin longer than the specified number of days. Recycle Bin items whose site has been deleted will not be included in the result set.

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

```
PROCEDURE proc_EnumRecycleBinItemsForCleanup (
    @Days             int,
    @RequestGuid       uniqueidentifier = NULL OUTPUT
);
```

);

@Days: An integer that specifies the number of days a Recycle Bin item needs to be in the Recycle Bin to be included in the result set. If this value is NULL, an empty result set MUST be returned. If this value is 0 or a negative number, then all of the Recycle Bin items MUST be returned.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST return the following result set:

3.1.5.18.1 Recycle Bin Items For Cleanup Result Set

The Recycle Bin Items For Cleanup Result Set returns a list of Recycle Bin items. The `proc_EnumRecycleBinItemsForCleanup` stored procedure MUST always return a Recycle Bin Items For Cleanup Result Set, and the maximum number of rows it can contain is 1000. The T-SQL syntax for the result set is as follows.

```
SiteId                uniqueidentifier,  
WebId                 uniqueidentifier,  
{DeleteTransactionId} uniqueidentifier;
```

SiteId: The site collection identifier of the site collection which contains the Recycle Bin item. SiteId MUST NOT be NULL.

WebId: The site identifier of the site which contains the Recycle Bin item. WebId MUST NOT be NULL.

{DeleteTransactionId}: The delete transaction identifier of the delete transaction. {DeleteTransactionId} MUST NOT be NULL.

3.1.5.19 proc_EnumRecycleBinToFreeSecondStageQuota

The `proc_EnumRecycleBinToFreeSecondStageQuota` stored procedure is called to enumerate the size and delete transaction identifier information of each deleted Recycle Bin item in the second-stage Recycle Bin.

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

```
PROCEDURE proc_EnumRecycleBinToFreeSecondStageQuota (  
    @SiteId            uniqueidentifier,  
    @RequestGuid       uniqueidentifier = NULL OUTPUT  
) ;
```

@SiteId: The site collection identifier of the site collection which contains the second-stage Recycle Bin.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST return the following result set:

3.1.5.19.1 Item Metadata Result Set

The Item Metadata Result Set returns metadata information about deleted items in the second-stage Recycle Bin of a site collection. It contains the size, in bytes, and the delete transaction identifier of the deleted Recycle Bin items. The T-SQL syntax for the result set is as follows.

```
{DeleteTransactionId}    uniqueidentifier,  
Size                     bigint;
```

{DeleteTransactionId}: The delete transaction identifier of the delete transaction on this Recycle Bin item.

Size: The size, in bytes, of the Recycle Bin item.

3.1.5.20 proc_EnumSitesForDeadWebCheck

The proc_EnumSitesForDeadWebCheck stored procedure is called to return information about top-level sites such as the number of days that have elapsed between the last site certification and the current date, the **e-mail address** of the site collection administrators, and the **notify count** of the top-level site.

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

```
PROCEDURE proc_EnumSitesForDeadWebCheck(  
    @RequestGuid          uniqueidentifier = NULL OUTPUT  
);
```

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST return the following result set:

3.1.5.20.1 Site Information Result Set

The Site Information result set returns information about top-level sites such as the number of days that have elapsed between the last site certification and the current date, the e-mail address of the site collection administrators, and the notify count of the top-level site. The Site Information result set MUST return one row for every combination of top-level site and site collection administrators for that site or no rows if there are no top-level sites. The T-SQL syntax for the result set is as follows.

```
Id                      uniqueidentifier,  
{DiffTodayCertDate}    int,  
DeadWebNotifyCount      smallint,  
FullUrl                 nvarchar(256),  
Title                  nvarchar(255),  
Language               int,  
tp_Email               nvarchar(255);
```

Id: The site identifier of the top-level site.

{DiffTodayCertDate}: The number of days between the certification date of the top-level site to the current date.

DeadWebNotifyCount: The notify count value of the top-level site.

FullUrl: The store-relative form of the top-level site.

Title: The title of the top-level site.

Language: The **language code identifier (LCID)** which indicates the language of a top-level site.

tp_Email: The e-mail address, if it exists, for the site collection administrators associated with the top-level site.

3.1.5.21 **proc_ForceDeleteList**

The `proc_ForceDeleteList` stored procedure is called to delete a list in a corrupted state. It is intended as a last resort to remove a list. The list will be permanently deleted without being moved to the Recycle Bin.

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

```
PROCEDURE proc_ForceDeleteList(  
    @SiteId          uniqueidentifier,  
    @DirName         nvarchar(256),  
    @LeafName        nvarchar(128),  
    @UserId          int,  
    @RequestGuid     uniqueidentifier = NULL OUTPUT  
) ;
```

@SiteId: The site collection identifier of the site collection which contains the list to be deleted.

@DirName: The store-relative form of the folder which contains the list specified by `@LeafName`.

@LeafName: The leaf name of the list to be deleted.

@UserId: The user identifier of the **current user**.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which **MUST** be one of the following values.

Value	Description
0	Successful completion.
2	The list does not exist.
87	The specified <code>@SiteId</code> , <code>@DirName</code> and <code>@LeafName</code> do not point to a root folder of a list.
8398	A list marked as undeletable cannot be deleted.

Result Sets: **MUST NOT** return any result sets.

3.1.5.22 **proc_GetAdminRecycleBinInfo**

The `proc_GetAdminRecycleBinInfo` stored procedure is called to get information about the first-stage Recycle Bin for a site collection, including the number of Recycle Bin items it contains and the overall size of those Recycle Bin items.

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

```
PROCEDURE proc_GetAdminRecycleBinInfo(  
    @SiteId          uniqueidentifier,  
    @ItemCount       int          OUTPUT,  
    @Size            bigint       OUTPUT,  
    @RequestGuid     uniqueidentifier = NULL OUTPUT  
);
```

@SiteId: The site collection identifier of the site collection which contains the Recycle Bin. This parameter MUST NOT be NULL.

@ItemCount: The total number of Recycle Bin items in the first-stage Recycle Bin of the site collection specified by the @SiteId parameter. This is an output parameter. This parameter MUST NOT be NULL and MUST be a non-negative numeric value.

@Size: The total size, in Bytes, of the Recycle Bin items in the first-stage Recycle Bin of the site collection specified by the @SiteId parameter. This is an output parameter. This parameter MUST NOT be NULL and MUST be a non-negative numeric value.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.5.23 proc_GetAdminRecycleBinItems

The proc_GetAdminRecycleBinItems stored procedure is called to get the full list of Recycle Bin items for a particular site collection.

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

```
PROCEDURE proc_GetAdminRecycleBinItems(  
    @SiteId          uniqueidentifier,  
    @RequestGuid     uniqueidentifier = NULL OUTPUT  
);
```

@SiteId: The site collection identifier of the site collection which contains the Recycle Bin whose Recycle Bin items MUST be returned. This parameter MUST NOT be NULL.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST return the following result set:

3.1.5.23.1 Admin Recycle Bin Items Result Set

The Admin Recycle Bin Items Result Set returns the details of a set of Recycle Bin items. There is no limit on the number of rows that the Admin Recycle Bin Items Result Set can contain. The T-SQL syntax for the result set is as follows.

Title	nvarchar(260),
DirName	nvarchar(256),


```

LeafName          nvarchar(128),
AuthorId          int,
AuthorName        nvarchar(255),
AuthorEmail       nvarchar(255),
DeleteDate        datetime,
{Size}            bigint,
DeleteUserId      int,
DeletedByName     nvarchar(255),
DeletedByEmail    nvarchar(255),
DeleteTransactionId varbinary(16),
ItemType          tinyint,
OriginalItemType  tinyint,
BinId             tinyint,
{tp_ImageUrl}     nvarchar(255),
ProgId            nvarchar(255),
WebId             uniqueidentifier;

```

Title: The display name of the Recycle Bin item. Title MUST NOT be NULL.

DirName: The directory name of the Recycle Bin item. DirName MUST NOT be NULL.

LeafName: The file name of the Recycle Bin item. LeafName MUST NOT be NULL.

AuthorId: The identifier of the User who is the **author** of the Recycle Bin item.

AuthorName: The name of the User who is the author of the Recycle Bin item.

AuthorEmail: The e-mail address of the User who is the author of the Recycle Bin item.

DeleteDate: The date when the Recycle Bin item was deleted. DeleteDate MUST NOT be NULL.

{Size}: The size, in Bytes, of the Recycle Bin item. Size MUST NOT be NULL.

DeleteUserId: The identifier of the User who deleted the Recycle Bin item.

DeletedByName: The display name of the User who deleted the Recycle Bin item.

DeletedByEmail: The e-mail address of the User who deleted the Recycle Bin item.

DeleteTransactionId: The delete transaction identifier of the delete transaction. DeleteTransactionId is a varbinary(16) value. It MUST contain a valid GUID and MUST NOT be NULL.

ItemType: A numerical value indicating the type of the Recycle Bin item. ItemType MUST NOT be NULL and MUST be one of the values in [Delete Item Type](#).

OriginalItemType: A numerical value indicating the type of the original item type for the Recycle Bin item that is the parent item in a cascading delete operation. OriginalItemtype MUST be NULL or one of the values in Delete Item Type.

BinId: A numerical value that indicates which [Recycle Bin Stage](#) the Recycle Bin item is in. BinId MUST NOT be NULL and MUST be one of the following values.

Value	Description
1	The Recycle Bin item is in the first-stage Recycle Bin.
2	The Recycle Bin item is in the second-stage Recycle Bin.

{tp_ImageUrl}: The store-relative URL for the icon of the list if the Recycle Bin item is a list or list item.

ProgId: The file type of the Recycle Bin item if it is a **file**.

WebId: The site identifier of the site which contains the Recycle Bin item. WebId MUST NOT be NULL.

3.1.5.24 **proc_GetAllSPWebIdentifiersGivenSiteGuid**

The `proc_GetAllSPWebIdentifiersGivenSiteGuid` stored procedure is called to get the store-relative URL of those sites whose parent site collection has the input site collection identifier.

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

```
PROCEDURE proc_GetAllSPWebIdentifiersGivenSiteGuid (  
    @SiteGuid      uniqueidentifier  
);
```

@SiteGuid: The site collection identifier of the parent site collection.

Return Code Values: This stored procedure MUST return 0 upon completion.

Result Sets: MUST return the following result set:

3.1.5.24.1 **All SPWeb Identifiers Given Site Guid Result Set**

The All SPWeb Identifiers Given Site Guid result set MUST contain no rows if no site's parent site collection matches the input site collection identifier. Otherwise it MUST contain one row for each site in the collection. The T-SQL syntax for the result set is:

```
FullUrl nvarchar (256) NOT NULL;
```

FullUrl: Gives the full URL to identify a site within a given site collection.

3.1.5.25 **proc_GetAuditEntries**

The `proc_GetAuditEntries` stored procedure is called to retrieve the audit entries for a site collection that satisfy the filter conditions.

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

```
PROCEDURE proc_GetAuditEntries (  
    @RowLimit      int,  
    @SiteId        uniqueidentifier,  
    @ItemId         uniqueidentifier = NULL,  
    @StartTime     datetime = NULL,  
    @EndTime       datetime = NULL,  
    @EventCSV      nvarchar(max) = NULL,  
    @UserId        int = NULL,  
    @DirName       nvarchar(260) = NULL  
);
```

@RowLimit: The maximum number of audit entries to retrieve.

@SiteId: The site collection identifier of the site collection.

@ItemId: The identifier of the object in the site collection. If this condition is not NULL, the protocol server returns audit entries that are applicable for this object, and @DirName SHOULD be NULL and MUST be ignored.

@StartTime: The start time condition for the audit entries, in Coordinated Universal Time (UTC). If this condition is not NULL, the protocol server returns audit entries that occurred after this time.

@EndTime: The end time query for the audit entries, in Coordinated Universal Time (UTC). If this condition is not NULL, the protocol server returns audit entries that occurred prior to this time.

@EventCSV: The comma-separated list of event types to include. The event types MUST be values from the [Audit Event Type](#) enumeration. For example, the @EventCSV value "0x00000001, 0x00000002" filters audit entries to check-in and check-out events only. If this is NULL, audit entries with any event type will be returned.

@UserId: The user identifier of the user who performed the operation that generated the audit entry. If this condition is not NULL, the protocol server returns audit entries that were performed by this user.

@DirName: The directory name query for the audit entries. If this condition is not NULL, the protocol server returns audit entries that were performed on objects with location starting with this directory name.

Return Code Values: An integer which MUST be 0.

Result Sets: This procedure MUST return a **Get Audit Entries Result Set**.

3.1.5.25.1 Get Audit Entries Result Set

The Get Audit Entries Result Set returns a list of audit entries.

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

SiteId	uniqueidentifier,
ItemId	uniqueidentifier,
ItemType	smallint,
UserId	int,
MachineName	nvarchar(128),
MachineIp	nvarchar(20),
DocLocation	nvarchar(260),
LocationType	tinyint,
Occurred	datetime,
Event	int,
EventName	nvarchar(128),
EventSource	tinyint,
SourceName	nvarchar(256),
EventData	nvarchar(max)

SiteId: The site collection identifier of the site collection which contains the object for which the audit entry was added.

ItemId: The identifier of the object.

ItemType: The type of object, which MUST be a value as specified in [\[MS-WSSFO2\]](#), Section [2.2.4.2](#).

UserId: The user identifier of the user who performed the operation that generated the audit entry.

MachineName: Reserved. MUST be NULL.

MachineIp: Reserved. MUST be NULL.

DocLocation: The location of the object.

LocationType: Reserved. MUST be 0x00.

Occurred: The date and time, in Coordinated Universal Time (UTC), when the operation occurred.

Event: The type of audit event, which MUST be a value from the [Audit Event Type](#) enumeration.

EventName: The name of the audit event.

EventSource: The source of the audit event, which MUST be a value from the [Audit Event Source](#) enumeration.

SourceName: The display name of the audit event source.

EventData: The event data of the audit event.

3.1.5.26 **proc_GetCustomizedDocumentsInWeb**

The `proc_GetCustomizedDocumentsInWeb` stored procedure is called to get document identifiers for the customized documents in a site.

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

```
PROCEDURE proc_GetCustomizedDocumentsInWeb (  
    @SiteId          uniqueidentifier,  
    @WebFullUrl      nvarchar(260),  
    @DocIdLast       uniqueidentifier,  
    @RequestGuid     uniqueidentifier = NULL OUTPUT  
);
```

@SiteId: The site collection identifier of the site collection which contains the requested documents.

@WebFullUrl: The store-relative URL of the site which contains the requested documents.

@DocIdLast: The lowest document identifier considered for the selection.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: This stored procedure MUST return 0 upon completion.

Result Sets: MUST return the following result set:

3.1.5.26.1 **DocumentID Result Set**

The DocumentID result set returns the document identifiers of the customized documents in the specified site which are greater than the `@DocIdLast` parameter. The document identifiers are arranged in an ascending order. If there are more than 1000 such document identifiers, then only the top 1000 are returned. The T-SQL syntax for the result set is as follows.

Id uniqueidentifier;

Id: The document identifier of the customized document.

3.1.5.27 proc_GetDeadWebInfo

The proc_GetDeadWebInfo stored procedure is called to get the last certification date and the notify count of the specified site collection.

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

```
PROCEDURE proc_GetDeadWebInfo (
    @SiteId           uniqueidentifier,
    @RequestGuid       uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The site collection identifier of the site collection.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST return the following result set:

3.1.5.27.1 Dead Web Information Result Set

The Dead Web Information result set returns information about the specified site collection. The Dead Web Information result set MUST return one row for the specified site collection or no rows if there is no such site collection. The T-SQL syntax for the result set is as follows.

```
CertificationDate       datetime
DeadWebNotifyCount      smallint
```

CertificationDate: The last certification date of the site collection.

DeadWebNotifyCount: The value of notify count of the site collection.

3.1.5.28 proc_GetDocLibrarySizes

The proc_GetDocLibrarySizes stored procedure is called to get the size, in Bytes, and metadata information of the document libraries of a site collection.

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

```
PROCEDURE proc_GetDocLibrarySizes (
    @SiteId           uniqueidentifier,
    @RequestGuid       uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The site collection identifier of the site collection.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: This stored procedure MUST return 0 upon completion.

Result Sets: MUST return the following result set:

3.1.5.28.1 Document Library Size Result Set

proc_GetDocLibrarySizes returns the size, in Bytes, and metadata information of the document libraries of a site collection. The Document Library Size result set MUST contain no rows if there is no such site collection or there are no document libraries in the site collection. Otherwise, it MUST contain one row for each document library in the site collection. The T-SQL syntax for the result set is as follows.

TotalSize	bigint,
tp_Id	uniqueidentifier,
tp_WebId	uniqueidentifier,
tp_Modified	datetime,
tp_ServerTemplate	int,
DirName	nvarchar(256),
LeafName	nvarchar(128),
tp_ImageUrl	nvarchar(255),
tp_Title	nvarchar(255),
tp_ItemCount	int,
WebTemplate	int,
Language	int,
FullUrl	nvarchar(256);

TotalSize: The sum of the document sizes, document version sizes, personalization sizes and Web Part sizes in Bytes.

tp_Id: The list identifier of the document library.

tp_WebId: The site identifier of the site containing the document library.

tp_Modified: The date and time the document library was last modified.

tp_ServerTemplate: The **list server template** used to create the list.

DirName: The directory name of the document library.

LeafName: The leaf name of the document library.

tp_ImageUrl: The store-relative form of the image associated with the document library.

tp_Title: The name of the document library.

tp_ItemCount: The number of items in the document library.

WebTemplate: An integer identifying the **site template** of the site which contains the document library.

Language: The LCID that indicates the language of a site.

FullUrl: The store-relative URL of the site.

3.1.5.29 proc_GetDocSizeInfo

The `proc_GetDocSizeInfo` stored procedure is called to get the size, in Bytes, and the metadata information (described in detail in the following Document Size result set section) of each document of a site collection.

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

```
PROCEDURE proc_GetDocSizeInfo (  
    @SiteId          uniqueidentifier,  
    @RequestGuid     uniqueidentifier = NULL OUTPUT  
);
```

@SiteId: The site collection identifier of the site collection.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: This stored procedure MUST return 0 upon completion.

Result Sets: MUST return the following result set:

3.1.5.29.1 Document Size Result Set

`proc_GetDocSizeInfo` returns the metadata and different kinds of size information for each document. The Document Size result set MUST contain no rows if there is no such site collection or there are no documents in the site collection. Otherwise, it MUST contain one row for each document in the site collection. The T-SQL syntax for the result set is as follows.

TotalSize	bigint,
TimeLastModified	datetime,
Id	uniqueidentifier,
DirName	nvarchar(256),
LeafName	nvarchar(128),
Size	int,
SetupPath	nvarchar(255),
{PersonalizationandWebPartSize}	bigint,
{DocumentVersionSize}	bigint,
DoclibRowId	int,
WebId	uniqueidentifier,
{DocListId}	uniqueidentifier,
FullUrl	nvarchar(256),
Language	int,
{WebPartTypeSize}	bigint;

TotalSize: The total size, in Bytes, of the document, personalization, Web Part and document version of a document.

TimeLastModified: The date and time the document was last modified.

Id: The document identifier of the document.

DirName: The directory name of the document.

LeafName: The leaf name of the document.

Size: The document size, in Bytes.

SetupPath: If the document is or ever was **uncustomized**, then this is the relative path of the document (relative to the **setup path**). This is an empty string otherwise.

{PersonalizationandWebPartSize}: The total size, in Bytes, of the personalization and Web Parts of the document.

{DocumentVersionSize}: The total size, in Bytes, of the historical document versions of the document.

DoclibRowId: The **row** identifier of the document in a list.

WebId: The site identifier of the site.

{DocListId}: The list identifier of the list which contains the document.

FullUrl: The store-relative form of the site.

Language: The LCID which indicates the language of a site.

{WebPartTypeSize}: The Web Part type size.

3.1.5.30 **proc_GetFirstUniqueAncestorWebUrl**

The `proc_GetFirstUniqueAncestorWebUrl` stored procedure is called to get the URL in store-relative form of the first unique **ancestor** of the specified site.

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

```
PROCEDURE proc_GetFirstUniqueAncestorWebUrl(  
    @WebId          uniqueidentifier,  
    @RequestGuid    uniqueidentifier = NULL OUTPUT  
) ;
```

@WebId: The site identifier of the requested site.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: This stored procedure MUST return 0 upon completion.

Result Sets: MUST return the following result set:

3.1.5.30.1 **First Ancestor Site Url Result Set**

`proc_GetFirstUniqueAncestorWebUrl` returns the URL in store-relative form of the first unique ancestor of the specified site. This result set MUST return no rows if the requested site cannot be found. Otherwise, it MUST return one row. The T-SQL syntax for the result set is as follows.

```
FullUrl    nvarchar(256) ;
```

FullUrl: The store-relative form of the first unique ancestor of the specified site.

3.1.5.31 proc_GetListBestMatch

The proc_GetListBestMatch stored procedure is called to get the list identifier and offset of the first list starting with the value in @PathSearch when names of all lists in the site are sorted alphabetically in ascending order.

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

```
PROCEDURE proc_GetListBestMatch (
    @WebId                uniqueidentifier,
    @PathSearch            nvarchar(128),
    @BestMatchListId       uniqueidentifier OUTPUT,
    @BestMatchOffset       int OUTPUT,
    @RequestGuid           uniqueidentifier = null OUTPUT
);
```

@WebId: The site identifier of the site that contains lists.

@PathSearch: The name of the list to search.

@BestMatchListId: The list identifier of the first list whose name starts with the value of @PathSearch when the names of all lists in the site are sorted alphabetically in ascending order.

@BestMatchOffset: The zero-based offset of the first list whose name starts with the value of @PathSearch when the names of all lists in the site are sorted alphabetically in ascending order.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: This stored procedure MUST return 0 upon completion.

Result Sets: MUST NOT return any result sets.

3.1.5.32 proc_GetListSizes

The proc_GetListSizes stored procedure is called to get the size, in Bytes, and metadata information of the lists that are not document libraries of a site collection.

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

```
PROCEDURE proc_GetListSizes (
    @SiteId                uniqueidentifier,
    @RequestGuid           uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The site collection identifier of the site collection.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: This stored procedure MUST return 0 upon completion.

Result Sets: MUST return the following result set:

3.1.5.32.1 Lists Size Result Set

The Lists Size result set returns the size, in Bytes, and metadata information about the lists that are not document libraries in a site collection. The Lists Size result set MUST contain no rows if a site

collection with the site collection identifier @SiteId does not exist or if all lists in the site collection are document libraries. Otherwise, it MUST contain one row for each list in the site collection that is not a document library. The T-SQL syntax for the result set is as follows.

```
TotalSize          bigint,
tp_Id              uniqueidentifier,
tp_WebId           uniqueidentifier,
tp_Modified        datetime,
tp_ServerTemplate  int,
DirName            nvarchar(256),
LeafName           nvarchar(128),
tp_ImageUrl        nvarchar(255),
tp_Title           nvarchar(255),
tp_ItemCount       int,
WebTemplate        int,
Language           int,
FullUrl            nvarchar(256);
```

TotalSize: The total size, in Bytes, of the document and user data of a list that is not a document library.

tp_Id: The list identifier of the list that is not a document library.

tp_WebId: The site identifier of the site to which the list belongs.

tp_Modified: The date and time the list was last modified.

tp_ServerTemplate: The **list template identifier** used to create the list.

DirName: The directory name of the list.

LeafName: The leaf name of the list.

tp_ImageUrl: The store-relative form of the image associated with the list.

tp_Title: The name of the list.

tp_ItemCount: The number of Items in the list.

WebTemplate: The value that indicates the site **template** used for the site.

Language: The LCID which indicates the language of a site.

FullUrl: The store-relative form of the site.

3.1.5.33 proc_GetListSubset

The proc_GetListSubset stored procedure is called to get a subset of the lists whose row numbers are greater or equal than @StartRow and less than @StartRow+@PageSize when all the lists in the site are sorted alphabetically in the direction set in @SortDirection.

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

```
PROCEDURE proc_GetListSubset (
    @WebId          uniqueidentifier,
    @PageSize       int,
    @StartRow       int,
```

```

        @SortDirection nvarchar(4)
    );

```

@WebId: The site identifier of the site.

@PageSize: The size of returned subset. @PageSize MUST be greater than 0.

@StartRow: The row number which starts to return the subset lists. @StartRow MUST be greater or equal than 0.

@SortDirection: The string which specifies whether the sort direction is in descending or ascending order. @SortDirection MUST be either "DESC" or "ASC"

Return Code Values: This stored procedure MUST return 0 upon completion.

Result Sets: This procedure MUST return the List Subset Result Set.

3.1.5.33.1 Get List Subset Result Set

The Get List Subset result set returns the list identifier and the name of the lists that are in the subset specified by @StartRow, @PageSize and @SortDirection. The Get List Subset result set MUST contain no rows if there is no lists in the site or no lists match the condition that the row numbers are greater or equal than @StartRow and less than @StartRow+@PageSize when sorted all lists by names with the direction set in @SortDirection.

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

```

tp_Id      uniqueidentifier NOT NULL,
tp_Title   nvarchar(255) NOT NULL,

```

tp_Id: The list identifier of the list.

tp_Title: The name of the list.

3.1.5.34 proc_GetRecycleBinItemInfo

The proc_GetRecycleBinItemInfo stored procedure is called to get information about a particular Recycle Bin item.

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

```

PROCEDURE proc_GetRecycleBinItemInfo (
    @SiteId                uniqueidentifier,
    @DeleteTransactionId    varbinary(16),
    @ItemType               int OUTPUT,
    @OriginalItemType        int OUTPUT,
    @ContainingListDeleted   bit OUTPUT,
    @ContainingListName      nvarchar(255) OUTPUT,
    @DirName                 nvarchar(256) OUTPUT,
    @LeafName                nvarchar(128) OUTPUT,
    @WebUrl                  nvarchar(256) OUTPUT,
    @RequestGuid             uniqueidentifier = NULL OUTPUT
);

```

@SiteId: The site collection identifier of the site collection that contains the Recycle Bin item.

@DeleteTransactionId: The delete transaction identifier of the delete transaction. DeleteTransactionId is a varbinary(16) type, and MUST contain a valid GUID.

@ItemType: A numerical value that indicates the type of the Recycle Bin item. ItemType MUST be one of the values in [Delete Item Type](#). This is an output parameter.

@OriginalItemType: A numerical value indicating the type of the original item type for the Recycle Bin item that is the parent item in a cascading delete operation. OriginalItemtype MUST be NULL or one of the values in Delete Item Type. This is an output parameter.

@ContainingListDeleted: A numeric value that indicates whether the list that contains the Recycle Bin item has been deleted. This is an output parameter. @ContainingListDeleted can be either NULL or one of the following values:

Value	Description
0	The Recycle Bin item is itself a list or the list that contains the Recycle Bin item has not been deleted.
1	The list that contains the Recycle Bin item has been deleted.

@ContainingListName: The name of the list that contains the Recycle Bin item. This is an output parameter.

@DirName: The directory name, of the Recycle Bin item. This is an output parameter.

@LeafName: The leaf name of the Recycle Bin item. This is an output parameter.

@WebUrl: The store-relative form of the site which contains the Recycle Bin item. This is an output parameter.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be one of the values in the following table.

Value	Description
0	Successful completion.
1168	The Recycle Bin item specified by the @DeleteTransactionId parameter could not be found in the Recycle Bin of the site collection specified by the @SiteId parameter.

Result Sets: MUST NOT return any result sets.

3.1.5.35 **proc_GetRecycleBinItems**

The proc_GetRecycleBinItems stored procedure is called to get the full list of Recycle Bin items for a specified site and User.

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

```
PROCEDUREproc_GetRecycleBinItems(  
    @SiteId            uniqueidentifier,  
    @WebId             uniqueidentifier,  
    @UserId            int,
```

```

        @RequestGuid      uniqueidentifier = NULL OUTPUT
    );

```

@SiteId: The site collection identifier of the site collection that contains the Recycle Bin items. SiteId value is ignored.

@WebId: The site identifier of the site that contains the Recycle Bin items.

@UserId: The user identifier of the User who deleted the Recycle Bin items.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST return the following result set:

3.1.5.35.1 Recycle Bin Items Result Set

The Recycle Bin Items Result Set returns the details about a set of Recycle Bin items. There is no limit on the number of rows that the Recycle Bin Items Result Set can contain. The T-SQL syntax for the result set is as follows.

```

Title                nvarchar(260),
DirName              nvarchar(256),
LeafName             nvarchar(128),
AuthorId             int,
AuthorName           nvarchar(255),
AuthorEmail          nvarchar(255),
DeleteDate           datetime,
{Size}               bigint,
DeleteUserId         int,
DeletedByName        nvarchar(255),
DeletedByEmail       nvarchar(255),
DeleteTransactionId  varbinary(16),
ItemType             tinyint,
OriginalItemType     tinyint,
BinId                tinyint,
{tp_ImageUrl}       nvarchar(255),
ProgId               nvarchar(255);

```

Title: The display name of the Recycle Bin item. Title MUST NOT be NULL.

DirName: The directory name of the Recycle Bin item. DirName MUST NOT be NULL.

LeafName: The leaf name of the Recycle Bin item. LeafName MUST NOT be NULL.

AuthorId: The identifier of the User who is the author of the Recycle Bin item.

AuthorName: The name of the User who is the author of the Recycle Bin item.

AuthorEmail: The e-mail address of the User who is the author of the Recycle Bin item.

DeleteDate: The date when the Recycle Bin item was deleted. DeleteDate MUST NOT be NULL.

{Size}: The size, in Bytes, of the Recycle Bin item. Size MUST NOT be NULL.

DeleteUserId: The user identifier of the User who deleted the Recycle Bin item.

DeletedByName: The login name of the User who deleted the Recycle Bin item.

DeletedByEmail: The e-mail address of the User who deleted the Recycle Bin item.

DeleteTransactionId: The delete transaction identifier of the delete transaction. DeleteTransactionId is a varbinary(16) value. It MUST contain a valid GUID and MUST NOT be NULL.

ItemType: A numerical value indicating the type of the Recycle Bin item. ItemType MUST NOT be NULL and MUST be one of the values in [Delete Item Type](#).

OriginalItemType: A numerical value indicating the type of the original item type for the Recycle Bin item that is the parent item in a cascading delete operation. OriginalItemtype MUST be NULL or one of the values in Delete Item Type.

BinId: A numerical value that indicates which [Recycle Bin Stage](#) the Recycle Bin item is in. BinId MUST NOT be NULL and MUST be 1.

{tp_ImageUrl}: The store-relative URL for the icon of the list if the Recycle Bin item is a list or list item.

ProgId: The file type if the Recycle Bin item is a file.

3.1.5.36 proc_GetSitecollectionBestMatch

The proc_GetSiteCollectionBestMatch stored procedure is called to get the site collection identifier and offset of the first site collection whose path starting with the value in @PathSearch when paths of all site collections' are sorted alphabetically in ascending order.

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

```
PROCEDURE proc_GetSitecollectionBestMatch (  
    @PathSearch          nvarchar(128),  
    @BestMatchSiteId     uniqueidentifier OUTPUT,  
    @BestMatchOffset     int OUTPUT  
) ;
```

@PathSearch: The path of the site collection to search. The path of a site collection is composed of the host header of the site collection and the store-relative URL of the root site in the site collection. The value MUST be as the following table:

Path Value	Description
host header + '/' + store-relative URL	When host header is not NULL and the store-relative URL is not empty
host header	When host header is not NULL and the store-relative URL is empty
'/' + store-relative URL	When host header is NULL

@BestMatchSiteId: The identifier of the first site collection whose path starts with the value of @PathSearch when paths of all site collections are sorted in ascending order.

@BestMatchOffset: The zero-based offset of the first site collection whose path starts with the value of @PathSearch when paths of all site collections are sorted in ascending order.

Return Code Values: This stored procedure MUST return 0 upon completion.

Result Sets: MUST NOT return any result sets.

3.1.5.37 proc_GetSiteCollectionSubset

The proc_GetSiteCollectionSubset stored procedure is called to get a subset of site collection identifiers, paths whose row numbers are greater or equal than @StartRow and less than @StartRow+@PageSize when the names of site collections are sorted alphabetically in the direction specified by the @SortDirection parameter.

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

```
PROCEDURE proc_GetSiteCollectionSubset (
    @PageSize      int,
    @StartRow      int,
    @SortDirection nvarchar(4)
);
```

@PageSize: The size of returned subset. @PageSize MUST be greater than 0.

@StartRow: The row number which starts to get the subset site collections. @StartRow MUST be greater or equal than 0.

@SortDirection: The string which specifies whether the sort direction is in descending order or ascending order. @SortDirection MUST be either "DESC" or "ASC"

Return Code Values: This stored procedure MUST return 0 upon completion.

Result Sets: MUST return Get Site Collection Subset Result Set.

3.1.5.37.1 Get Site Collection Subset Result Set

The Get Site Collection Subset result set returns the site collection identifiers and the paths of the site collections that are specified by @StartRow, @PageSize and @SortDirection. The Get Site Collection Subset result set MUST contains no rows if there is no site collections or no site collections match the condition that the row numbers are greater or equal than @StartRow and less than @StartRow+@PageSize when sorted all the site collections by names with the direction set in @SortDirection.

The T_SQL syntax for the result set is as follows:

```
Id      uniqueidentifier NOT NULL,
Path    nvarchar(385)     NOT NULL,
```

Id: The site collection identifier.

Path: The path of the site collection. The path is composed of the host header of the site collection and the store-relative URL of the root site. The value MUST be as the following table:

Path Value	Description
host header + '/' + store-relative URL	When host header is not NULL and the store-relative URL is not empty

Path Value	Description
host header	When host header is not NULL and the store-relative URL is empty
'/' + store-relative URL	When host header is NULL

3.1.5.38 proc_GetSiteQuota

The proc_GetSiteQuota stored procedure is called to retrieve quota information about a site collection.

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

```
PROCEDURE proc_GetSiteQuota(
    @WebSiteId      uniqueidentifier,
    @RequestGuid     uniqueidentifier = NULL OUTPUT
);
```

@WebSiteId: The site collection identifier of the site collection.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST return the following result set:

3.1.5.38.1 Quota Information Result Set

The Quota Information result set returns information about the quota settings for the specified site collection. If the specified site collection does not exist, Quota Information result set MUST contain no rows. Otherwise, the Quota Information result set MUST contain exactly one row referring to the specified site collection.

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

```
QuotaTemplateId      smallint,
DiskQuota             bigint,
DiskWarning           bigint,
UserQuota             int,
ResourceUsageMaximum float,
ResourceUsageWarning float;
```

QuotaTemplateId: The **quota template identifier** associated with the site collection. QuotaTemplateId MUST be either zero or NULL if the site collection has no quota templates associated to it.

DiskQuota: The disk quota size in Bytes. DiskQuota MUST be zero or NULL if the site collection has no disk quota associated to it.

DiskWarning: The **quota warning level** in Bytes. It MUST be a value between zero and the quota size. DiskWarning MUST be zero or NULL if the site collection has no quota warning level associated to it.

UserQuota: The maximum number of Users. UserQuota MUST be zero or NULL if the site collection has no maximum number of users associated to it. UserQuota can only be larger than zero if the server is in Active Directory account creation mode.

ResourceUsageMaximum: The maximum resource usage value allowed for the specific site collection. The value MUST NOT be NULL.

ResourceUsageWarning: The resource usage warning level for the specific site collection. The value MUST NOT be NULL.

3.1.5.39 **proc_GetSiteUsage**

The `proc_GetSiteUsage` stored procedure is called to get the size on disk and bandwidth that the specified site collection is utilizing.

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

```
PROCEDURE proc_GetSiteUsage (
    @WebSiteId          uniqueidentifier,
    @RequestGuid        uniqueidentifier = NULL OUTPUT
);
```

@WebSiteId: A site collection identifier of the site collection that is being queried.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST return the following result set:

3.1.5.39.1 **Usage Totals Result Set**

The Usage Totals result set returns disk and bandwidth usage for the specified site collection. The Usage Totals result set will be returned always and contain one row.

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

```
DiskUsed    bigint,
BWUsed      bigint;
```

DiskUsed: An integer that specifies the amount of storage, in Bytes, that the content of the site collection is consuming.

BWUsed: An integer that specifies the cumulative bandwidth, in Bytes, that the site collection consumed on the previous day.

3.1.5.40 **proc_GetSizeOfWebPartsOnPage**

The `proc_GetSizeOfWebPartsOnPage` stored procedure is called to get the size, in Bytes, of all the Web Parts and personalization data associated with the Web Parts on a Web Part Page, and the size, in Bytes, of file fragments data associated with this file.

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

```

PROCEDURE proc_GetSizeOfWebPartsOnPage (
    @SiteId          uniqueidentifier,
    @WebId           uniqueidentifier,
    @DirName         nvarchar(256),
    @LeafName        nvarchar(128),
    @UserId          int,
    @RequestGuid     uniqueidentifier = NULL OUTPUT
);

```

@SiteId: The site collection identifier of the site collection.

@WebId: The site identifier of the site.

@DirName: The directory name of the requested document.

@LeafName: The leaf name of the requested document.

@UserId: The user identifier of the User.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST one of the values in the following table.

Value	Description
0	Successful completion.
2	The document does not exist.

Result Sets: MUST return 0 result sets if the document does not exist. Otherwise the following result sets will be returned

3.1.5.40.1 Webparts Size Result Set

The Webparts Size result set MUST return 1 row, which specifies the size, in Bytes, of all the Web Parts and personalization data associated with the Web Parts on a Web Part Page. The T-SQL syntax for the result set is as follows.

```
{WebPartSize}bigint;
```

{WebPartSize}: The size, in Bytes, of all the Web Parts and personalization data associated with the Web Parts on a Web Part Page.

3.1.5.40.2 AllFileFragmentsBlob Size Result Set

The AllFileFragmentsBlob Size Result Set MUST return 1 row, which specify the size, in Bytes, of file fragments data associated with this file. The T-SQL syntax for the result set is as follows.

```
{AllFileFragmentsBlobSize}    bigint;
```

{AllFileFragmentsBlobSize}: The size, in Bytes, of file fragments data associated with this file.

3.1.5.41 **proc_GetSPSiteGuidsGivenHostHeaderPattern**

The `proc_GetSPSiteGuidsGivenHostHeaderPattern` is called to get the site collection identifier of those site collections which have a host header specified to them.

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

```
PROCEDURE proc_GetSPSiteGuidsGivenHostHeaderPattern (  
);
```

Return Code Values: This stored procedure MUST return 0 upon completion.

Result Sets: MUST return the following result set:

3.1.5.41.1 **Site Guids With Host Header Result Set**

The Site Guids With Host Header Result Set MUST contain no rows if no site collection has a host header defined. Otherwise it must contain one row for each site collection with a host header defined. The T-SQL syntax for the result set is

```
Id uniqueidentifier NOT NULL,  
HostHeader nvarchar (128) NOT NULL;
```

Id: The site collection identifier of the site collection.

HostHeader: The host header representation name of the site collection.

3.1.5.42 **proc_GetSPSiteGuidsGivenIdentity**

The `proc_GetSPSiteGuidsGivenIdentity` stored procedure is called to get the site collection identifiers and the full URL associated with a site that has no parent site. Furthermore the site collection itself must have no host header associated with it.

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

```
PROCEDURE proc_GetSPSiteGuidsGivenIdentity (  
);
```

Return Code Values: This stored procedure MUST return 0 upon completion.

Result Sets: MUST return the following result set:

3.1.5.42.1 **Site Guids With Identity Result Set**

The Site Guids With Identity result set MUST contain no rows if all sites collections have a host header associated with them. Otherwise it must contain one row for each site collection whose host header is not defined and that has sites in its collection which have no parent sites. The T-SQL syntax for the result set is

```
FullUrl      nvarchar (256) NOT NULL,  
SiteId       uniqueidentifier NOT NULL;
```

FullUrl: The store-relative URL that represents the name of the site collection.

SiteId: The site collection identifier of the site collection.

3.1.5.43 **proc_GetSPSiteGuidsGivenLockState**

The `proc_GetSPSiteGuidsGivenLockState` stored procedure is called to get the site collection identifiers and the BitFlags of those site collections which have a lock state associated with them. If a **site subscription** identifier is passed then only those site collections belonging to that site subscription will be considered; otherwise all site collections in the farm are considered.

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

```
PROCEDURE proc_GetSPSiteGuidsGivenLockState (  
    @SiteSubscriptionId varbinary(16) = NULL  
) ;
```

@SiteSubscriptionId: The identifier of the Site Subscription to which the site collection is subscribed to.

Return Code Values: This stored procedure MUST return 0 upon completion.

Result Sets: MUST return the following result set:

3.1.5.43.1 **Site Guids With Lock State Result Set**

`proc_GetSPSiteGuidsGivenLockState` returns the site collection identifiers and the BitFlags of the site collections that have a lock state defined. The Site Guids With Lock State result set MUST contain no rows if no site collection has a lock state defined. Otherwise it must contain one row for each site collection in the farm which has a lock state defined for it. The T-SQL syntax for the result set is:

```
Id uniqueidentifier NOT NULL,  
BitFlags int NOT NULL;
```

Id: The site collection identifier of the site collection.

BitFlags: An integer whose specific bits specify the lock state of that site collection. If the bit is set the property is enabled. The specific bits used are as defined in the following table:

Bit number	Property
0	To set/reset Read lock.
1	To set/reset Write lock.
17	To set/reset Read Only lock.

3.1.5.44 **proc_GetSPSiteGuidsGivenOwner**

The `proc_GetSPSiteGuidsGivenOwner` stored procedure is called to get the site collection identifiers and the owner name of the site collections when the input pattern for the name of the owner of the site collection is specified. If a site subscription identifier is passed then only those site collections

belonging to that site subscription will be considered; otherwise all site collections in the farm are considered.

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

```
PROCEDURE proc_GetSPSiteGuidsGivenOwner (  
    @Operator          int,  
    @Pattern           nvarchar(256),  
    @SiteSubscriptionId varbinary(16) = NULL  
);
```

@Operator: A numeric value which determines which comparison operator is to be used for the query. The mapping between numbers and comparison operators is as follows:

Value	Description
1	Use the "Equals" operator in the query.
2	Use the "Not equals" operator in the query.
3	Use the "Like" operator in the query.
4	Use the "Not like" operator in the query.

@Pattern: A string to denote the login name of the of the site collection owner. The string can contain the SQL wild cards "%" and "_".

@SiteSubscription: Gives the identifier of the Site Subscription to which this site collection is subscribed to.

Return Code Values: This stored procedure MUST return 0 upon completion.

Result Sets: MUST return the following result set if @Operator is 1, 2, 3, or 4. MUST NOT return any result sets otherwise.

3.1.5.44.1 Site Guids Given Owner Result Set

proc_GetSPSiteGuidsGivenOwner returns the site collection identifiers and the owner name of the site collections when an owner name is given as the input pattern. The Site Guids Given Owner result set MUST contain no rows if no site collection owner matches the input pattern. Otherwise it must contain one row for each site collection whose owner name matches the input pattern. The T-SQL syntax for the result set is

```
Id uniqueidentifier NOT NULL,  
tp_Login nvarchar(255)
```

Id: The site collection identifier of the site collection

tp_login: The login name of the owner of the site collection.

3.1.5.45 proc_GetSPSiteGuidsGivenSecondaryOwner

The proc_GetSPSiteGuidsGivenSecondaryOwner stored procedure is called to get the site collection identifiers and the secondary owner's login name of the site collections when the input pattern for the name of the secondary owner of the site collection is specified. If a site subscription identifier is

passed then only those site collections belonging to that site subscription will be considered; otherwise all site collections in the farm are considered.

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

```
PROCEDURE proc_GetSPSiteGuidsGivenSecondaryOwner (
    @Operator          int,
    @Pattern            nvarchar(256),
    @SiteSubscriptionId varbinary(16) = NULL
);
```

@Operator: A numeric value which determines which comparison operator is to be used for the query. The mapping between numbers and comparison operators is as follows:

Value	Description
1	Use the "Equals" operator in the query.
2	Use the "Not equals" operator in the query.
3	Use the "Like" operator in the query.
4	Use the "Not like" operator in the query.

@Pattern: A string to denote the login name of the secondary owner of the site collection. The string can contain the SQL wild cards "%" and "_".

@SiteSubscription: Gives the identifier of the Site Subscription to which this site collection is subscribed.

Return Code Values: This stored procedure MUST return 0 upon completion.

Result Sets: MUST return the following result set if @Operator is 1, 2, 3, or 4. MUST NOT return any result sets otherwise.

3.1.5.45.1 Site Guids Given Secondary Owner Result Set

proc_GetSPSiteGuidsGivenSecondaryOwner returns the site collection identifiers and the secondary owner's login name of the site collections whose secondary owner login name matches the input pattern. The Site Guids Given Secondary Owner result set MUST contain no rows if no secondary site collection owner matches the input pattern. Otherwise it must contain one row for each site collection whose secondary owner name matches the input pattern. The T-SQL syntax for the result set is

```
Id uniqueidentifier NOT NULL,
tp_Login nvarchar(255)
```

Id: The site collection identifier of the site collection.

tp_login: The login name of the secondary owner of the site collection.

3.1.5.46 proc_GetSPWebIdentifiersGivenTitle

The proc_GetSPWebIdentifiersGivenTitle is called to get the site identifiers of those sites in a specific site collection whose display name matches the input pattern. Match conditions are specified with an operator as described in the following table.

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

```
PROCEDURE proc_GetSPWebIdentifiersGivenTitle (  
    @Operator    int,  
    @Pattern     nvarchar(256),  
    @SiteGuid    uniqueidentifier  
) ;
```

@Operator: A numeric value which determines which comparison operator is to be used for the query. The mapping between numbers and comparison operators is as follows:

Value	Description
1	Use "Equals" operator in the query.
2	Use the "Not equals" operator in the query.
3	Use the "Like" operator in the query.
4	Use the "Not like" operator in the query.

@Pattern: A string to denote the display name of the site. The string can contain the SQL wild cards "%" and "_".

@SiteGuid: The site collection identifier of the site collection.

Return values: This stored procedure MUST return 0 upon completion.

Result Sets: MUST return the following result set if @Operator is 1, 2, 3, or 4. Otherwise, MUST NOT return any result sets:

3.1.5.46.1 Get SPWeb Identifiers Given Title Result Set

The Get SPWeb Identifiers Given Title Result Set MUST contain no rows if there is no match that can be found. Otherwise it must contain one row for each matching site.

The T-SQL syntax for the result set is

```
FullUrl    nvarchar(256)    NOT NULL,  
Title      nvarchar(255)    NOT NULL
```

FullUrl: The store-relative URL of the site.

Title: The display name of the site.

3.1.5.47 proc_GetTimerLock

The proc_GetTimerLock stored procedure is called to request a **content database lock** for the specified protocol client.

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

```
PROCEDURE proc_GetTimerLock(  
    @ServerId            uniqueidentifier,  
    @LockTimeout         int,  
    @LockStatus          int OUTPUT,  
    @OverrideServerId    uniqueidentifier OUTPUT,  
    @RequestGuid         uniqueidentifier = NULL OUTPUT  
);
```

@ServerId: The configuration object identifier of the protocol client requesting the lock.

@LockTimeout: The maximum age, in minutes, of an existing content database lock before it will be considered expired.

@LockStatus: A Lock Status Type which indicates the status of the requested content database lock. The value is an integer and MUST be one of the following values.

Value	Description
1	Another protocol client already holds the content database lock.
2	Content database lock has been acquired by, or refreshed for, the protocol client requesting the lock.
3	Content database lock acquired by the protocol client requesting the lock. Previous lock had expired.
4	Unexpected failure.

@OverrideServerId: The configuration object identifier of the protocol client that held an expired lock which was overwritten. This output parameter MUST be set if the @LockStatus output parameter is set to "3".

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be one of the following values.

Value	Description
0	Successful completion.
167	An error occurred while retrieving the state of the content database lock.

Result Sets: MUST NOT return any result sets.

3.1.5.48 proc_GetTotalDiscussionsSize

The proc_GetTotalDiscussionsSize stored procedure is called to get the total size of web discussions , in Bytes, for the specified site collection.

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

```
PROCEDURE proc_GetTotalDiscussionsSize (  
    @SiteId            uniqueidentifier,  
    @RequestGuid       uniqueidentifier = NULL OUTPUT
```



```
);
```

@SiteId: The site collection identifier for a site collection.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST return the following result set:

3.1.5.48.1 Discussions Size Result Set

The Discussions Size result set returns the total discussions size, in Bytes, for the site collection. The Discussions Size result set MUST contain one row. The T-SQL syntax for the result set is as follows.

```
{DiscussionsSize}    bigint;
```

{DiscussionsSize}: The total discussions size, in Bytes, of the site collection specified by input parameter @SiteId.

3.1.5.49 proc_GetUniqueScopesInWeb

The proc_GetUniqueScopesInWeb stored procedure is called to get the Access Control List (ACL) information for all the unique security scopes of the **subsites** or lists contained in the specified site.

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

```
PROCEDURE proc_GetUniqueScopesInWeb(  
    @SiteId            uniqueidentifier,  
    @WebId             uniqueidentifier,  
    @SelectSubWebs     bit,  
    @SelectDoclibs     bit,  
    @RequestGuid       uniqueidentifier = NULL OUTPUT  
);
```

@SiteId: The site collection identifier of the site collection.

@WebId: The site identifier of the site.

@SelectSubWebs: A bit specifying whether to get the unique security scopes of the subsites or the lists of the specified site. When set to 1, proc_GetUniqueScopesInWeb MUST return the unique security scopes of the subsites. Otherwise it MUST return the unique security scopes of the lists.

@SelectDoclibs: A bit specifying whether to get the unique security scopes of the document libraries under the specified site. When @SelectSubWebs is set to 1, @SelectDocLibs is ignored. When @SelectDocLibs is set to 0, proc_GetUniqueScopesInWeb MUST return the unique security scopes of all the lists, including document libraries. Otherwise proc_GetUniqueScopesInWeb MUST return the unique security scopes of only document libraries.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: This stored procedure MUST return 0 upon completion.

Result Sets: MUST return the following result set:

3.1.5.49.1 Unique Scopes under Site Result Set

The Unique Scopes under Site result set returns the access control list (ACL) information for all the unique security scopes of subsites or lists contained in a specified site. This result set MUST return a row for each security scope found. The T-SQL syntax for the result set is as follows.

```
ScopeId          uniqueidentifier,  
Acl              image,  
AnonymousPermMask  bigint;
```

ScopeId: The scope identifier of the security scope.

Acl: The binary serialization of the access control list (ACL) of the security scope in the access control list (ACL) Format as described in [\[MS-WSSFO2\]](#), Section [2.2.5.6](#).

AnonymousPermMask: The Rights Mask, as described in [\[MS-WSSFO2\]](#), Section [2.2.3.14](#), that indicates the rights granted to an Anonymous User or a user who has no specific rights in this security scope.

3.1.5.50 proc_GetUserStorageInfo

The proc_GetUserStorageInfo stored procedure is called to get the user identifier, login name and the size, in Bytes, of the personalizations and the users' Web Parts on a particular Web Part Page.

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

```
PROCEDURE proc_GetUserStorageInfo (  
    @SiteId          uniqueidentifier,  
    @WebId           uniqueidentifier,  
    @DirName         nvarchar(256),  
    @LeafName        nvarchar(128),  
    @RequestGuid     uniqueidentifier = NULL OUTPUT  
);
```

@SiteId: The site collection identifier of the site collection that contains the requested document.

@WebId: The site identifier of the site that contains the requested document.

@DirName: The directory name of the requested document.

@LeafName: The leaf name of the requested document.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be one of the following values.

Value	Description
0	Successful completion.
2	The document does not exist.

Result Sets: MUST return 0 result sets if the document does not exist or the following result set when the document exists.

3.1.5.50.1 User Storage Info Result Set

The User Storage Info result set returns the user identifier, login name and the total size, in Bytes, of the personalizations and the users' Web Parts on a particular Web Part Page. There is one row returned for each user that customizes the Web Part Page.

The T-SQL syntax for the result set is defined in the [User Storage Info Result Set](#).

3.1.5.51 proc_GetWebBestMatch

The proc_GetWebBestMatch stored procedure is called to get the site identifier and offset of the first site whose store-relative URL starts with the value in @PathSearch when the store-relative URL of all sites in the site collection are sorted alphabetically in ascending order.

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

```
PROCEDURE proc_GetWebBestMatch (
    @SiteId            uniqueidentifier,
    @PathSearch        nvarchar(128),
    @BestMatchWebId    uniqueidentifier OUTPUT,
    @BestMatchOffset   int OUTPUT,
    @RequestGuid       uniqueidentifier = null OUTPUT
);
```

@SiteId: The site collection identifier of the site collection which contains the sites being searched.

@PathSearch: The store-relative URL of the site to search.

@BestMatchWebId: The site identifier of the first site whose store-relative URL starts with the value of @PathSearch when store-relative URL of all sites in the site collection are sorted in ascending order.

@BestMatchOffset: The zero-based offset of the first site whose store-relative URL starts with the value of @PathSearch when store-relative URL of all sites in the site collection are sorted alphabetically in ascending order.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: This stored procedure MUST return 0 upon completion.

Result Sets: MUST NOT return any result sets.

3.1.5.52 proc_GetWebSubset

The proc_GetWebSubset stored procedure is called to get a subset of sites including site identifiers, store-relative URL and path of the sites. The returned sites have row numbers greater than or equal to @StartRow and less than @StartRow+@PageSize when the names of all sites are sorted alphabetically in the direction specified by the @SortDirection parameter.

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

```
PROCEDURE proc_GetWebSubset (
    @SiteId            uniqueidentifier,
    @PageSize          int,
    @StartRow          int,
    @SortDirection     nvarchar(4)
```

);

@SiteId: The site collection identifier of the site collection that contains the sites.

@PageSize: The total number of returned rows. @PageSize MUST be greater than 0.

@StartRow: The row number which starts to get the subset lists. @StartRow MUST be greater than or equal to 0.

@SortDirection: The string which specifies whether the sort direction is descending order or ascending order. @SortDirection MUST be either "DESC" or "ASC"

Return Code Values: This stored procedure MUST return 0 upon completion.

Result Sets:

This procedure MUST return Get Web Subset Result Set.

3.1.5.52.1 Get Web Subset Result Set

The Get Web Subset result set returns the site identifier, store-relative URL and path of the sites that are specified by @StartRow, @PageSize and @SortDirection. The Get Web Subset result set MUST contains no rows if there is no sites in the site collection or no sites match the condition that the row numbers are greater than or equal to @StartRow and less than @StartRow+@PageSize when names of all lists in the site are sorted in the direction set in @SortDirection.

The T_SQL syntax for the result set is as follows:

```
Id          uniqueidentifier NOT NULL,  
FullUrl     nvarchar(256) NOT NULL,  
Path        nvarchar(385) NOT NULL,
```

Id: The site identifier of the site.

FullUrl: The store-relative URL of the site.

Path: The path of the site. The path is composed of the host header of the site collection and the store-relative URL of the site. The value MUST be as the following table:

Path Value	Description
host header + '/' + store-relative URL	When host header is not NULL and the store-relative URL is not empty
host header	When host header is not NULL and the store-relative URL is empty
'/' + store-relative URL	When host header is NULL

3.1.5.53 proc_MakeExceptionForThrottle

The proc_MakeExceptionForThrottle stored procedure is called to exclude a list from a query size threshold.

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

```

PROCEDURE proc_MakeExceptionForThrottle (
    @WebId          uniqueidentifier,
    @ListId         uniqueidentifier,
    @NoThrottle     bit,
    @RequestGuid    uniqueidentifier = NULL OUTPUT
);

```

@WebId: The site identifier of the site that contains the list.

@ListId: The list identifier of the list that is to be excluded from a query size threshold.

@NoThrottle: A bit flag specifying whether the list should be excluded from a query size threshold. If the flag is set to 0, the list is not excluded from a query size threshold. If the flag is set to 1, the list is excluded from a query size threshold.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: This stored procedure MUST return 0 upon completion.

Result Sets: MUST NOT return any result sets.

3.1.5.54 proc_MoveRecycleBinItemToSecondStage

The `proc_MoveRecycleBinItemToSecondStage` stored procedure is called to move a Recycle Bin item from the first-stage Recycle Bin to the second-stage Recycle Bin.

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

```

PROCEDURE proc_MoveRecycleBinItemToSecondStage(
    @SiteId          uniqueidentifier,
    @WebId           uniqueidentifier,
    @UserId          int,
    @ThresholdRowCount int,
    @DeleteTransactionId varbinary(16),
    @SecondStageRecycleBinQuota int,
    @SpaceRequired    bigint OUTPUT,
    @DiskUsedBefore   bigint= 0 OUTPUT,
    @DiskUsedAfter    bigint= 0 OUTPUT,
    @MaxSSDiskSpace   bigint= 0 OUTPUT,
    @RequestGuid     uniqueidentifier = NULL OUTPUT
);

```

@SiteId: The site collection identifier of the site collection which contains the Recycle Bin item. This parameter MUST NOT be NULL.

@WebId: The site identifier of the site which contains the Recycle Bin item. If the `@UserId` parameter is NOT 0, then this parameter MUST NOT be NULL.

@UserId: A numeric value that MUST either be 0 or the user identifier of the User who deleted the Recycle Bin item. If the value is 0, the store procedure can move the recycle bin item of any user. Otherwise, it can only move the item in the recycle bin of the user specified by `@UserId`. This parameter MUST NOT be NULL.

@ThresholdRowCount: If value is not 0, it specifies the Maximum number of items the container is allowed to have when the recycle item being moved is a container such as a list, folder or folder within a list). If the container has more items than this value, the move operation will fail. If value is

0, there is no limit. The value MUST be 0 or the same as the query size threshold on the specific web application.

@DeleteTransactionId: The delete transaction identifier of the delete transaction. DeleteTransactionId is a varbinary(16) type, but it MUST contain a valid GUID and MUST NOT be NULL.

@SecondStageRecycleBinQuota: The percentage of the site collection quota allocated to the second-stage Recycle Bin. This parameter MUST NOT be NULL and MUST be a positive whole number. If the size of the Recycle Bin item is larger than the size determined by this percentage of the site collection quota, then the Recycle Bin item MUST be permanently deleted. If the size of the Recycle Bin item is less than the size determined by this percentage of the site collection quota, but there is not enough free space within the size determined by this percentage of the site collection quota, then the Recycle Bin item MUST NOT be permanently deleted and MUST NOT be moved to the second-stage Recycle Bin.

@SpaceRequired: A value indicating the amount of additional space, in Bytes, that would be required to store the Recycle Bin item in the second-stage Recycle Bin for the specified @SecondStageRecycleBinQuota parameter and the site collection quota. This parameter is an output parameter.

@DiskUsedBefore: A value indicating the amount of storage space, in Bytes, used by the second-stage Recycle Bin before moving the Recycle Bin item to the second-stage Recycle Bin. This parameter is an output parameter and the default value is 0.

@DiskUsedAfter: A value indicating the amount of storage space, in Bytes, used by the second-stage Recycle Bin after moving the Recycle Bin item to the second-stage Recycle Bin. This parameter is an output parameter and the default value is 0.

@MaxSSDiskSpace: A value indicating the maximum available storage space, in Bytes, of the second-stage Recycle Bin for the specified @SecondStageRecycleBinQuota parameter and the site collection quota. This parameter is an output parameter and the default value is 0.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be one of the following values.

Value	Description
0	Successful completion.
36	The number of items in the container is greater than @ThresholdRowCount
112	There is not enough space. This happens if the @SecondStageRecycleBinQuota parameter is 0 or negative, or if there is not enough remaining space in the specified percentage of the site collection quota to contain the Recycle Bin item.
1168	The Recycle Bin item specified by either the @SiteId and @DeleteTransactionId parameters or the @SiteId, @WebId, @UserId, and @DeleteTransactionId parameters could not be found.
1359	An internal error occurred.

Result Sets: MUST NOT return any result sets.

3.1.5.55 proc_GetStorageMetrics

The proc_GetStorageMetrics stored procedure is called to get the list of storage metrics for a document specified by a URL and its child objects in the content database. Available storage metrics include total size and last modified time. Total size specifies the sum of the size of a document, the size of the document's parts and the size of the document's child objects. Last modified time specifies the last time a document or one of its child objects was modified.

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

```
PROCEDURE proc_GetStorageMetrics (
    @SiteId uniqueidentifier,
    @UserId int,
    @AttachmentsFlag tinyint,
    @ItemLimit int,
    @RootUrl nvarchar(260) = '',
    @OrderBy int = 1,
    @Asc bit = 0,
    @RequestGuid uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The site collection identifier of the site collection containing the document specified by a URL.

@UserId: The user identifier for the current user.

@AttachmentsFlag: An attachments flag value specifying whether the document is, or is contained within, a folder for attachments.

@ItemLimit: This parameter MUST be ignored if set to a value that is less than or equal to 0. Otherwise, if this parameter is greater than 0 and the document specified by the @RootUrl parameter contains a sum total of list items and folders that exceed the value specified by this parameter, the stored procedure MUST fail execution. See Return Values below for more information.

@RootUrl: The URL in store-relative form specifying the container document.

@OrderBy: An integer value specifying the metric to use to order the child documents in the result set. It MUST be either 0 or 1. If @OrderBy is 0, child documents are ordered by TotalSize. If @OrderBy is 1, child documents are ordered by LastModified.

@Asc: A bit value specifying the ordering direction of child documents. If @Asc is 0, child documents are ordered in descending order else child documents are ordered in ascending order.

@RequestGuid: The optional request identifier for the current request.

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

Value	Description
0	Successful execution.
3	The document URL is not valid, or the document is not a folder or site.
36	The stored procedure failed to complete successfully. This value is returned if the @ItemLimit parameter is greater than 0, this list is not exempt from resource throttling operations, and the number of list items combined with the number of folders in the list exceeds the @ItemLimit

Value	Description
	parameter.

This stored procedure MUST return three result sets, as described below, in the following order.

3.1.5.55.1 Individual URL Security Result Set

The Individual URL Security Result Set contains security information about the specified document. If the document does not exist, but the specified document URL is within a list or document library, security information is returned from the document's parent object, such as the list or document library within which it would be contained.

The Individual URL Security Result Set MUST only be returned if the document URL is contained within a list or document library. Otherwise, the NULL Individual URL Security Result Set MUST be returned instead. If returned, the Individual URL Security Result Set MUST contain a single row.

The Individual URL Security Result Set is defined in [\[MS-WSSFO2\]](#) section 3.1.5.46.1 and NULL Individual URL Security Result Set is defined in [\[MS-WSSFO2\]](#) section 2.2.6.14.

3.1.5.55.2 Storage Metrics Result Set

The Storage Metrics result set returns storage metrics information about a document specified by the @RootUrl and its child documents. The first row in the result set contains information for the document specified by the @RootUrl and the remaining rows contain information for the child documents.

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

```
WebId uniqueidentifier,
Id uniqueidentifier,
Type tinyint,
Name nvarchar(128),
DisplayName nvarchar(255),
TotalSize bigint,
LastModified datetime,
Acl varbinary(max),
AnonymousPermMask bigint,
DraftOwnerId int,
Level tinyint,
tp_Flags bigint
```

WebId: The site identifier of the site containing the document.

Id: The document identifier of the document.

Type: An integer identifier specifying the document's **document store type**.

Name: The leaf name of the document location.

DisplayName: The display name for the document. If the Type is site, DisplayName contains the Title of the site. If the Type is folder and the document is the root folder of a list, then DisplayName contains the Title of the list. Otherwise, DisplayName contains the leaf name of the document.

TotalSize: The total size for the sum of the document's size, size of its parts and size of its child objects in Bytes.

LastModified: The last modified time of the document or one of its child objects.

Acl: The binary serialization of the Windows SharePoint Services access control list (ACL) for the document. The Windows SharePoint Services access control list (ACL) is either explicitly defined for the document, or inherited from the parent object of the document. This value can be NULL.

AnonymousPermMask: A Windows SharePoint Services Rights Mask that indicates the permissions granted anonymous users or those users that have no specific permissions to the document. This value can be NULL if anonymous access to the document is not allowed.

DraftOwnerId: The user identifier of the user that published the document as a draft. This value MUST be NULL if the document is not a draft version.

Level: A publishing level type value specifying the publishing status of the document.

tp_Flags: The **list flags** value for the list that contains the document.

3.1.5.55.3 Title Result Set

The Title result set returns a list of URL, Title pairs for the parent site(s) and the list (if exist) of the document specified by the @RootUrl.

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

```
Url nvarchar(260),  
Title nvarchar(255)
```

Url: The Url in store-relative form for a parent site or list.

Title: The title of a parent site or list which is at the location specified by the Url.

3.1.5.56 proc_ProcessStorageMetricsChanges

The proc_ProcessStorageMetricsChanges stored procedure is called to process pending storage metrics changes for site collections. For pending storage metrics of a site collection to get processed, proc_GetStorageMetrics MUST be executed once for the site collection before calling proc_ProcessStorageMetricsChanges if it has not been called in the last 5 days.

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

```
PROCEDURE proc_ProcessStorageMetricsChanges (  
    @StartingSiteId uniqueidentifier,  
    @ExecutionDuration int,  
    @LastProcessedSiteId uniqueidentifier OUTPUT  
) ;
```

@StartingSiteId: The site identifier for the site collection to start processing at.

@ExecutionDuration: The maximum duration in seconds to process pending changes.

@LastProcessedSiteId: The site identifier for the last site collection processed. This value MUST be set to **empty GUID** if no site collections were processed.

Return Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.5.57 **proc_QMChangeSiteDiskUsedAndContentTimestamp**

The `proc_QMChangeSiteDiskUsedAndContentTimeStam`p stored procedure is called to update the disk-used size of a site collection.

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

```
PROCEDURE proc_QMChangeSiteDiskUsedAndContentTimestamp (
    @SiteId                uniqueidentifier,
    @cbDelta                bigint,
    @fIncrementTimestamp    bit,
    @DocId                 uniqueidentifier
);
```

@SiteId: The site collection identifier of the site collection to update.

@cbDelta: The change in the disk-used size in bytes.

@fIncrementTimestamp: A bit flag specifying whether to update the time stamp of `LastContentChange` of the site collection as described in [\[MS-WSSCCSP2\]](#), section 3.1.4.20.1. If the value is 1, `LastContentChange` of the site collection MUST be updated with the current time. If the value is 0, `LastContentChange` MUST be unmodified.

@DocId: The document identifier of the document to update storage metrics. If `@DocId` is NULL, storage metrics MUST not be updated.

Return Code Values: This stored procedure MUST return 0 upon completion .

Result Sets: MUST NOT return any result sets.

3.1.5.58 **proc_QMGetDiskWarning**

The `proc_QMGetDiskWarning` stored procedure is called to get a list of site collections whose disk-used size, in Bytes, is larger than the quota warning level.

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

```
PROCEDURE proc_QMGetDiskWarning(
    @dtLast                datetime OUTPUT,
    @dtCur                 datetime OUTPUT,
    @RequestGuid            uniqueidentifier = NULL OUTPUT
);
```

@dtLast: An output parameter that contains the date and time when the last **quota warning** was sent.

@dtCur: An output parameter that contains the current date and time.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: This stored procedure MUST return 0 upon completion.

Result Sets: MUST return the following result set:

3.1.5.58.1 Disk Warning Result Set

The Disk Warning result set returns the information needed for the quota warnings to be sent out. The Disk Warning result set contains information about the top-level sites whose size is larger than the quota warning level, the e-mail addresses of the corresponding site collection administrators, the URLs of the sites, and their language code identifiers (LCIDs). The Disk Warning result set **MUST** return one row for every combination of top-level site and site collection administrator for that site collection, or no rows if there are no top-level sites. The T-SQL syntax for the result set is as follows.

```
FullUrl      nvarchar(256),
Id           uniqueidentifier,
tp_Email     nvarchar(255),
Language     int;
```

FullUrl: The store-relative URL of the site collection.

Id: The site collection identifier of the site collection.

tp_Email: The e-mail address of the site collection administrator.

Language: The LCID which indicates the language of the site collection.

3.1.5.59 proc_QMMarkDiskWarning

The proc_QMMarkDiskWarning stored procedure is called to update the stored date and time of the last quota warning.

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

```
PROCEDURE proc_QMMarkDiskWarning(
    @dtLast      datetime,
    @dtCur       datetime,
    @RequestGuid uniqueidentifier = NULL OUTPUT
);
```

@dtLast: The date and time when the last quota warning was sent.

@dtCur: The current date and time.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: This stored procedure **MUST** return 0 upon completion.

Result Sets: **MUST NOT** return any result sets.

3.1.5.60 proc_RestoreRecycleBinItem

The proc_RestoreRecycleBinItem stored procedure is called to restore a Recycle Bin item.

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

```
PROCEDURE proc_RestoreRecycleBinItem(
    @SiteId      uniqueidentifier,
    @WebId       uniqueidentifier,
```

```

@UserId          int,
@ThresholdRowCount int,
@TranLockStatus  tinyint,
@DeleteTransactionId varbinary(16),
@FailedDirName    nvarchar(256) = NULL OUTPUT,
@RequestGuid      uniqueidentifier = NULL OUTPUT
);

```

@SiteId: The site collection identifier of the site collection that contains the Recycle Bin item. This parameter MUST NOT be NULL.

@WebId: The site identifier of the site that contains the Recycle Bin item. If the @UserId parameter is NOT 0, then this parameter MUST NOT be NULL.

@UserId: A numeric value that MUST either be 0 or the user identifier of the User who deleted the Recycle Bin item. If the value is 0, the restore is issued by an administrator. This parameter MUST NOT be NULL.

@ThresholdRowCount: If value is not 0, it specifies the Maximum number of items the container is allowed to have when the recycle item being restored is a container such as a list, folder or folder within a list). If the container has more items than this value, the restore operation will fail. If value is 0, there is no limit. The value MUST be 0 or the same as the query size threshold on the specific web application.

@TranLockStatus: The value that specifies whether a short-term **transaction application lock** was taken for the specified Site. This parameter MUST specify a valid value from the following table:

Value	Description
1	<p>The short-term transaction application lock is not needed. This MUST be set when either of the following conditions are true:</p> <ul style="list-style-type: none"> ▪ The Recycle Bin Item is a document version. ▪ The Recycle Bin Item is a list. ▪ The Recycle Bin Item is a folder with lists. ▪ The Recycle Bin Item is an attachment. ▪ The Recycle Bin Item is a document, list item or folder and the following are true: <ul style="list-style-type: none"> ▪ The list containing the Item has no relationship lookup fields with cascading behavior or restrict behavior. ▪ There is no relationship lookup field which has the list containing the item as the target list and has cascading behavior or restrict behavior.
2	<p>The short-term transaction application lock is needed. This MUST be set when either of the following conditions are true:</p> <ul style="list-style-type: none"> ▪ The Recycle Bin Item is a parent item of a cascading delete operation. ▪ The Recycle Bin Item is a document, list item or folder and the following are true: <ul style="list-style-type: none"> ▪ The list containing the item has at least one relationship lookup field with cascading

Value	Description
	<p>behavior or restrict behavior.</p> <ul style="list-style-type: none"> There is at least one relationship lookup field which has the list containing the item as the target list and has cascading behavior or restrict behavior.

@DeleteTransactionId: The delete transaction identifier of the delete transaction. DeleteTransactionId is a varbinary(16) type. it MUST contain a valid GUID and MUST NOT be NULL.

@FailedDirName: An output parameter contains the dependent directory indicated by the item's **DirName** that cannot be found. The default value is NULL. NULL means there is no dependent directory. This output parameter is only set if the dependent directory exists.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be one of the following values.

Value	Description
0	Successful completion.
3	The Recycle Bin item could not be restored because the parent folder of the Recycle Bin item, as indicated by the item's DirName , could not be found.
36	The number of items in the container is greater than @ThresholdRowCount
80	<p>The Recycle Bin item could not be restored for one of the following reasons:</p> <ul style="list-style-type: none"> The Recycle Bin item is a list item in a survey list that doesn't allow multiple votes and another vote is already present for the User. The Recycle Bin item is a file, list, folder, or attachment and another document with the same path exists. The Recycle Bin item is a list and another list with the same display name exists. The Recycle Bin item is a folder which contains lists and one or more lists exist with the same display name as one of the lists in that folder.
87	The Recycle Bin item is a file or an attachment and could not be restored because an error occurred recreating the image or thumbnail for the Recycle Bin item.
212	The site collection is locked and the Recycle Bin item cannot be restored.
301	@TranLockStatus is set to 1 but the restore operation requires short-term transaction application lock
1168	Either the Recycle Bin item could not be found or the site that contains the Recycle Bin item could not be found.
1359	An internal error occurred.
1816	The site collection does not have enough remaining quota available to restore the Recycle Bin item.
1979	The Recycle Bin item could not be restored for one of the following reasons:

Value	Description
	<ul style="list-style-type: none"> ▪ The list that contains the Recycle Bin item could not be found. ▪ The Recycle Bin item is a file version, but the corresponding file could not be found. ▪ The Recycle Bin item is an attachment or list item version, but the corresponding list item could not be found.

Result Sets: MUST NOT return any result sets.

3.1.5.61 **proc_RevertDocContentStreams**

The `proc_RevertDocContentStreams` stored procedure is called to revert a customized document to a previous uncustomized state.

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

```

PROCEDURE proc_RevertDocContentStreams (
    @SiteId          uniqueidentifier,
    @WebId            uniqueidentifier,
    @WebFullUrl       nvarchar(260),
    @DocDirName       nvarchar(256),
    @DocLeafName      nvarchar(128),
    @UserId           int,
    @RequestGuid      uniqueidentifier = NULL OUTPUT
);

```

@SiteId: The site collection identifier of the site collection that contains the requested document.

@WebId: The site identifier of the site that contains the requested document.

@WebFullUrl: This parameter MUST be ignored.

@DocDirName: The directory name of the requested document.

@DocLeafName: The leaf name of the requested document.

@UserId: The user identifier of the User calling the stored procedure.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be one of the following values.

Value	Description
0	Successful completion.
2	The specified document was NOT found.

Result Sets: MUST NOT return any result sets.

3.1.5.62 proc_ScorchList

The proc_ScorchList stored procedure is called to delete a list and all its contents if the list belongs to one of the following three categories:

- Lists with no parent site
- Lists with documents that have no parent list
- Lists with list items that have no parent list.

It is not used on any other type of list. To delete a list that does not match one of the three criteria earlier in this section, use proc_DeleteUrl, as defined in [\[MS-WSSFO2\]](#) section 3.1.4.13, instead.

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

```
PROCEDURE proc_ScorchList(  
    @ListId            uniqueidentifier,  
    @RequestGuid       uniqueidentifier = NULL OUTPUT  
);
```

@ListId: The list identifier of the list to be deleted.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be one of the following values.

Value	Description
0	Successful completion.
3	The list does not exist.
8398	A list marked as undeletable cannot be deleted.

Result Sets: MUST NOT return any result sets.

3.1.5.63 proc_ScorchWeb

The proc_ScorchWeb stored procedure is called to delete a site and its contents with no parent site or site collection. To delete a site that does not meet this criterion, use proc_DeleteWeb, as defined in [\[MS-WSSDLIM2\]](#) section 3.1.4.35 instead.

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

```
PROCEDURE proc_ScorchWeb (  
    @WebId            uniqueidentifier,  
    @ThresholdRowCount int,  
    @RequestGuid       uniqueidentifier = NULL OUTPUT  
);
```

@WebId: The site identifier of the site to be deleted.

@ThresholdRowCount: Specifies the maximum number of items the site is allowed to have when this stored procedure is called. If the site has more items than this value, the operation will fail. If value is 0, there is no limit.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be one of the following values.

Value	Description
0	Successful completion.
3	The site does not exist.
36	The number of items in the site is greater than @ThresholdRowCount

Result Sets: MUST NOT return any result sets if the site does not exist. Otherwise, it MUST return the following result set.

3.1.5.63.1 Audit Mask Result Set

The Audit Mask result set returns audit configuration information. The Audit Mask result set MUST be returned with one row of audit configuration information upon successful completion.

The Audit Mask result set is specified in [\[MS-WSSFO2\]](#) section 3.1.4.20.13.

3.1.5.64 proc_SecBackupAllWebMembers

The proc_SecBackupAllWebMembers stored procedure is called to return information about All Site Members.

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

```
PROCEDURE proc_SecBackupAllWebMembers (
    @SiteId          uniqueidentifier,
    @WebId           uniqueidentifier,
    @RequestGuid     uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The site collection identifier that contains the site.

@WebId: The site identifier whose All Site Members are to be returned.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: This stored procedure MUST return 0 upon completion.

Result Sets: MUST return the following result set:

3.1.5.64.1 UserInfo Result Set

The UserInfo result set returns information about All Site Members. The result set contains one row per non-deleted User. The T-SQL syntax for the result set is as follows.

```
tp_Id          int,
```



```

tp_Title      nvarchar(255),
tp_Login      nvarchar(255),
tp_Email      nvarchar(255),
tp_Notes      nvarchar(1023),
tp_Deleted    int;

```

tp_Id: The user identifier of the user.

tp_Title: The display name of the user.

tp_Login: The login name of the user.

tp_Email: The e-mail address of the user.

tp_Notes: A string that contains information about the user.

tp_Deleted: An integer value indicating whether the user has been deleted from the site collection. This value MUST be 0.

3.1.5.65 **proc_SecGetListItemSecurity**

The `proc_SecGetListItemSecurity` stored procedure is called to get the security information of a list item.

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

```

PROCEDURE proc_SecGetListItemSecurity(
    @SiteId          uniqueidentifier,
    @ListId          uniqueidentifier,
    @ItemId          int,
    @RequestGuid     uniqueidentifier = NULL OUTPUT
);

```

@SiteId: The site collection identifier of the site collection that contains the requested list item.

@ListId: The list identifier of the list that contains the requested list item.

@ItemId: The item identifier of the requested list item.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: This stored procedure MUST return 0 upon completion.

Result Sets: MUST return the following result set.

3.1.5.65.1 **Access Control List Result Set**

The Access Control List result set returns access control list (ACL) information of the list item specified by the input parameters. The Access Control List result set MUST contain one row if the list item exists. If the list item does not exist, the result set MUST contain 0 rows. The T-SQL syntax for the result set is as follows.

```

Acl          image,
AnonymousPermMask  bigint;

```

Acl: The binary serialization of the access control list (ACL) in effect for the list item, in the format described in [\[MS-WSSFO2\]](#), Section [2.2.5.6](#).

AnonymousPermMask: The rights mask, as described in [\[MS-WSSFO2\]](#), Section [2.2.3.14](#), that indicates the rights granted to any Anonymous User for this list item.

3.1.5.66 **proc_SecRemoveExternalSecurityProvider**

The `proc_SecRemoveExternalSecurityProvider` stored procedure is called to remove the External Security Provider that enforces user authorization on the site.

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

```
PROCEDURE proc_SecRemoveExternalSecurityProvider(  
    @SiteId          uniqueidentifier,  
    @WebId           uniqueidentifier,  
    @RequestGuid     uniqueidentifier = NULL OUTPUT  
);
```

@SiteId: The site collection identifier of the site collection that contains the specified site. `@SiteId` MUST NOT be NULL.

@WebId: The site identifier of the site for which the External Security Provider will be removed. `@WebId` MUST NOT be NULL.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.5.67 **proc_SetAuditMask**

The `proc_SetAuditMask` stored procedure is called to set the audit flags, as specified in [\[MS-WSSFO2\]](#) Section [2.2.3.1](#), directly applicable to an object.

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

```
PROCEDURE proc_SetAuditMask(  
    @ItemType        tinyint,  
    @SiteId          uniqueidentifier,  
    @DirName         nvarchar(256),  
    @LeafName        nvarchar(128),  
    @AuditFlags      int,  
    @RequestGuid     uniqueidentifier = NULL OUTPUT  
);
```

@ItemType: The type of object which MUST be a value from the Audit Item Type enumeration specified in [\[MS-WSSFO2\]](#) Section [2.2.4.2](#).

@SiteId: The site collection identifier of the site collection which contains the object for which the audit flags are being requested.

@DirName: The directory name of the object.

@LeafName: The leaf name of the object.

@AuditFlags: The audit flags directly applicable to the object, which MUST be an audit flags bit mask as specified in [MS-WSSFO2] Section [2.2.3.1](#).

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be one of the following values.

Value	Description
0	Successful completion.
2	If the object identified by the specified @ItemType, @SiteId, @DirName and @LeafName parameters does not exist, the protocol server MUST return 2.

Result Sets: MUST NOT return any result sets.

3.1.5.68 proc_SetDeadWebNotificationCount

The proc_SetDeadWebNotificationCount stored procedure is called to set a value of notify count for the specified site collection and to log the update event.

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

```
PROCEDURE proc_SetDeadWebNotificationCount (
    @SiteId          uniqueidentifier,
    @NotifyCount      smallint,
    @RequestGuid      uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The site collection identifier of the site collection. This value MUST NOT be NULL.

@NotifyCount: The value of notify count. This value MUST NOT be NULL.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.5.69 proc_SetListRequestAccess

The proc_SetListRequestAccess stored procedure is called to update the request access setting of the list specified by the @ListId parameter. When the parameter @RequestAccess is set to 1 or NULL, access requests are enabled and a user that gets access denied can then submit a request to access the list. When the parameter @RequestAccess is set to 0, access requests are disabled and a user that gets access denied will not be able to request access to the list.

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

```
PROCEDURE proc_SetListRequestAccess(
    @SiteId          uniqueidentifier,
    @WebId           uniqueidentifier,
    @ListId          uniqueidentifier,
    @RequestAccess   bit,
```

```

        @RequestGuid    uniqueidentifier = NULL OUTPUT
    );

```

@SiteId: The site collection identifier of the site collection that contains the specified list.

@WebId: The site identifier of the site that contains the specified list.

@ListId: The list identifier of the list.

@RequestAccess: A bit flag specifying whether the request access setting of the list is enabled or disabled. If the flag is set to 0, the list does not allow access requests. If the flag is set to 1 or NULL, the list allows access requests.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: This stored procedure MUST return 0 upon completion.

Result Sets: MUST NOT return any result sets.

3.1.5.70 **proc_SetSiteQuota**

The proc_SetSiteQuota stored procedure is called to set quota settings on a specified site collection.

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

```

PROCEDURE proc_SetSiteQuota(
    @WebSiteId          uniqueidentifier,
    @quotaId            smallint,
    @diskQuota          bigint,
    @diskWarning        bigint,
    @userQuota          int,
    @resourceUsageMaximum float,
    @resourceUsageWarning float,
    @RequestGuid        uniqueidentifier = NULL OUTPUT
);

```

@WebSiteId: The site collection identifier of the site collection. This MUST NOT be NULL.

@quotaId: The value that specifies the quota template associated with the site collection. This can be NULL if the site collection is not to be associated to any quota template. When this is not NULL, it MUST be the identifier of the quota template.

@diskQuota: The quota size of the site collection in Bytes. This can be NULL if no quota is wanted for the site collection.

@diskWarning: The disk space, in Bytes, that will trigger a warning message. This MUST be a value between 0 and @diskQuota. This can be NULL if no warning message is wanted for the site collection.

@userQuota: The maximum number of Users allowed for the site collection or NULL if no maximum number is defined. This MUST be NULL in **domain account mode**. In Active Directory account creation mode, @userQuota can be either the maximum number of Users allowed or NULL.

@resourceUsageMaximum: The maximum resource usage value allowed for the specific site collection.

@resourceUsageWarning: The resource usage warning level for the specific site collection.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.5.71 **proc_SetSubscription**

The `proc_SetSubscription` stored procedure is called to set the site subscription identifier of a site collection.

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

```
PROCEDURE proc_SetSubscription (  
    @SiteId            uniqueidentifier,  
    @SubscriptionId    varbinary(16),  
    @RequestGuid       uniqueidentifier = NULL OUTPUT  
);
```

@SiteId: The site collection identifier of the site collection.

@SubscriptionId: The identifier of the site subscription.

@RequestGuid: The optional request identifier for the current request.

Return values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.5.72 **proc_SiteCollectionExists**

The `proc_SiteCollectionExists` stored procedure is called to determine if the content database contains a site collection with a given site collection identifier.

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

```
PROCEDURE proc_SiteCollectionExists (  
    @SiteId            uniqueidentifier  
);
```

@SiteId: The site collection identifier of the site collection for which the content database will be searched.

Return Code Values: An integer which MUST be one of the following values.

Value	Description
0	The content database does not contain a site collection with a site collection identifier equal to @SiteId .
1	The content database contains a site collection with a site collection identifier equal to @SiteId .

3.1.5.73 proc_SizeOfPersonalizationsPerUser

The proc_SizeOfPersonalizationsPerUser stored procedure is called to get the user identifier, login name, and the total size, in bytes, of the personalization and Web Parts on a particular Web Part Page.

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

```
PROCEDURE proc_SizeOfPersonalizationsPerUser (  
    @SiteId          uniqueidentifier,  
    @DocId           uniqueidentifier,  
    @RequestGuid     uniqueidentifier = NULL OUTPUT  
);
```

@SiteId: The site collection identifier of the site collection that contains the requested document.

@DocId: The document identifier of the document.

@RequestGuid: The optional request identifier for the current request.

Return Code Values: This stored procedure MUST return 0 upon completion.

Result Sets: MUST return the following result set:

3.1.5.73.1 User Storage Info Result Set

The User Storage Info result set returns the user identifier, login name and the total size, in bytes, of the personalizations and Web Parts on a particular Web Part Page. There is 1 row returned for each user that customizes the Web Part Page.

The T-SQL syntax for the result set is defined in the [User Storage Info Result Set](#).

3.1.5.74 proc_TrimAuditEntries

The proc_TrimAuditEntries stored procedure is called to delete audit entries for a site collection that occurred prior to a given date.

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

```
PROCEDURE proc_TrimAuditEntries (  
    @SiteId          uniqueidentifier,  
    @UserId          int,  
    @EndDate         datetime,  
    @EntriesDeleted  bigint OUTPUT,  
);
```

@SiteId: The site collection identifier of the site collection.

@UserId: The user identifier of the user who performed the trim operation.

@EndDate: The end time condition for the audit entries to be deleted, in Coordinated Universal Time (UTC).

@EntriesDeleted: The number of entries deleted.

Return values:

Value	Description
0	Successful completion.
2	The site collection does not exist.

Result Sets: MUST NOT return any result sets.

3.1.5.75 **proc_UpdateDiskUsed**

The `proc_UpdateDiskUsed` stored procedure is called to update the disk-used size, in Bytes, of a site collection.

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

```
PROCEDURE proc_UpdateDiskUsed (
    @SiteId                uniqueidentifier,
    @bUpdateTimeStampForce bit = 0
);
```

@SiteId: The site collection identifier of the site collection whose disk-used size is to update.

@bUpdateTimeStampForce: A bit flag specifying whether to update the time stamp of `LastContentChange` of the site collection as described in [\[MS-WSSCCSP2\]](#), section 3.1.4.20.1. If the value is 1, `LastContentChange` of the site collection MUST be updated with the current time. If the value is 0, `LastContentChange` MUST be unmodified.

Return Code Values: This stored procedure MUST return 0 upon completion.

Result Sets: MUST NOT return any result sets.

3.1.5.76 **proc_UpdateStatistics**

The `proc_UpdateStatistics` stored procedure is called to update statistics that are used internally by the back-end database server on user tables in the content database. Statistics are used by the database server to optimize query performance.

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

```
PROCEDURE proc_UpdateStatistics();
```

`proc_UpdateStatistics` MUST NOT take any parameters.

Return Code Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.1.5.77 **proc_GetDatabaseInformation**

The `proc_GetDatabaseInformation` stored procedure is called to retrieve a specific property which is stored in the database, and referenced by a name.

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

```

PROCEDURE proc_GetDatabaseInformation (
    @Name      nvarchar(128)
);

```

@Name: The name of the property to be retrieved.

Return Code Values: This stored procedure MUST return 0 upon completion.

Result Sets: MUST return the following result set:

3.1.5.77.1 Database Information Result Set

The Database Information result set returns Value, in nvarchar(1023), corresponding to the value of the @Name. There is one row returned if @Name is defined, otherwise zero rows are returned.

3.1.5.78 proc_SetDatabaseInformation

The proc_SetDatabaseInformation stored procedure is called to update a specific property which is stored in the database and referenced by a name. If the property does not exist, then the proc_SetDatabaseInformation stored procedure creates the property and sets its value.

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

```

PROCEDURE proc_SetDatabaseInformation (
    @Name      nvarchar(128),
    @Value     nvarchar(1023)
);

```

@Name: The name of the property to be saved.

@Value: The value of the property to be saved.

Return Code Values: This stored procedure MUST return 0 upon completion.

Result Sets: MUST NOT return any result sets.

3.1.6 Timer Events

If the execution timeout event is triggered, the execution of the stored procedure is terminated and the call fails

3.1.7 Other Local Events

None.

3.2 Front-end Web Server Client Details

The front-end Web server acts as a client when it calls the back-end database server to request execution of stored procedures.

3.2.1 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.

The front-end Web server can maintain the following sets of data for this protocol within object structures. There is no requirement for the data within these structures to be a complete representation of all data maintained on the back-end database server. These structures can be populated as various requests to the back-end database server are fulfilled. Data maintained on the front-end Web server can be discarded after individual sequences of requests have finished as part of a response for a higher level event.

- Configuration
- Site Collections
- Sites
- Lists
- List Items
- Documents
- Users
- Groups

3.2.2 Timers

A connection timeout timer is set up on the front-end Web server to govern the total connection time for any requests to the back-end database server. The amount of time is governed by a timeout value configured on the front-end Web server for all back-end database server connections.

3.2.3 Initialization

The front-end Web server **MUST** validate the user making the request before calling the stored procedures. The site collection identifier and the user identifier for the user making the request are looked up by the front-end Web server before calling additional stored procedures.

3.2.4 Message Processing Events and Sequencing Rules

The front-end Web server handles each stored procedure with the same processing method of calling the stored procedure and waiting for the return code and any result sets that will be returned.

The front-end Web server can execute dynamically generated SQL queries against the stored procedures or the tables and views used within the database. Unless otherwise specified, any data addition, removal, or modification **MUST** occur only by calling the listed stored procedures. SQL queries **MUST NOT** attempt to add, remove, or update data in any table or view in the content database or configuration database, unless explicitly described in this section.

3.2.5 Timer Events

If the connection timeout event is triggered, the connection and the stored procedure call fails.

3.2.6 Other Local Events

None

4 Protocol Examples

This section provides specific example scenarios for end-to-end data query and update comments as part of file, user, and group administration operations. These examples describe in detail the process of communication between the various server components.

4.1 Auditing Operations

This example describes the requests a protocol client makes to delete audit entries from a site collection that occurred prior to a given date.

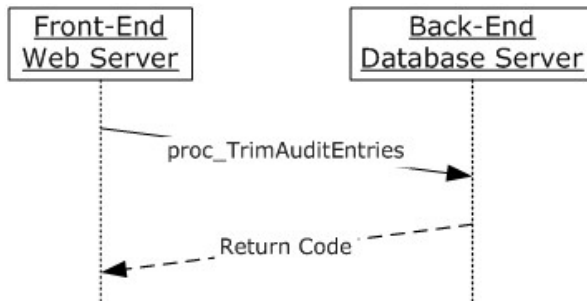


Figure 9: Deleting audit entries for a site collection

The protocol client sends the `proc_TrimAuditEntries` T-SQL message to the protocol server. The protocol server removes the audit entries prior to the date specified and then returns either success or failure. The protocol server also returns the number of entries in the audit log which were deleted.

4.2 Quota Management Operations

This section provides example of quota management operations.

4.2.1 Querying Quota

This example describes the requests made and responses returned when a User retrieves the quota information for a site collection.

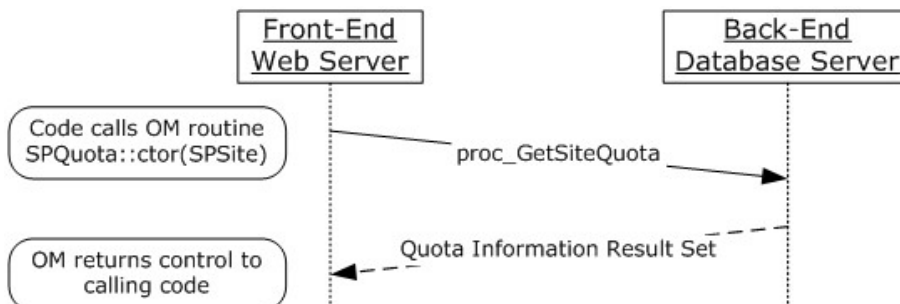


Figure 10: Retrieving Quota Information for a Site Collection

This scenario is initiated when a user requests quota information for a site collection.

For simplicity's sake the example assumes that:

1. The code has already instantiated the required site collection object
2. The code has already instantiated the quota template object if the site collection uses a quota template to set its quota.

The following actions happen:

1. The front-end Web server requests for the quota information of a site by calling the stored procedure `proc_GetSiteQuota`.
2. The stored procedure returns the [Quota Information Result Set](#).
3. The front-end Web server returns to the user values from the Quota Information Result Set.

4.2.2 Updating Quota

This example describes the requests made and responses returned when a User requests to update the quota information for a site collection. The quota for a site collection can be set either from a quota template or the values can be set directly.

4.2.2.1 Setting Quota from a Quota Template

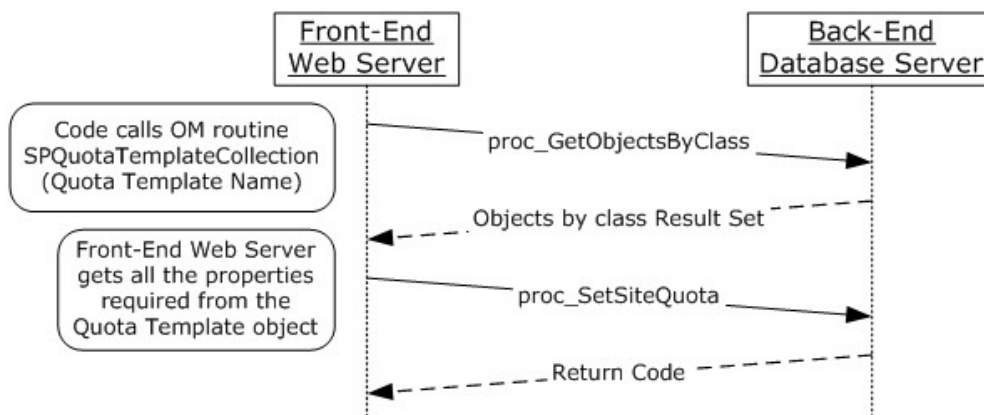


Figure 11: Setting a Quota Using a Quota Template

The following actions happen:

1. The front-end Web server requests for all the quota templates available by calling the stored procedure `proc_GetObjectsByClass` as described in [\[MS-WSSFO2\]](#) section 3.1.5.37.
2. When a User selects a particular quota template to be applied, the front-end Web server constructs the quota template object from the serialized data.
3. The front-end Web server gets the information about the allowed disk space, disk space warning limit, and the maximum number of users allowed if in Active Directory account creation mode from the quota template object.
4. The front-end Web server updates the quota for the site collection using a call to the `proc_SetSiteQuota` stored procedure specifying the quota template identifier, disk space, disk space warning limit, and maximum number of users allowed as parameters.

4.2.3 Get Usage Information for a Site Collection

The front-end Web server calls the `proc_GetSiteUsage` stored procedure to get the usage information for a site collection.

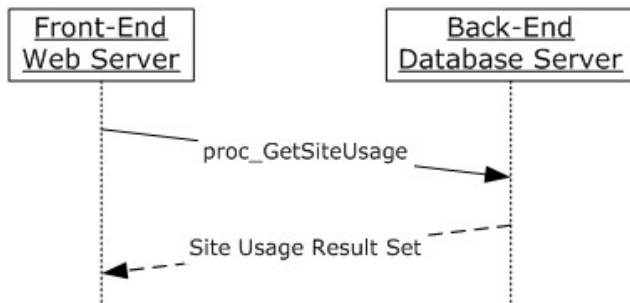


Figure 12: Retrieving Usage Information for a Site Collection

The actions that happen are:

1. The front-end Web server calls the `proc_GetSiteUsage` stored procedure on a content database.
2. The back-end database server returns a Usage Totals Result Set, as defined in section [3.1.5.39.1](#).

4.2.4 Warning Site Collections Which Are Near the Allowed Disk Space

This example describes the actions that the front-end Web server performs during a timer job to get information about site collections that are near the limits set by a quota and the owners that need to be warned.

The following diagram shows the sequence of calls the front-end Web server performs for quota warning operations.

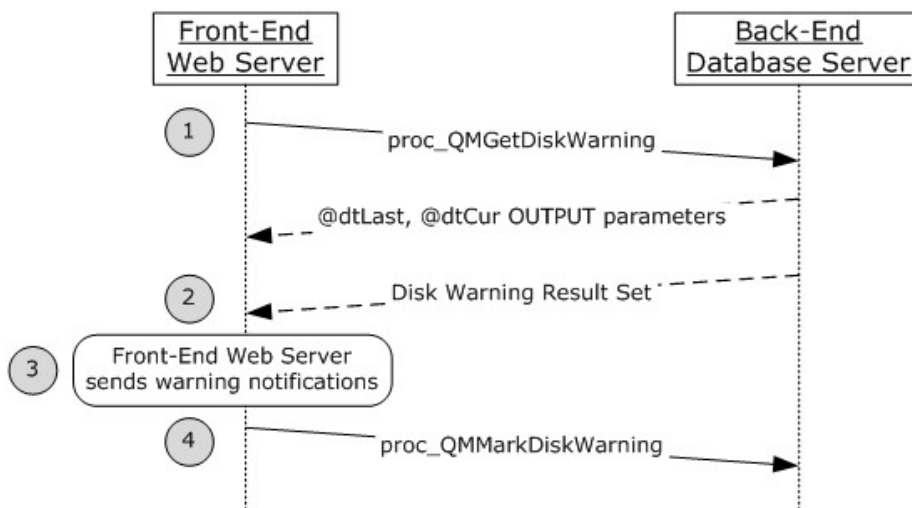


Figure 13: Warning Site Collections near a Quota Limit

The actions that happen are:

1. The front-end Web server calls the `proc_QMGetDiskWarning` stored procedure on a content database.
2. The back-end database server returns the last time when a quota warning was sent, the current time, and the list of all site collections along with the e-mail addresses of their site collection administrators who have crossed the quota warning limits and need to be warned.
3. The front-end Web server sends the notifications to the site collection administrators of the site collections.
4. The front-end Web server calls the `proc_QMMarkDiskWarning` indicating to the back-end database server that the notifications have been sent for the site collections.

4.3 Recycle Bin Operations

This section provides examples of recycle bin operations.

4.3.1 Query Items in First-Stage Recycle Bin

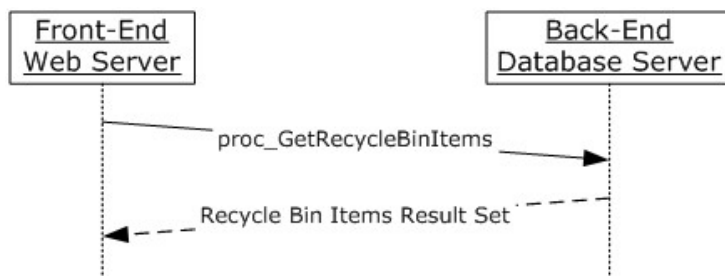


Figure 14: Retrieve Items in the First-Stage Recycle Bin

This example describes the actions the front-end Web server takes when querying items in the first-stage Recycle Bin. The actions that happen are:

1. The front-end Web server calls the `proc_GetRecycleBinItems` stored procedure on a content database.
2. The back-end database server returns a Recycle Bin items result set, as defined in [Recycle Bin Items Result Set](#).

4.3.2 Delete a First-Stage Recycle Bin Item to Second-Stage Recycle Bin

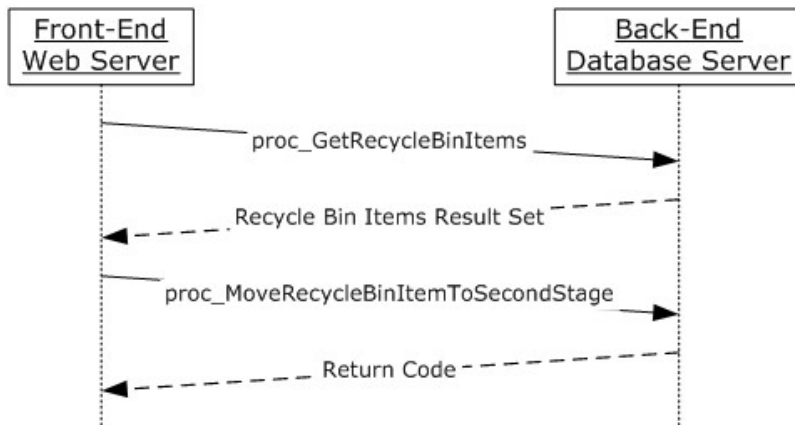


Figure 15: Move an Item from a First-stage Recycle Bin to a Second-stage Recycle Bin

This example describes the actions a front-end Web server takes when querying and deleting an item in the first-stage Recycle Bin. The actions that happen are:

1. The front-end Web server calls the `proc_GetRecycleBinItems` stored procedure on a content database.
2. The back-end database server returns a Recycle Bin Items result set as defined in [Recycle Bin Items Result Set](#).
3. The front-end Web server chooses an item from the result set to delete and calls `proc_MoveRecycleBinItemToSecondStage` with appropriate parameters.
4. The back-end database server returns an error code indicating if the operation succeeded or not.

4.3.3 Restore a First-Stage Recycle Bin Item

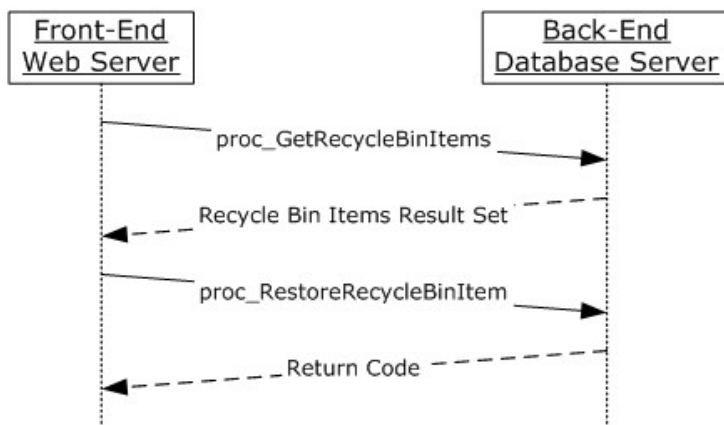


Figure 16: Restore an item from the First-stage Recycle Bin

This example describes the actions the front-end Web server takes when querying and restoring an item in the first-stage Recycle Bin to a list. The actions that happen are:

1. The front-end Web server calls the `proc_GetRecycleBinItems` stored procedure on a content database to enumerate items in the first-stage Recycle Bin.
2. The back-end database server returns a Recycle Bin Items result set as defined in [Recycle Bin Items Result Set](#).
3. The front-end Web server chooses an item from the result set to restore and calls `proc_RestoreRecycleBinItem` with appropriate parameters.
4. The back-end database server returns an error code indicating if the operation succeeded or not.

4.3.4 Delete a Second-Stage Recycle Bin Item

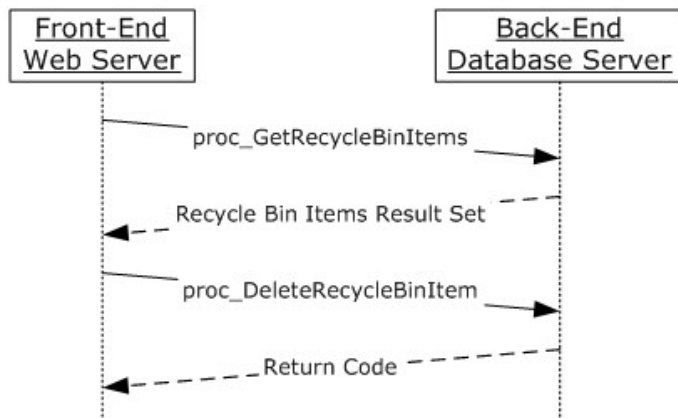


Figure 17: Delete a Second-stage Recycle Bin Item

This example describes the actions a front-end Web server takes when querying an item in the second-stage Recycle Bin and deleting it permanently from the physical disk of the back-end database server. The actions that happen are:

1. The front-end Web server calls the `proc_GetRecycleBinItems` stored procedure on a content database to enumerate items in the second-stage Recycle Bin.
2. The back-end database server returns a Recycle Bin Items result set as defined in [Recycle Bin Items Result Set](#).
3. The front-end Web server chooses an item from the result set to delete and calls `proc_DeleteRecycleBinItem`.
4. The back-end database server returns an error code indicating if the operation succeeded or not.

4.4 Security Operations

This section provides examples of security operations.

4.4.1 Remove External Security Provider

This example describes the interactions made when a user removes External Security Provider from a site.

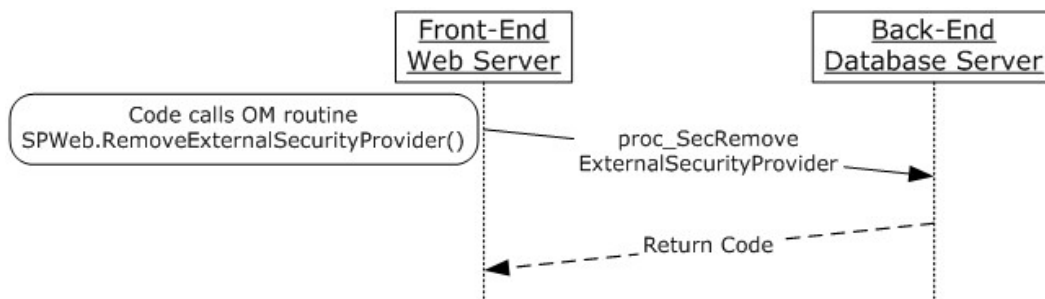


Figure 18: Removing an External Security Provider

This scenario is initiated by a call to the object model command `SPWeb.RemoveExternalSecurityProvider()`. For simplicity's sake, this example assumes that the code has already instantiated the site collection (`SPSite`) and site (`SPWeb`) object.

The following actions happen:

1. The front-end Web server calls the stored procedure `proc_SecRemoveExternalSecurityProvider` using the site collection identifier and site identifier that was initialized before.
2. The back-end database server returns return code 0.

4.4.2 Get the ACL of a Specific `SPListItem`

This example describes the interactions made when the user wants to get the access control list (ACL) about a specific list item.

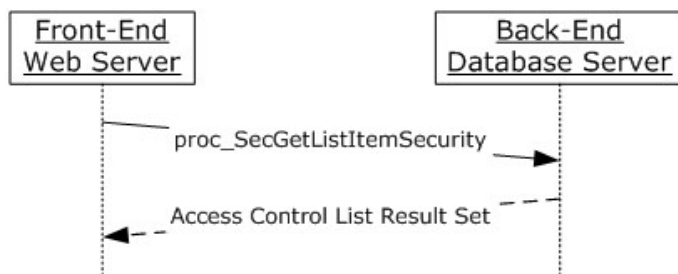


Figure 19: Retrieving an ACL for a List Item

This scenario is initiated when the user wants to update a list item with unique permission. The front-end Web server checks if the user has enough permission before updating the item. For simplicity's sake, this example assumes that the code has already instantiated the site collection (`SPSite`), site (`SPWeb`), and list (`SPList`) objects.

The following actions happen:

1. The front-end Web server calls the stored procedure `proc_SecGetListItemSecurity` using the site collection identifier and list identifier.
2. The back-end database server returns the access control list (ACL) and Windows SharePoint Services Rights Masks for anonymous users as defined in [\[MS-WSSFO2\]](#) section 2.2.3.14 of the list item.

3. The front-end Web server uses the returned access control list (ACL) and Windows SharePoint Services Rights Masks for anonymous users as defined in [\[MS-WSSFO2\]](#) section 2.2.3.14 to check if current user has enough permission to update the list item.

4.4.3 Retrieve All Site Members

This example describes the interactions made when the user wants to get information about All Site Members such as for backup.

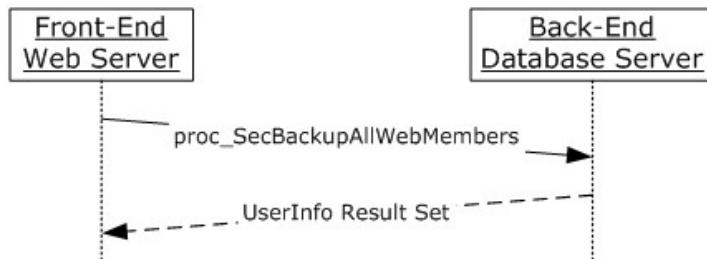


Figure 20: Retrieve All Site Members

The front-end Web server gets All Site Members from the back-end database server and returns to the user. For simplicity's sake, this example assumes that the code has already instantiated the site collection (SPSite) and site (SPWeb) objects.

The following actions happen:

1. The front-end Web server calls the stored procedure `proc_SecBackupAllWebMembers` using the site collection identifier and site identifier that were initialized before.
2. The back-end database server returns the [UserInfo Result Set](#) for the site.

4.5 Database Integrity and Maintenance Operations

This section provides examples of database integrity and maintenance operations.

4.5.1 Find Orphaned Objects for Repair

This example describes the interaction between the front-end Web server and the back-end database server when searching for orphaned objects for repair purposes.

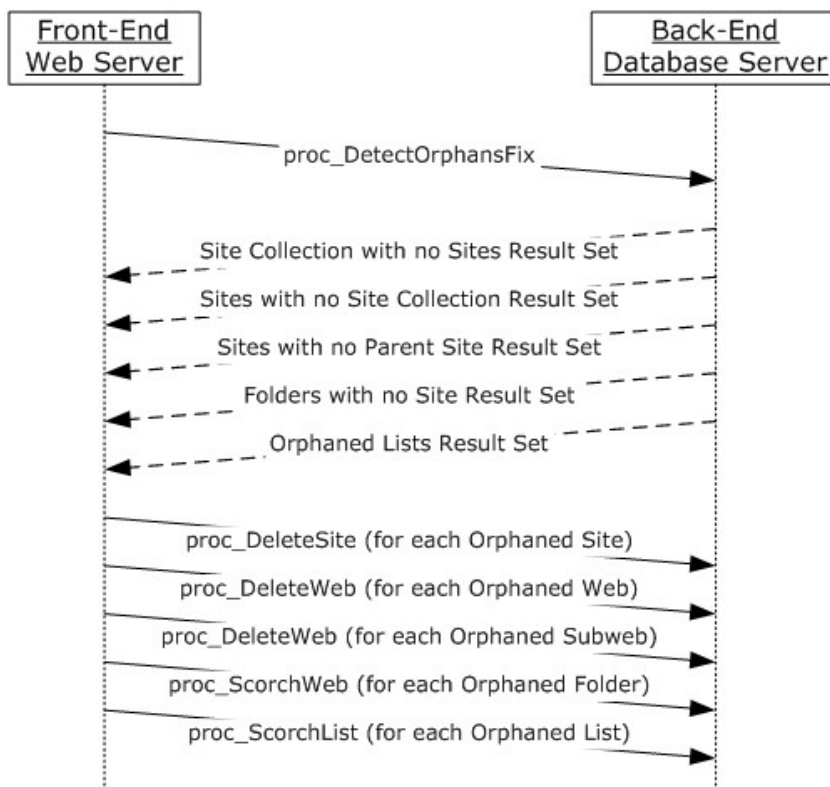


Figure 21: Find Orphaned Objects to Repair

The following actions occur:

1. The front-end Web server calls the stored procedure `proc_DetectOrphansFix`.
2. The back-end database server returns five result sets as follows:
 - Site Collection with no Sites result set (orphaned site collections) as defined in [Site Collection with no Sites Result Set](#).
 - Sites with no Site Collection result set (orphaned sites) as defined in [Sites with no Site Collection Result Set](#).
 - Sites with no Parent Site result set (orphaned subsites) as defined in [Sites with no Parent Site Result Set](#).
 - Folders with no Site result set (orphaned folders) as defined in [Folders with no Site Result Set](#).
 - Orphaned Lists result set as defined in [Orphaned Lists Result Set](#).
3. The front-end Web server deletes orphaned site collections by calling the stored procedure `proc_DeleteSite`, as defined in [\[MS-WSSDLIM2\]](#) section 3.1.4.31, for each site collection.
4. The front-end Web server deletes orphaned sites by calling stored procedure `proc_DeleteWeb`, as defined in [\[MS-WSSDLIM2\]](#) section 3.1.4.35, for each site.
5. The front-end Web server deletes orphaned subsites by calling stored procedure `proc_DeleteWeb`, as defined in [\[MS-WSSDLIM2\]](#) section 3.1.4.35, for each subsite.

6. The front-end Web server deletes orphaned folders by calling the stored procedure `proc_ScorchWeb` (section [3.1.5.63](#)) for each folder.
7. The front-end Web server deletes orphaned lists by calling the stored procedure `proc_ScorchList` for each list.

4.6 Query Operations

This section provides examples of query operations.

4.6.1 Filtered Query Operations by Owner

This example describes the interactions made between the front-end Web server and the back-end database server when the protocol client wants to obtain the site collection identifier of the site collection when the input pattern for the name of the owner of this site collection is specified.



Figure 22: Retrieving the site collection by owner

The following actions happen:

1. The front-end Web server calls the stored procedure `proc_GetSPSiteGuidsGivenOwner`
2. The back-end database server returns the [Site Guids Given Owner Result Set](#)

4.6.2 Filtered Query Operations by Best Match

This example describes the interactions made between the front-end Web server and the back-end database server when the protocol client wants to get the site identifier and offset of the first matched site.

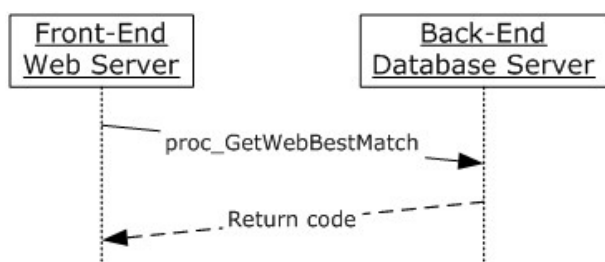


Figure 23: Retrieving the site identifier and offset of the first matched site

The following actions happen:

1. The front-end Web server calls the stored procedure `proc_GetWebBestMatch`

2. The back-end database server returns:
3. The site identifier of the first found site through the output parameter @BestMatchWebId
4. The zero-based offset of the first found site through the output parameter @BestMatchOffset

4.6.3 Pages Query Operations

This example describes the interactions made between the front-end Web server and the back-end database server when the protocol client wants to get the subset of lists for a specific site.

This stored procedure can be called multiple times to get next set or other set of lists. Value for @StartRow is used to specify the row number to start with and value for @PageSize is used to specify the size of returned subset.

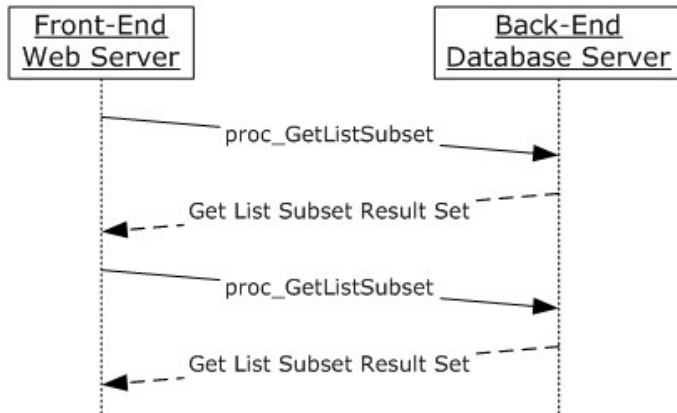


Figure 24: Retrieving the List Subset

The following actions happen:

1. The front-end Web server calls the stored procedure proc_GetListSubset
2. The back-end database server returns the [Get List Subset Result Set](#).
3. The front-end Web server calls the stored procedure proc_GetListSubset with a different value for @StartRow
4. The back-end database server returns the Get List Subset Result Set.

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 Protocol [\[MS-TDS\]](#).

The database access account used by the front-end Web server must have access to the appropriate content database on the back-end database server. If the account does not have the correct access rights, access will be denied when attempting to set up the Tabular Data Stream Protocol connection to the content database or when calling the stored procedures.

Interactions with SQL are susceptible to tampering and other forms of security risks. Implementers are advised to sanitize input parameters for stored procedures prior to invoking the stored procedure.

5.2 Index of Security Parameters

Security Parameter	Section
proc_SecBackupAllWebMembers	3.1.5.64
proc_SecGetListItemSecurity	3.1.5.65
proc_SecRemoveExternalSecurityProvider	3.1.5.66

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® 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 3.1.5.7:](#) SharePoint Foundation 2010 will in certain cases pass an @EventData parameter that is more than 4000 characters in length, for example when @Event is 0x00000006 (Content Type updated).

[<2> Section 3.1.5.8:](#) SharePoint Foundation 2010 will in certain cases pass an @EventData parameter that is more than 4000 characters in length, for example when @Event is 0x00000006 (Content Type updated).

7 Change Tracking

No table of changes is available. The document is either new or has had no changes since its last release.

8 Index

A

Abstract data model

[client](#) 96

[server](#) 23

[Applicability](#) 15

[Audit event source simple type](#) 16

[Audit event type simple type](#) 16

[Auditing operations example](#) 99

B

[Back-end database interface](#) 23

[Binary structures - overview](#) 18

[Bit fields - overview](#) 18

C

[Capability negotiation](#) 15

[Change tracking](#) 112

Client

[abstract data model](#) 96

[front-end Web server interface](#) 96

[initialization](#) 97

[local events](#) 98

[message processing](#) 97

overview ([section 3](#) 23, [section 3.2](#) 96)

[sequencing rules](#) 97

[timer events](#) 97

[timers](#) 97

Common data types

[overview](#) 16

D

Data model - abstract

[client](#) 96

[server](#) 23

Data types

[audit event source simple type](#) 16

[audit event type simple type](#) 16

[common](#) 16

[delete item type simple type](#) 17

[recycle bin stage simple type](#) 18

Data types - simple

[audit event source](#) 16

[audit event type](#) 16

[delete item type](#) 17

[overview](#) 16

[recycle bin stage](#) 18

[Database integrity and maintenance operations](#)

[example](#) 106

[Delete item type simple type](#) 17

E

Events

[local - client](#) 98

[local - server](#) 96

[timer - client](#) 97

[timer - server](#) 96

Examples

[auditing operations](#) 99

[database integrity and maintenance operations](#) 106

[overview](#) 99

[query operations](#) 108

[quota management operations](#) 99

[recycle bin operations](#) 102

[security operations](#) 104

F

[Fields - vendor-extensible](#) 15

[Flag structures - overview](#) 18

[fn_CompareTZTransitionDate method](#) 31

[fn_EscapeForLike method](#) 32

[fn_GetRootFolder method](#) 33

[fn_HtmlEncode method](#) 33

[fn_IsOverQuotaOrWriteLocked method](#) 34

[fn_LocalDayFromUTCDate method](#) 34

[Folders with No Site result set](#) 19

[Front-end Web server interface](#) 96

G

[Glossary](#) 8

H

Higher-layer triggered events

[server](#) 31

I

[Implementer - security considerations](#) 110

[Index of security parameters](#) 110

[Informative references](#) 11

Initialization

[client](#) 97

[server](#) 31

Interfaces - client

[front-end Web server](#) 96

Interfaces - server

[back-end database](#) 23

[Introduction](#) 8

L

Local events

[client](#) 98

[server](#) 96

M

Message processing

[client](#) 97

Messages

binary structures	18
bit fields	18
common data types	16
enumerations	16
flag structures	18
Folders with No Site result set	19
Orphaned Lists result set	19
RecycleBin table structure	21
result sets	18
simple data types	16
Site Collection with No Sites result set	18
Sites with No Parent Site result set	19
Sites with No Site Collection result set	18
table structures	20
transport	16
User Storage Info result set	20
view structures	20
XML structures	22
Methods	
fn_CompareTZTransitionDate	31
fn_EscapeForLike	32
fn_GetRootFolder	33
fn_HtmlEncode	33
fn_IsOverQuotaOrWriteLocked	34
fn_LocalDayFromUTCDate	34
proc_AddAuditEntry	35
proc_AddAuditEntryUrl	36
proc_CalculateAndUpdateSiteDiskUsed	38
proc_ConfirmSiteUsage	38
proc_ConvertStringToDate	39
proc_CountAuditEntries	40
proc_DefragmentIndices	40
proc_DeleteRecycleBinItem	41
proc_DetectOrphans	42
proc_DetectOrphansFix	43
proc_DTSetRelationship	44
proc_EnumRecycleBinItemsForCleanup	44
proc_EnumRecycleBinToFreeSecondStageQuota	45
proc_EnumSitesForDeadWebCheck	46
proc_ForceDeleteList	47
proc_GetAdminRecycleBinInfo	47
proc_GetAdminRecycleBinItems	48
proc_GetAllSPWebIdentifiersGivenSiteGuid	50
proc_GetAuditEntries	50
proc_GetCustomizedDocumentsInWeb	52
proc_GetDatabaseInformation	95
proc_GetDeadWebInfo	53
proc_GetDocLibrarySizes	53
proc_GetDocSizeInfo	55
proc_GetFirstUniqueAncestorWebUrl	56
proc_GetListBestMatch	57
proc_GetListSizes	57
proc_GetListSubset	58
proc_GetRecycleBinItemInfo	59
proc_GetRecycleBinItems	60
proc_GetSiteCollectionBestMatch	62
proc_GetSiteCollectionSubset	63
proc_GetSiteQuota	64
proc_GetSiteUsage	65
proc_GetSizeOfWebPartsOnPage	65
proc_GetSPSiteGuidsGivenHostHeaderPattern	67
proc_GetSPSiteGuidsGivenIdentity	67
proc_GetSPSiteGuidsGivenLockState	68
proc_GetSPSiteGuidsGivenOwner	68
proc_GetSPSiteGuidsGivenSecondaryOwner	69
proc_GetSPWebIdentifiersGivenTitle	71
proc_GetStorageMetrics	79
proc_GetTimerLock	71
proc_GetTotalDiscussionsSize	72
proc_GetUniqueScopesInWeb	73
proc_GetUserStorageInfo	74
proc_GetWebBestMatch	75
proc_GetWebSubset	75
proc_MakeExceptionForThrottle	76
proc_MoveRecycleBinItemToSecondStage	77
proc_ProcessStorageMetricsChanges	81
proc_QMChangeSiteDiskUsedAndContentTimeSta	82
proc_QMGetDiskWarning	82
proc_QMMarkDiskWarning	83
proc_RestoreRecycleBinItem	83
proc_RevertDocContentStreams	86
proc_ScorchList	87
proc_ScorchWeb	87
proc_SecBackupAllWebMembers	88
proc_SecGetListItemSecurity	89
proc_SecRemoveExternalSecurityProvider	90
proc_SetAuditMask	90
proc_SetDatabaseInformation	96
proc_SetDeadWebNotificationCount	91
proc_SetListRequestAccess	91
proc_SetSiteQuota	92
proc_SetSubscription	93
proc_SiteCollectionExists	93
proc_SizeOfPersonalizationsPerUser	94
proc_TrimAuditEntries	94
proc_UpdateDiskUsed	95
proc_UpdateStatistics	95
N	
Normative references	10
O	
Orphaned Lists result set	19
Overview (synopsis)	11
P	
Parameters - security index	110
Preconditions	15
Prerequisites	15
proc_AddAuditEntry method	35
proc_AddAuditEntryUrl method	36
proc_CalculateAndUpdateSiteDiskUsed method	38
proc_ConfirmSiteUsage method	38
proc_ConvertStringToDate method	39
proc_CountAuditEntries method	40
proc_DefragmentIndices method	40
proc_DeleteRecycleBinItem method	41
proc_DetectOrphans method	42

[proc_DetectOrphansFix_method](#) 43
[proc_DTSetRelationship_method](#) 44
[proc_EnumRecycleBinItemsForCleanup_method](#) 44
[proc_EnumRecycleBinToFreeSecondStageQuota_method](#) 45
[proc_EnumSitesForDeadWebCheck_method](#) 46
[proc_ForceDeleteList_method](#) 47
[proc_GetAdminRecycleBinInfo_method](#) 47
[proc_GetAdminRecycleBinItems_method](#) 48
[proc_GetAllSPWebIdentifiersGivenSiteGuid_method](#) 50
[proc_GetAuditEntries_method](#) 50
[proc_GetCustomizedDocumentsInWeb_method](#) 52
[proc_GetDatabaseInformation_method](#) 95
[proc_GetDeadWebInfo_method](#) 53
[proc_GetDocLibrarySizes_method](#) 53
[proc_GetDocSizeInfo_method](#) 55
[proc_GetFirstUniqueAncestorWebUrl_method](#) 56
[proc_GetListBestMatch_method](#) 57
[proc_GetListSizes_method](#) 57
[proc_GetListSubset_method](#) 58
[proc_GetRecycleBinItemInfo_method](#) 59
[proc_GetRecycleBinItems_method](#) 60
[proc_GetSitecollectionBestMatch_method](#) 62
[proc_GetSiteCollectionSubset_method](#) 63
[proc_GetSiteQuota_method](#) 64
[proc_GetSiteUsage_method](#) 65
[proc_GetSizeOfWebPartsOnPage_method](#) 65
[proc_GetSPSiteGuidsGivenHostHeaderPattern_method](#) 67
[proc_GetSPSiteGuidsGivenIdentity_method](#) 67
[proc_GetSPSiteGuidsGivenLockState_method](#) 68
[proc_GetSPSiteGuidsGivenOwner_method](#) 68
[proc_GetSPSiteGuidsGivenSecondaryOwner_method](#) 69
[proc_GetSPWebIdentifiersGivenTitle_method](#) 71
[proc_GetStorageMetrics_method](#) 79
[proc_GetTimerLock_method](#) 71
[proc_GetTotalDiscussionsSize_method](#) 72
[proc_GetUniqueScopesInWeb_method](#) 73
[proc_GetUserStorageInfo_method](#) 74
[proc_GetWebBestMatch_method](#) 75
[proc_GetWebSubset_method](#) 75
[proc_MakeExceptionForThrottle_method](#) 76
[proc_MoveRecycleBinItemToSecondStage_method](#) 77
[proc_ProcessStorageMetricsChanges_method](#) 81
[proc_QMChangeSiteDiskUsedAndContentTimestamp_method](#) 82
[proc_QMGetDiskWarning_method](#) 82
[proc_QMMarkDiskWarning_method](#) 83
[proc_RestoreRecycleBinItem_method](#) 83
[proc_RevertDocContentStreams_method](#) 86
[proc_ScorchList_method](#) 87
[proc_ScorchWeb_method](#) 87
[proc_SecBackupAllWebMembers_method](#) 88
[proc_SecGetListItemSecurity_method](#) 89
[proc_SecRemoveExternalSecurityProvider_method](#) 90
[proc_SetAuditMask_method](#) 90
[proc_SetDatabaseInformation_method](#) 96

[proc_SetDeadWebNotificationCount_method](#) 91
[proc_SetListRequestAccess_method](#) 91
[proc_SetSiteQuota_method](#) 92
[proc_SetSubscription_method](#) 93
[proc_SiteCollectionExists_method](#) 93
[proc_SizeOfPersonalizationsPerUser_method](#) 94
[proc_TrimAuditEntries_method](#) 94
[proc_UpdateDiskUsed_method](#) 95
[proc_UpdateStatistics_method](#) 95
[Product behavior](#) 111

Q

[Query operations example](#) 108
[Quota management operations example](#) 99

R

[Recycle bin operations example](#) 102
[Recycle bin stage simple type](#) 18
[RecycleBin table structure](#) 21

References

[informative](#) 11
[normative](#) 10
[Relationship to other protocols](#) 14
 Result sets - messages
 [Folders with No Site](#) 19
 [Orphaned Lists](#) 19
 [Site Collection with No Sites](#) 18
 [Sites with No Parent Site](#) 19
 [Sites with No Site Collection](#) 18
 [User Storage Info](#) 20
 Result sets - overview 18

S

Security

[implementer considerations](#) 110
[parameter index](#) 110
[Security operations example](#) 104

Sequencing rules

[client](#) 97

Server

[abstract data model](#) 23
[back-end database interface](#) 23
[fn_CompareTZTransitionDate_method](#) 31
[fn_EscapeForLike_method](#) 32
[fn_GetRootFolder_method](#) 33
[fn_HtmlEncode_method](#) 33
[fn_IsOverQuotaOrWriteLocked_method](#) 34
[fn_LocalDayFromUTCDate_method](#) 34
[higher-layer triggered events](#) 31
[initialization](#) 31
[local events](#) 96
 overview ([section 3](#) 23, [section 3.1](#) 23)
[proc_AddAuditEntry_method](#) 35
[proc_AddAuditEntryUrl_method](#) 36
[proc_CalculateAndUpdateSiteDiskUsed_method](#) 38
[proc_ConfirmSiteUsage_method](#) 38
[proc_ConvertStringToDate_method](#) 39
[proc_CountAuditEntries_method](#) 40

[proc DefragmentIndices method](#) 40
[proc DeleteRecycleBinItem method](#) 41
[proc DetectOrphans method](#) 42
[proc DetectOrphansFix method](#) 43
[proc DTSetRelationship method](#) 44
[proc EnumRecycleBinItemsForCleanup method](#) 44
[proc EnumRecycleBinToFreeSecondStageQuota method](#) 45
[proc EnumSitesForDeadWebCheck method](#) 46
[proc ForceDeleteList method](#) 47
[proc GetAdminRecycleBinInfo method](#) 47
[proc GetAdminRecycleBinItems method](#) 48
[proc GetAllSPWebIdentifiersGivenSiteGuid method](#) 50
[proc GetAuditEntries method](#) 50
[proc GetCustomizedDocumentsInWeb method](#) 52
[proc GetDatabaseInformation method](#) 95
[proc GetDeadWebInfo method](#) 53
[proc GetDocLibrarySizes method](#) 53
[proc GetDocSizeInfo method](#) 55
[proc GetFirstUniqueAncestorWebUrl method](#) 56
[proc GetListBestMatch method](#) 57
[proc GetListSizes method](#) 57
[proc GetListSubset method](#) 58
[proc GetRecycleBinItemInfo method](#) 59
[proc GetRecycleBinItems method](#) 60
[proc GetSiteCollectionBestMatch method](#) 62
[proc GetSiteCollectionSubset method](#) 63
[proc GetSiteQuota method](#) 64
[proc GetSiteUsage method](#) 65
[proc GetSizeOfWebPartsOnPage method](#) 65
[proc GetSPSiteGuidsGivenHostHeaderPattern method](#) 67
[proc GetSPSiteGuidsGivenIdentity method](#) 67
[proc GetSPSiteGuidsGivenLockState method](#) 68
[proc GetSPSiteGuidsGivenOwner method](#) 68
[proc GetSPSiteGuidsGivenSecondaryOwner method](#) 69
[proc GetSPWebIdentifiersGivenTitle method](#) 71
[proc GetStorageMetrics method](#) 79
[proc GetTimerLock method](#) 71
[proc GetTotalDiscussionsSize method](#) 72
[proc GetUniqueScopesInWeb method](#) 73
[proc GetUserStorageInfo method](#) 74
[proc GetWebBestMatch method](#) 75
[proc GetWebSubset method](#) 75
[proc MakeExceptionForThrottle method](#) 76
[proc MoveRecycleBinItemToSecondStage method](#) 77
[proc ProcessStorageMetricsChanges method](#) 81
[proc QMChangeSiteDiskUsedAndContentTimestamp method](#) 82
[proc QMGetDiskWarning method](#) 82
[proc QMMarkDiskWarning method](#) 83
[proc RestoreRecycleBinItem method](#) 83
[proc RevertDocContentStreams method](#) 86
[proc ScorchList method](#) 87
[proc ScorchWeb method](#) 87
[proc SecBackupAllWebMembers method](#) 88
[proc SecGetListItemSecurity method](#) 89
[proc SecRemoveExternalSecurityProvider method](#) 90
[proc SetAuditMask method](#) 90
[proc SetDatabaseInformation method](#) 96
[proc SetDeadWebNotificationCount method](#) 91
[proc SetListRequestAccess method](#) 91
[proc SetSiteQuota method](#) 92
[proc SetSubscription method](#) 93
[proc SiteCollectionExists method](#) 93
[proc SizeOfPersonalizationsPerUser method](#) 94
[proc TrimAuditEntries method](#) 94
[proc UpdateDiskUsed method](#) 95
[proc UpdateStatistics method](#) 95
[timer events](#) 96
[timers](#) 31
Simple data types
[audit event source](#) 16
[audit event type](#) 16
[delete item type](#) 17
[overview](#) 16
[recycle bin stage](#) 18
[Site Collection with No Sites result set](#) 18
[Sites with No Parent Site result set](#) 19
[Sites with No Site Collection result set](#) 18
[Standards assignments](#) 15
Structures
[binary](#) 18
[table and view](#) 20
[XML](#) 22
T
Table structures
[RecycleBin](#) 21
[Table structures - overview](#) 20
Timer events
[client](#) 97
[server](#) 96
Timers
[client](#) 97
[server](#) 31
[Tracking changes](#) 112
[Transport](#) 16
Triggered events - higher-layer
[server](#) 31
U
[User Storage Info result set](#) 20
V
[Vendor-extensible fields](#) 15
[Versioning](#) 15
[View structures - overview](#) 20
X
[XML structures](#) 22