[MS-WSSFO2]:

Windows SharePoint Services (WSS): File Operations Database Communications Version 2 Protocol

Revision Summary

Date	Revision History	Revision Class	Comments
8/14/2009	1.0	Major	First Release.
9/25/2009	1.1	Minor	Updated the technical content.
11/6/2009	1.1.1	Editorial	Revised and edited the technical content.
12/18/2009	2.0	Major	Updated and revised the technical content.
1/29/2010	2.0.1	Editorial	Revised and edited the technical content.
3/12/2010	3.0	Major	Updated and revised the technical content.
4/23/2010	3.0.1	Editorial	Revised and edited the technical content.
6/4/2010	4.0	Major	Updated and revised the technical content.
7/16/2010	4.0	None	No changes to the meaning, language, or formatting of the technical content.
8/27/2010	4.0	None	No changes to the meaning, language, or formatting of the technical content.
10/8/2010	4.0	None	No changes to the meaning, language, or formatting of the technical content.
11/19/2010	4.0	None	No changes to the meaning, language, or formatting of the technical content.
1/7/2011	5.0	Major	Significantly changed the technical content.
2/11/2011	5.0	None	No changes to the meaning, language, or formatting of the technical content.
3/25/2011	5.0	None	No changes to the meaning, language, or formatting of the technical content.
5/6/2011	5.0	None	No changes to the meaning, language, or formatting of the technical content.
6/17/2011	6.0	Major	Significantly changed the technical content.
9/23/2011	7.0	Major	Significantly changed the technical content.
12/16/2011	8.0	Major	Significantly changed the technical content.
3/30/2012	8.0	None	No changes to the meaning, language, or formatting of the technical content.
7/12/2012	8.0	None	No changes to the meaning, language, or formatting of the technical content.
9/12/2012	8.0	None	No changes to the meaning, language, or formatting of the technical content.
10/8/2012	8.0	None	No changes to the meaning, language, or formatting of the technical content.
2/11/2013	8.0	None	No changes to the meaning, language, or formatting of the technical content.

Date	Revision History	Revision Class	Comments
7/30/2013	8.0	None	No changes to the meaning, language, or formatting of the technical content.
11/18/2013	8.0	None	No changes to the meaning, language, or formatting of the technical content.
2/10/2014	8.0	None	No changes to the meaning, language, or formatting of the technical content.
4/30/2014	8.1	Minor	Clarified the meaning of the technical content.
7/31/2014	9.0	Major	Significantly changed the technical content.
10/30/2014	9.1	Minor	Clarified the meaning of the technical content.
6/23/2016	9.2	Minor	Clarified the meaning of the technical content.
9/14/2016	9.2	None	No changes to the meaning, language, or formatting of the technical content.
9/19/2017	9.3	Minor	Clarified the meaning of the technical content.
12/12/2017	9.3	None	No changes to the meaning, language, or formatting of the technical content.

Table of Contents

1 Introduction

The File Operations Database Communications Version 2 Protocol specifies the communication sequences used by front-end web servers to perform data query and update commands on back-end database servers as part of file, user, and group administration operations.

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

1.1 Glossary

This document uses the following terms:

access control list (ACL): A list of access control entries (ACEs) that collectively describe the security rules for authorizing access to some resource; for example, an object or set of objects.

Active Directory: A general-purpose network directory service. **Active Directory** also refers to the Windows implementation of a directory service. **Active Directory** stores information about a variety of objects in the network. User accounts, computer accounts, groups, and all related credential information used by the Windows implementation of Kerberos are stored in **Active Directory**. **Active Directory** is either deployed as Active Directory Domain Services (AD DS) or Active Directory Lightweight Directory Services (AD LDS). [MS-ADTS] describes both forms. For more information, see [MS-AUTHSOD] section 1.1.1.5.2, Lightweight Directory Access Protocol (LDAP) versions 2 and 3, Kerberos, and DNS.

alert: An Internet message that is sent to subscribers automatically to notify them when user-defined criteria are met. Alerts are generated automatically when items such as documents, webpages, list items, sites, or other resources on a server are changed.

anonymous user: A user who presents no credentials when identifying himself or herself. The process for determining an anonymous user can differ based on the authentication protocol, and the documentation for the relevant authentication protocol should be consulted.

assembly name: The name of a collection of one or more files that is versioned and deployed as a unit. See also assembly.

attachment: An external file that is included with an Internet message or associated with an item in a SharePoint list.

audit entry: Information that is recorded about an operation on an object that is stored on a server.

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

author: The user who created a **list item**.

back-end database server: A server that hosts data, configuration settings, and stored procedures that are associated with one or more applications.

backward link: A hyperlink between a referenced document and a referencing party. For example, if Document A contains a hyperlink to Document B, then Document B has a backward link to Document A.

Boolean: An operation or expression that can be evaluated only as either true or false.

bot: A structured HTML comment that is processed by a front-end web server when the containing document is opened by or saved to the server. Also referred to as web bot.

build dependency set: A serialized .NET Framework object that represents a set of file dependencies.

change log: A log of changes, such as add and delete, that are made to objects that are stored on a **back-end database server**. Applications can use this information to identify changes that occurred on those objects.

character set: A mapping between the characters of a written language and the values that are used to represent those characters to a computer.

check in: The process of placing a file or project into a source repository. This releases the lock for editing and enables other users to view the updated file or check out the file. See also **check out**.

check out: The process of retrieving a writable copy of a file or project from a source repository. This locks the file for editing to prevent other users from overwriting or editing it inadvertently.

checked out: A **publishing level** that indicates that a document has been created and locked for exclusive editing by a user in a version control system.

code page: An ordered set of characters of a specific script in which a numerical index (code-point value) is associated with each character. Code pages are a means of providing support for **character sets** and keyboard layouts used in different countries. Devices such as the display and keyboard can be configured to use a specific code page and to switch from one code page (such as the United States) to another (such as Portugal) at the user's request.

Collaborative Application Markup Language (CAML): An XML-based language that is used to describe various elements, such as queries and views, in sites that are based on SharePoint Products and Technologies.

collation order: A rule for establishing a sequence for textual information.

Component Object Model (COM): An object-oriented programming model that defines how objects interact within a single process or between processes. In **COM**, clients have access to an object through interfaces implemented on the object. For more information, see [MS-DCOM].

contact: An object of the contact class that represents a company or person whom a user can contact.

content database: A database that is stored on a **back-end database server** and contains stored procedures, site collections, and the contents of those site collections.

content type: A named and uniquely identifiable collection of settings and fields that store metadata for individual items in a SharePoint list. One or more content types can be associated with a list, which restricts the contents to items of those types.

content type identifier: A unique identifier that is assigned to a **content type**.

Coordinated Universal Time (UTC): A high-precision atomic time standard that approximately tracks Universal Time (UT). It is the basis for legal, civil time all over the Earth. Time zones around the world are expressed as positive and negative offsets from UTC. In this role, it is also referred to as Zulu time (Z) and Greenwich Mean Time (GMT). In these specifications, all references to UTC refer to the time at UTC-0 (or GMT).

current user: The user who is authenticated during processing operations on a **front-end web server** or a **back-end database server**.

current version: The latest version of a document that is available to a user, based on the permissions of the user and the publishing level of the document.

custom action: An extension to the user interface, such as a button on a toolbar or a link on a site settings page.

default view: The layout and organization of a document or list that appears automatically when users open that document or display that list.

directory name: A segment of a **store-relative URL** that refers to a directory. A directory name is everything that appears before the last slash in a store-relative form URL.

dirty: The condition of an entity, such as a component or a file, that indicates that the entity or properties of the entity were changed after the entity was last saved.

discussion board: A list in which users can read, post, and reply to messages from other users who are members of the same discussion board.

display name: A text string that is used to identify a principal or other object in the user interface. Also referred to as title.

distribution list: A collection of users, computers, contacts, or other groups that is used only for email distribution, and addressed as a single recipient.

document: An object in a **content database** such as a file, folder, **list**, or **site**. Each object is identified by a URI.

document flag: A 4-byte unsigned integer bit mask that provides metadata about the document.

document identifier: A GUID that identifies a document.

document library: A type of list that is a container for documents and folders.

document stream: A byte stream that is associated with a document, such as the content of a file. Some documents do not have document streams.

document template: A file that serves as the basis for new documents.

document version: A copy of a list item that has a version number. A document version can be either a historical version or a current version.

domain group: A container for security and distribution groups. A domain group can also contain other domain groups.

draft: A version of a document or list item that does not have a publishing level of "Published" or "Checked Out".

editor: The user who last modified an item or document in a SharePoint list.

email address: A string that identifies a user and enables the user to receive Internet messages.

email enabled list: A SharePoint list that is configured to accept incoming email messages.

event: (1) Any significant occurrence in a system or an application that requires users to be notified or an entry to be added to a log.

(2) An action or occurrence to which an application might respond. Examples include state changes, data transfers, key presses, and mouse movements.

event host: A site collection, **site**, **list**, list item, **workflow**, feature, or content type that hosts an event receiver.

event receiver: A structured modular component that enables built-in or user-defined managed code classes to act upon objects, such as list items, **lists**, or content types, when specific triggering actions occur.

event sink: A structured, modular component that enables built-in or user-defined classes to act on documents in document libraries when specific triggering actions occur. Event sinks are a deprecated, implementation-specific capability of Windows SharePoint Services 2.0. In Windows SharePoint Services 3.0 and Microsoft SharePoint Foundation 2010, they are replaced by the capabilities of event receivers.

external security provider: An external object that manages permissions on a site.

feature: A package of SharePoint elements that can be activated or deactivated for a specific feature scope.

feature identifier: A **GUID** that identifies a **feature**.

field: A container for metadata within a SharePoint list and associated list items.

field definition: The definition of a field in the **Collaborative Application Markup Language (CAML)**.

field identifier: A GUID that is used to identify a field.

file: A single, discrete unit of content.

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

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

form: A document with a set of controls into which users can enter information. Controls on a form can be bound to elements in the data source of the form, such as fields and groups. See also bind.

forward link: A hyperlink between a referenced document and a referencing party. For example, if Document A contains a hyperlink to Document B, Document A has a forward link to Document B.

front-end web server: A server that hosts webpages, performs processing tasks, and accepts requests from protocol clients and sends them to the appropriate back-end server for further processing.

full URL: A string of characters in a standardized format that identifies a document or resource on the World Wide Web.

fully qualified class name: A class name that includes namespace information. Use of a fully qualified class name ensures that the class name is treated as unique.

globally unique identifier (GUID): A term used interchangeably with universally unique identifier (UUID) in Microsoft protocol technical documents (TDs). Interchanging the usage of these terms does not imply or require a specific algorithm or mechanism to generate the value. Specifically, the use of this term does not imply or require that the algorithms described in [RFC4122] or [C706] must be used for generating the **GUID**. See also universally unique identifier (UUID).

group: A named collection of users who share similar access permissions or roles.

hash: A fixed-size result that is obtained by applying a one-way mathematical function, which is sometimes referred to as a hash algorithm, to an arbitrary amount of data. If the input data changes, the hash also changes. The hash can be used in many operations, including authentication and digital signing.

historical version: Any version of a document or list item that is not one of the current versions. Depending on configuration settings, historical versions can be retained in a **back-end database server**, and might not be visible to specific users.

home page: On the World Wide Web, a document that serves as a starting point for a set of webpages and other files in a website.

host header: An Internet host and port number that identifies a network resource.

host name: The name of a physical server, as described in [RFC952].

HTTP GET: An HTTP method for retrieving a resource, as described in [RFC2616].

HTTP HEAD: An HTTP method for retrieving header information for a resource, as described in [RFC2616].

Hypertext Markup Language (HTML): An application of the Standard Generalized Markup Language (SGML) that uses tags to mark elements in a document, as described in [HTML].

Hypertext Transfer Protocol (HTTP): An application-level protocol for distributed, collaborative, hypermedia information systems (text, graphic images, sound, video, and other multimedia files) on the World Wide Web.

Information Rights Management (IRM): A technology that provides persistent protection to digital data by using encryption, certificates, and authentication. Authorized recipients or users acquire a license to gain access to the protected files according to the rights or business rules that are set by the content owner.

internal version number: A number that increases monotonically and is used to identify conflicts when saving an item.

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

item identifier: An integer that uniquely identifies an item in a SharePoint list.

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

leaf name: The segment of a URL that follows the last slash. If the resource is a directory, the leaf name can be an empty string.

level: A relative position in a hierarchy of data. A level is frequently used when describing how to navigate a hierarchy in an Online Analytical Processing (OLAP) database or a PivotTable report.

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

list identifier: A GUID that is used to identify a **list** in a site collection.

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

list item identifier: See **item identifier**.

list template: An XML-based definition of list settings, including fields and views, and optionally list items. List templates are stored in .stp files in the content database.

list template identifier: A GUID that is used to identify a list template for a SharePoint list.

list view: A named collection of settings for querying and displaying items in a SharePoint list. There are two types of views: Personal, which can be used only by the user who created the view; and Public, which can be used by all users who have permission to access to the site.

locale: A collection of rules and data that are specific to a language and a geographical area. A locale can include information about sorting rules, date and time formatting, numeric and monetary conventions, and character classification.

locked: The condition of a cell, worksheet, or other object that restricts edits or modifications to it by users.

login name: A string that is used to identify a user or entity to an operating system, directory service, or distributed system. For example, in Windows-integrated authentication, a login name uses the form "DOMAIN\username".

major version: An iteration of a software component, document, or list item that is ready for a larger group to see, or has changed significantly from the previous major version. For an item on a SharePoint site, the **minor version** is always "0" (zero) for a major version.

master page: An ASP.NET file that has a predefined layout that can include static text, **HTML** elements, and server controls.

meeting instance: A collection of data for a meeting that occurs only once or a single occurrence of a meeting that occurs multiple times. The data can be stored in a client application or on a website.

Meeting Workspace site: A SharePoint site that is based on a Meeting Workspace site template and has a template ID value of "2". A Meeting Workspace site is used for planning, posting, and working together on meeting materials.

member: A user in the Members group of a site.

minor version: An iteration of a software component, document, or list item that is in progress or has changed only slightly from the previous version. For an item on a SharePoint site, the minor version number is never "0" (zero) and is incremented for each new version of an item, unless a **major version** is explicitly published. When minor versioning is disabled on a SharePoint site, only major version numbers are incremented, and the minor version is always "0" (zero).

mobile device: A small computing device that is easily portable and can be used in various environments.

moderation status: A content approval status that indicates whether a list item was approved by a moderator.

navigation node: An element in the navigational structure of a site. The element is a link or a series of links to a specific page in the site.

navigation node element identifier: An integer that identifies a navigation node. This value is unique for every navigation node in the navigational structure of a SharePoint site.

navigation structure: A hierarchical organization of links between related content on a site.

NULL GUID: A **GUID** of all zeros.

owner: A **security principal** who has the requisite permission to manage a security group.

page: A file that consists of HTML and can include references to graphics, scripts, or dynamic content such as Web Parts.

parent site: The site that is above the current site in the hierarchy of the site collection.

path segment: A portion of a URI, as described in [RFC3986]. See also path component.

permission: A rule that is associated with an object and that regulates which users can gain access to the object and in what manner. See also rights.

permission level: A set of permissions that can be granted to principals or SharePoint groups on an entity such as a site, list, folder, item, or document.

personal view: A view of a list that is created by a user for personal use. The view is unavailable to other users.

principal: An authenticated entity that initiates a message or channel in a distributed system.

published: A condition of portions of a workbook that are marked as being available to the user when that workbook is processed by a protocol server.

published version: The version of a list item that is approved and can be seen by all users. The user interface (UI) version number for a published version is incremented to the next positive major version number and the minor version is "0" (zero). See also **major version** and **minor version**.

publishing level: An integer that is assigned to a document to indicate the publishing status of that version of the document.

read-only mode: An attribute that indicates that an object cannot be changed or deleted. The object can only be accessed or displayed.

Recycle Bin: The location where deleted files are stored until they are either restored, if they were deleted erroneously, or destroyed permanently.

request identifier: A **GUID** that is used to identify a specific action or procedure that is sent to a protocol server or a protocol client.

result set: A list of records that results from running a stored procedure or query, or applying a filter. The structure and content of the data in a result set varies according to the implementation.

return code: A code that is used to report the outcome of a procedure or to influence subsequent events when a routine or process terminates (returns) and passes control of the system to another routine. For example, a return code can indicate whether an operation was successful.

role: A symbolic name that defines a class of users for a set of components. A role defines which users can call interfaces on a component.

role assignment: An association between a principal or a site group and a role definition.

role definition: A named set of permissions for a SharePoint site. See also **permission level**.

role identifier: An integer that uniquely identifies a role definition within a SharePoint site.

root folder: The folder at the top of a hierarchy of folders in a list.

sandboxed solution: A custom solution that can be deployed to a site by a site collection administrator, without approval from the server farm administrator.

scope identifier: A GUID that uniquely identifies a scope within a site collection.

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

security group: A named group of principals on a SharePoint site.

security identifier (SID): An identifier for **security principals** that is used to identify an account or a group. Conceptually, the **SID** is composed of an account authority portion (typically a domain) and a smaller integer representing an identity relative to the account authority, termed the relative identifier (RID). The **SID** format is specified in [MS-DTYP] section 2.4.2; a string representation of **SIDs** is specified in [MS-DTYP] section 2.4.2 and [MS-AZOD] section 1.1.1.2.

security policy: In the form of a collection of security policy settings, the policy itself is an expression of administrative intent regarding how computers and resources on their network should be secured.

security principal: An identity that can be used to regulate access to resources. A security principal can be a user, a computer, or a group that represents a set of users.

security provider: A Component Object Model (COM) object that provides methods that return custom information about the security of a site.

security role: A defined set of access privileges. The security role that is assigned to a user determines the tasks that a user can perform and which parts of the user interface a user can view.

security scope: A tree structure of objects in which every object has the same security settings as the root.

server-relative URL: A relative URL that does not specify a scheme or host, and assumes a base URI of the root of the host, as described in [RFC3986].

setup path: The location where supporting files for a product or technology are installed.

short-term lock: A type of check-out process in Windows SharePoint Services. Short-term checkouts are implicit and are done when a file is opened for editing. A lock is applied to the file while it is being edited in the client application so that other users cannot modify it. After the client application is closed, the lock is released.

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

site collection: A set of websites that are in the same **content database**, have the same owner, and share administration settings. A site collection can be identified by a **GUID** or the **URL** of the top-level site for the site collection. Each site collection contains a top-level site, can contain one or more subsites, and can have a shared navigational structure.

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

site collection identifier: A GUID that identifies a site collection. In stored procedures, the identifier is typically "@SiteId" or "@WebSiteId". In databases, the identifier is typically "SiteId/tp_SiteId".

site definition: A family of site definition configurations. Each site definition specifies a name and contains a list of associated site definition configurations.

site identifier: A GUID that is used to identify a site in a **site collection**.

site template: An XML-based definition of site settings, including formatting, lists, views, and elements such as text, graphics, page layout, and styles. Site templates are stored in .stp files in the content database.

site-relative URL: A URL that is relative to the site that contains a resource and does not begin with a leading slash (/).

stored procedure: A precompiled collection of SQL statements and, optionally, control-of-flow statements that are stored under a name and processed as a unit. They are stored in a SQL database and can be run with one call from an application. Stored procedures return an integer return code and can additionally return one or more result sets. Also referred to as sproc.

store-relative form: See **store-relative URL**.

store-relative URL: A URL that consists only of a **path segment** and does not include the leading and trailing slash.

stream: An element of a compound file, as described in [MS-CFB]. A stream contains a sequence of bytes that can be read from or written to by an application, and they can exist only in storages.

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

SystemID: A binary identifier that is used to uniquely identify a **security principal**. For Windows integrated authentication, it is a **security identifier (SID)**. For an ASP.NET Forms Authentication provider, it is the binary representation that is derived from a combination of the provider name and the user login name.

thicket: A means of storing a complex HTML document with its related files. It consists of a thicket main file and a hidden thicket folder that contains a thicket manifest and a set of thicket supporting files that, together, store the referenced content of the document.

thicket folder: A hidden folder that contains a thicket manifest and a set of thicket supporting files that, together, store the referenced content of a complex HTML document.

thicket main file: The core file of a complex HTML document. It references contained elements such as graphics, pictures, or other media that are stored as thicket supporting files in a thicket folder. The thicket main file is the target that is used by a protocol client to access the content of the document.

thicket supporting file: A file that contains a graphic element, a picture, or other media that is referenced by the **thicket main file** and is stored in the **thicket folder**.

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

unique identifier (UID): A pair consisting of a **GUID** and a version sequence number to identify each resource uniquely. The UID is used to track the object for its entire lifetime through any number of times that the object is modified or renamed.

user account directory path: A string representation of the Lightweight Directory Access Protocol (LDAP) distinguished name for an Active Directory Domain Services (AD DS) container. It defines a set of users, as described in [RFC4514].

user identifier: An integer that uniquely identifies a **security principal** as distinct from all other **security principals** and site groups within the same site collection.

user information list: A list that contains items, each of which represents a **security principal** in a site collection. Each site collection has only one such list and it resides in the top-level site of the site collection.

user name: A unique name that identifies a specific user account. The user name of an account is unique among the other group names and user names within its own domain or workgroup.

UTF-8: A byte-oriented standard for encoding Unicode characters, defined in the Unicode standard. Unless specified otherwise, this term refers to the UTF-8 encoding form specified in [UNICODE5.0.0/2007] section 3.9.

version: See displayed version, **historical version**, **major version**, and **minor version**.

view: See form view (Microsoft InfoPath), **list view** (SharePoint Products and Technologies), or View (Microsoft Business Connectivity Services).

virus scanner: Software that is used to search for and remove computer viruses, worms, and Trojan horses.

web application: A container in a configuration database that stores administrative settings and entry-point **URLs** for **site collections**.

web bot: See **bot**.

Web Part: A reusable component that contains or generates web-based content such as **XML**, **HTML**, and scripting code. It has a standard property schema and displays that content in a cohesive unit on a webpage. See also Web Parts Page.

Web Part zone identifier: A string that identifies a Web Part zone on a Web Parts Page.

Windows code page: A table that relates the character codes (code point values) that are used by an application to keys on a keyboard or to characters on a display. This provides support for **character sets** and keyboard layouts for different countries or regions. Also referred to as character set or charset.

Windows collation name: A string identifier that follows the format of the Transact-Structured Query Language (T-SQL) COLLATE clause.

workflow: A structured modular component that enables the automated movement of documents or items through a specific sequence of actions or tasks that are related to built-in or user-defined business processes.

workflow identifier: A GUID that is used to identify a workflow.

workflow instance: An instance of a workflow association that performs on a list item the process that is defined in a workflow template.

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

XML document: A document object that is well formed, as described in [XML10/5], and might be valid. An XML document has a logical structure that is composed of declarations, elements, comments, character references, and processing instructions. It also has a physical structure that is composed of entities, starting with the root, or document, entity.

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

XML Path Language (XPath): A language used to create expressions that can address parts of an XML document, manipulate strings, numbers, and Booleans, and can match a set of nodes in the document, as specified in [XPATH]. XPath models an XML document as a tree of nodes of different types, including element, attribute, and text. XPath expressions can identify the nodes in an XML document based on their type, name, and values, as well as the relationship of a node to other nodes in the document.

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

1.2 References

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

1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information.

[HTML] World Wide Web Consortium, "HTML 4.01 Specification", W3C Recommendation 24 December 1999, http://www.w3.org/TR/html4/

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

[MC-FPSEWM] Microsoft Corporation, "FrontPage Server Extensions: Website Management Protocol".

[MS-DTYP] Microsoft Corporation, "Windows Data Types".

[MS-FPSE] Microsoft Corporation, "FrontPage Server Extensions Remote Protocol".

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

[RFC1950] Deutsch, P., and Gailly, J-L., "ZLIB Compressed Data Format Specification version 3.3", RFC 1950, May 1996, http://www.ietf.org/rfc/rfc1950.txt

[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

[RFC2660] Rescorla, E., and Schiffman, A., "The Secure HyperText Transfer Protocol", RFC 2660, August 1999, http://www.rfc-editor.org/rfc/rfc2660.txt

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

[TSQL-Ref] Microsoft Corporation, "Transact-SQL Reference (Transact-SQL)", http://msdn.microsoft.com/en-us/library/dd884419.aspx

[XPATH] Clark, J. and DeRose, S., "XML Path Language (XPath), Version 1.0", W3C Recommendation, November 1999, http://www.w3.org/TR/xpath/

1.2.2 Informative References

[MS-WSSO] Microsoft Corporation, "Windows SharePoint Services Overview".

[RFC2518] Goland, Y., Whitehead, E., Faizi, A., et al., "HTTP Extensions for Distributed Authoring - WebDAV", RFC 2518, February 1999, http://www.ietf.org/rfc/rfc2518.txt

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

1.3 Overview

This protocol specifies the communication between the **front-end web server** and the **back-end database server** used to satisfy requests involving **file** access and administration of users and **groups**. This client-to-server protocol uses the Tabular Data Stream Protocol [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 T-SQL language [TSQL-Ref] is used to define the queries and returned data which is transported over Tabular Data Stream.

End user Clients use remote file access protocols to communicate with the Windows SharePoint Services front-end web servers, specifically using FrontPage Server Extensions Remote Protocol (as described in [MS-FPSE]), **Hypertext Transfer Protocol (HTTP)** (as described in [RFC2616]), and WebDAV (as described in [RFC2518]).

Further information about the interoperation of the clients with the front-end web server, and the front-end web server with the back-end database server, can be found in the Windows SharePoint Services Technical Document [MS-WSSO].

1.3.1 File Operations

This protocol provides methods for retrieving and manipulating files' properties, along with support for retrieving and manipulating files' security information. When client requests for files or file 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 that in turn can be used for further calls to other stored procedures. The front-end web server turns the values in the stored procedures' **return codes** and **result sets** into the data and metadata for the files requested by the client, and sends it back to the client using the same protocol used by the initial request.

1.3.2 User and Group Operations

This protocol provides methods for retrieving and manipulating information about individual users and **groups**, along with support for retrieving information from **Active Directory** about users. 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 whether the data is already populated in the local objects that represent the specific user or groups. If it does not exist, 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 stored procedures' return codes and result sets into objects that contain the data and metadata for the requested users or groups, and uses the objects according to implementation-specific procedures.

1.4 Relationship to Other Protocols

This protocol relies on [MS-TDS] as its transport protocol to invoke stored procedures to inspect and manipulate **document** properties via result sets and return codes. Database queries or calls to stored procedures, and the returned result sets, are written in the [TSQL-Ref] language.

Requests to a front-end web server via FrontPage Server Extensions (as specified in [MS-FPSE]) and WebDAV (as specified in [RFC2518]) rely on this protocol, via the front-end web server, to retrieve and manipulate file and security information persistently stored on the back-end database server and to service requests for files and their properties from their clients.

1.5 Prerequisites/Preconditions

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

For operations defined in this document, any file access, addition, or modification has to be to a valid location, such as a **site**, **list**, **document library**, **folder**, or document, as defined by the data within the tables and the front-end web server, in order for the request to be successfully processed. The user making the request to the front-end web server has to have adequate **permission** to access the content of the specified valid location in order for the request to be successfully processed.

1.6 Applicability Statement

This protocol is only applicable to front-end web servers when communicating with the back-end database server for file, user, and **group** administration operations.

1.7 Versioning and Capability Negotiation

The front-end web server and back-end database server use this protocol to perform explicit **version** verifications. The front-end web server calls stored procedure **proc_GetVersion** to retrieve version information from the back-end database server, which it uses to decide whether to connect with the back-end database server. The version information is stored in the **Versions** table (section 2.2.7.11).

1.8 Vendor-Extensible Fields

For complex types (defined in section 2.2.8.3), an implementation of protocol servers can optionally support reading, writing, and persisting additional arbitrary elements or attributes. Unless otherwise specified in the underlying schema definition, this capability MUST NOT be used by vendors or by protocol server implementations to provide extensions to the schema specified in this document.

The **Customization** element, which is specified in the **FieldDefinition** type (section 2.2.8.3.3), can be used by third parties to store additional data on a **field**.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

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

2.2 Message Syntax

The following are common data types used in conjunction with this protocol. The low **level** data type and size are specified using commonly-known data type descriptions. It is possible that the variable could be stored in multiple T-SQL data types, depending on the actual implementation of each stored procedure, result set, or database table.

2.2.1 Simple Data Types

2.2.1.1 Content Type Identifier

A **content type identifier** is a numeric string value of arbitrary but limited length, which uniquely identifies a **content type**, stored on the **back-end database server** (BEDS) as a T-SQL varbinary(512).

2.2.1.2 Document Identifier

A **document identifier** is a **GUID**, as specified in [MS-DTYP] section 2.3.4, used to uniquely identify a document within a **site collection**. Specialized varieties of document identifier include **site identifiers** (section 2.2.1.11) and **list identifiers** (section 2.2.1.5).

2.2.1.3 Event Receiver Identifier

An **event receiver** identifier is a GUID used to uniquely identify an event receiver within a site collection.

2.2.1.4 Feature Identifier

A **feature identifier** is a GUID used to uniquely identify a **feature** within a site collection.

2.2.1.5 List Identifier

A **list identifier** is a variety of **document identifier** (section 2.2.1.2), which is a GUID used to uniquely identify a **list** within a site collection.

2.2.1.6 List Item Identifier

A **list item identifier** is a 4-byte integer value used to uniquely identify a **list item** within any **list** in a particular site collection.

2.2.1.7 Role Identifier

A **role identifier** is a 4-byte integer value used to uniquely identify a **role definition** within a site collection.

Role identifier value	Role definition
1073741825	Guest
1073741826	Reader
1073741827	Contributor
1073741828	Web Designer
1073741829	Administrator

2.2.1.8 Scope Identifier

A **scope identifier** is a GUID used to uniquely identify a scope within a site collection.

2.2.1.9 Site Collection Identifier

A **site collection identifier** is a GUID used to uniquely identify a site collection within a content database.

2.2.1.10 Site Group Identifier

A **site group** identifier is a 4-byte integer value used to uniquely identify a site group within a site collection. Site group identifiers are assigned from the same numbering space as **user identifiers** (section 2.2.1.13) and cannot overlap. Values of -1 and 0 are reserved to indicate invalid or unknown user or site group identifiers.<1>

2.2.1.11 Site Identifier

A **site identifier** is a variety of **document identifier** (section 2.2.1.2). The site identifier is a GUID used to uniquely identify a **site** within a site collection.

2.2.1.12 SystemID

A **SystemID** is a binary value of arbitrary but limited length that uniquely identifies a **principal**, stored on the BEDS as a T-SQL varbinary(512).

2.2.1.13 User Identifier

A **user identifier** is a 4-byte integer value used to uniquely identify a **principal** within a site collection. User identifiers are assigned from the same numbering space as site **group** identifiers (section 2.2.1.10) and cannot overlap. Certain user identifiers are predefined such as the system account (1073741823), the site collection **owner**, and the secondary site collection **contact**. Values of -1 and 0 are reserved to indicate invalid or unknown user or site group identifiers.

2.2.1.14 View Identifier

A view identifier is a 4-byte integer value used to identify a **view** within a **list** or document library. A view identifier is unique only within a particular list or document library.

2.2.1.15 Web Part Identifier

A **Web Part** identifier is a GUID used to uniquely identify a Web Part within a site collection.

2.2.1.16 Workflow Identifier

A **workflow identifier** is a GUID used to uniquely identify a **workflow** within a site collection.

2.2.2 Bit Fields and Flag Structures

2.2.2.1 Audit Flags

Audit Flags is a 4-byte unsigned integer bit mask that tracks operations to be audited on a given object. Auditing is an implementation-specific capability, which can have one or more flags set. The values of the **Audit Flags** bits are specified as follows.

Value	Description
0x00000001	Audit checkout operations.
0x00000002	Audit checkin operations.
0x00000004	Audit view operations.
0x00000008	Audit delete operations.
0x00000010	Audit update operations.
0x00000020	Audit content type update operations.
0x00000040	Audit child object deletion operations.
0x00000080	Audit List Schema change operations.
0x00000100	Audit security change operations.
0x00000200	Audit undelete operations.
0x00000400	Audit **workflow** operations.
0x00000800	Audit copy operations.
0x00001000	Audit move operations.
0x00002000	Audit search operations.
0xFFFFC000	Unused.

2.2.2.2 Configuration Object Status

Configuration Object Status is a 4-byte unsigned integer that describes the status of the associated configuration object. Valid values of the **Configuration Object Status** bits are specified in the following table. The semantic meaning of each value is implementation-specific to the service that owns the configuration object.

Value	Description
0x00000000	The Configuration Object is provisioned and online.
0x00000001	The Configuration Object is disabled. The resources necessary to run this Configuration Object are on the machine, but the administrator needs to provision the Configuration Object and enable it.
0x00000002	The Configuration Object is offline for some unknown reason.
0x00000003	The administrator has issued the command to unprovision the Configuration Object into a disabled state, but the unprovisioning job has not completed yet.
0x00000004	The administrator has issued the command to provision the Configuration Object and turn it online, but the provisioning job has not completed yet.
0x00000005	The administrator has issued the command to upgrade the Configuration Object into an online state, but the upgrade job has not completed yet.

2.2.2.3 Doc Flags

Doc Flags is a 4-byte unsigned integer bit mask that provides metadata about the document, which can have one or more flags set. The values of the **Doc Flags** bits are specified as follows.

Value	Description
0x00000001	This document contains dynamic content to be sent through the **Collaborative Application Markup Language (CAML)** interpreter, an implementation-specific dynamic content generation component. An example of this would be a category **Web bot** present in the source of the **page**.
0x00000002	The document is a "sub image" of another document. This is set if this document is an automatically generated thumbnail or web image based on another **item** in the store.
0x00000004	The document is a type for which there was a registered parser available at the time it was saved. A parser is an implementation-specific component that can extract data and metadata from a document, which can then be used to build a list of hyperlinks and fields for content types.
0x00000008	The document is a type that can contain hyperlinks.
0x00000010	The document has an associated resource in the "_private" folder that SHOULD be renamed in parallel when this file is renamed. An example of this is the count file for a hit counter Web bot.
0x00000020	The document is currently **checked out** to a user.
0x00000040	The document is unghosted.
0x00000080	The document is, by default, a page that contains a personalized view showing the personalized and customized Web Parts of the **current user**.
0x00000100	The document is a type that can have a **document stream**.
0x00000200	The document is currently checked out to a location on the user's client system.
0x00000400	The document has child documents created by the document transformations feature.
0x00000800	The document is only a namespace entry for a **list item** (that is, it corresponds to an item in a non-document library **list** that SHOULD be filtered out from file system-centric enumerations).

Value	Description
0x00001000	Unused.
0x00002000	The document has properties in its metainfo defining a custom order of the content types. This is only valid for folders.
0x00004000	The document MUST be unghosted when "undirtied" (that is, when dependency updates are performed for the document). This is used for documents such as a document library template, which is provisioned as ghosted but ought to be unghosted to demote content type information on the containing document library whenever that information is updated.
0x00008000	Used when a 0-byte document is saved to a document library with required **check out** and at least one required **field**.
0x00010000	The document uses external storage.
0x00020000	There is a shared lock on the document.
0x00040000	The document is the Welcome page of the site.
0x00080000	Accessing the document does not require extra permission that is normally required for documents in private list.
0xFFFF0000	Currently unused and MUST be ignored.

2.2.2.4 Document Store Type

The **Document Store** type is a 1-byte unsigned integer value that specifies the type of a document or the target of a link within or to a document. The only valid values for a **Document Store** type are specified as follows.

Value	Description
0x00	File
0x01	Folder
0x02	Site
0x80	Backward Link, can only be a return value and MUST NOT be used as an input value for a stored procedure.

2.2.2.5 List Flags

List Flags is an 8-byte unsigned integer bit mask that provides metadata about the **list**, which can have one or more flags set. **List Flags** identify an implementation-specific capability. The values of the **List Flags** bits are specified as follows.

Value	Description
0x0000000000000001	This list is an "ordered list".
0x0000000000000002	This bit MUST be ignored.
0x0000000000000004	This list is "undeletable" (that is, it is crucial to the functioning of the containing

Value	Description
	site or site collection).
0x0000000000000008	**Attachments** on list items are disabled. This bit MUST be set if the list is a document library or survey.
0x0000000000000010	This list is a "catalog" (for example, a Web part gallery or **master page** gallery).
0x0000000000000020	This list is associated with a site using the meetings workspace **site template** and contains data scoped to each instance of a recurring meeting.
0x0000000000000040	This list generates **alerts** when a list item is assigned to a user.
0x0000000000000080	This list has versioning enabled, and supports creating **historical versions** of list items when changes occur. This bit MUST be ignored for lists with a **List Base** type (section 2.2.3.11) of survey.
0x0000000000000100	This list MUST be hidden from enumeration functions. This is intended for lists implementing infrastructure for an application.
0x0000000000000200	This list is configured to bring up a page to fill out a **form** to request access from the **owner** when a user is denied access while browsing its list items.
0x0000000000000400	This list has moderation enabled, requiring an approval process when content is created or modified.
0x0000000000000800	If this list is a survey, it will allow multiple responses for a given user rather than restricting users to a single response. This flag MUST be ignored for lists that do not have a **List Base** type of survey.
0x0000000000001000	This list uses the value of each **field's ForcedDisplay** attribute (section 2.2.8.3.3.2) when presenting data from that field. This is commonly used in anonymous surveys to display common placeholder text wherever the respondent's name would normally appear.
0x0000000000002000	This list MUST NOT be serialized as part of saving this site as a site template.
0x0000000000004000	The **List Server Template** (section 2.2.3.12) for this list can only be instantiated in the root site of a given site collection.
0x0000000000008000	When a **List Server Template** is being created for this list, documents in the root of the list can also be serialized.
0x0000000000010000	Insertion of list items via email is enabled for this list.
0x0000000000020000	This is a "private" list. When a **List Server Template** based on this list is created, the new list can be given an **access control list (ACL)** so that only its owner and administrators can access the list.
0x0000000000040000	This document library requires the user to check out documents before modifying them.
0x0000000000080000	This list supports creation of **minor versions** on item revisions.
0x0000000000100000	This document library requires that users have the EditListItems right to see minor versions of documents.
0x0000000000200000	This document library requires that users have the ApproveItems right to see minor versions of documents.
0x0000000000400000	This list supports displaying a user interface for manipulating multiple content types (for example, a list that contains both announcements and tasks).
0x0000000000800000	This list has had its schema customized from the version that exists in the on-disk

Value	Description
	schema file that was used to create it.
0x0000000001000000	Document parsers in this list generate thumbnail files corresponding to documents saved to this list. This bit MUST be ignored for lists that are not document libraries.
0x0000000002000000	Documents or list item attachments in this list can be directly browsed to by anyone who has access to the list itself. This is useful for shared resources such as the master page gallery, where one page can be used throughout a site collection in scopes with varying permissions.
0x0000000004000000	This list has **workflows** associated with it.
0x0000000008000000	This list MUST NOT be automatically exported when exporting a list that references it. Exporting is an implementation-specific capability.
0x0000000010000000	Applications generating server transformations of list items in this list SHOULD default to open the list item in a browser rather than in a separate client-side application. Server transformations are performed by server-side document viewers that can allow clients to view documents without additional client software. Server transformations are an implementation-specific feature.
0x0000000020000000	Creation of folders is blocked in this list.
0x0000000040000000	This list disallows advanced view functionality, such as the datasheet view and views involving Web part to Web part connections.
0x0000000080000000	This list specifies custom sorting orders for the list of content types available on a per-folder basis.
0x0000000100000000	This list MUST NOT be exported as part of a migration package. Migration packages are an implementation-specific feature.
0x0000000200000000	This list SHOULD have its schema cached in memory when possible, rather than retrieving the schema every time the list is accessed.
0x0000000400000000	This bit MUST NOT be returned by the database.
0x0000000800000000	This list's content is not processed by a search crawler.
0x0000001000000000	Data from this list MUST be included when it is saved as a **List Server Template**, even if not otherwise requested.
0x0000002000000000	Content type manipulation is disabled on this list.
0x0000004000000000	RSS feed syndication is disabled for this list.
0x0000008000000000	**Information Rights Management (IRM)** is enabled for documents in this list.
0x0000010000000000	Expiration of IRM rights is enabled for this list. Setting this bit requires that the IRM enabled bit also be set.
0x0000020000000000	Documents that do not have a registered IRM protector will be blocked from this list.
0x0000080000000000	If this list is an events list, this list supports a user interface which allows the user to select user identities when adding new list items.
0x0000100000000000	This list has data validation criteria used to perform custom validation rules prior to the list being updated.
0x0000200000000000	If this list is an events list, this list supports a user interface for adding resources.

Value	Description
0x0000400000000000	The data for this list comes from an external data source.
0x0000800000000000	Calculated expressions in this list MUST preserve NULL values rather than converting them to empty strings.
0x0001000000000000	This list has list scoped **custom actions**.
0x0002000000000000	This site SHOULD NOT be taken offline by a client application.
0x0004000000000000	Protocol clients MUST enforce validation rules defined for this list. Validation rules are an implementation-specific capability.
0x0008000000000000	The flag for this list with value 0x0000000010000000 MUST override any server-wide settings for server transformations. Server transformations are an implementation-specific feature.
0x0010000000000000	This list is part of the infrastructure of the site that contains it and client applications SHOULD treat it as a top level navigation object when displaying information about the site that contains it.
0x0020000000000000	Views of this list SHOULD NOT display options to bulk edit data in a datasheet view.
0x0040000000000000	Protocol clients SHOULD prevent browser execution of active content from files in this library.
0x0080000000000000	The user interface for forms in this list SHOULD navigate to form pages instead of hosting the list form in a modal dialog.
0x0100000000000000	Calculated expressions in this list MUST return error values in the case of data type mismatch.
0x0200000000000000	This list at one time supported creation of minor versions on item revisions, so minor versions might exist for items in this list.
0xFC00000000000000	Unused.

2.2.2.6 Publishing Level Type

A **Publishing Level** type is an 8-bit unsigned integer that describes the **publishing level** of a version of a document. Only the following values are valid for the **Publishing Level** type.

Value	Description
1	The default value, referring to a **published** document shown to all users who have access to its storage location.
2	This document is in a **draft** state and SHOULD only be shown to users who have **permissions** to see documents in a draft state.
255	This document is checked out to a particular user and SHOULD only be shown to that user.

2.2.2.7 Put Flags Type

Put Flags type is a 4-byte integer bit mask containing option flags for adding or updating a document. Zero or more of the following bit flags can be set in a **Put Flags** type. The values of the **PutFlags** bits are specified as follows.

Value	Description
0x00000001	MUST be ignored by the back-end database server.
0x00000002	Unconditionally update the document.
0x00000004	MUST be ignored by the back-end database server.
0x00000008	Keep the document checked out.
0x00000010	MUST be ignored by the back-end database server.
0x00000020	Check the document in.
0x00000040	MUST be ignored by the back-end database server.
0x00000080	Create a **thicket** or thicket supporting folders.
0x00000100	MUST be ignored by the back-end database server.
0x00000200	MUST be ignored by the back-end database server.
0x00000400	MUST be ignored by the back-end database server.
0x00000800	MUST be ignored by the back-end database server.
0x00001000	Create a new UI version of the document, even if it is in a **short-term lock**.
0x00002000	Set only when a document migration operation is occurring.
0x00004000	MUST be ignored by the back-end database server.
0x00008000	MUST be ignored by the back-end database server.
0x00010000	Publish the document: change UI version to published.
0x00020000	Overwrite the document without updating the displayed version of the document.
0x00040000	Unpublish document: Change UI version from published to draft.
0x00080000	Document updated from manifest file: Add document at the default publishing level.
0x00100000	The document is being added or updated as part of a system update: Do not update the last modification time and user.
0x00200000	MUST be ignored by the back-end database server.
0x00400000	Unused.
0x00800000	Do not increment the current internal version counter value for the document. This flag can be set only if the updater can tolerate having his or her changes overwritten by another user in case of a conflict.
0x01000000	Unused.
0x02000000	Keep the document checked out to the user's local disk.
0xFC000000	Unused.

2.2.2.8 Rename Flags

Rename Flags is a 4-byte integer bit mask containing option flags for renaming a document. This bit mask can have zero or more flags set. The reserved values of the **RenameFlags** bits are specified as follows.

Value	Description
0x00000000	Default behavior: Rename all dependent **items**.
0x00000001	Do not update all related documents.<2>
0x00000002	MUST be ignored by the back-end database server.
0x00000004	Server SHOULD find **backward links** in order to update other documents linking to the original document.
0x00000008	MUST be ignored by the back-end database server.
0x00000010	MUST be ignored by the back-end database server.
0x00000020	Allow renaming of sites.
0x00000040	Set the **Doc Flags** (section 2.2.2.3) bit 0x00000004 to match this value when a file's extension changes.
0x00000080	Set the **Doc Flags** bit 0x00000008 to match this value when a file's extension changes.
0x00000100	This value is only used internally within implementation-specific code.
0x00000200	Allow move into the forms directory.
0x00000400	Current user can view draft versions of the source documents in a move operation.
0x00000800	Allow rename operation on a **thicket main file** with missing **thicket supporting files**.
0x00001000	MUST be ignored by the back-end database server.
0x00002000	Overwrite existing files in the destination if the source file is newer.
0x00004000	MUST be ignored by the back-end database server.
0xFFFF8000	Unused.

2.2.2.9 Site Collection Flags

Site Collection Flags is a 4-byte, unsigned integer bit mask that specifies properties that are global to a site collection. This bit mask can have zero or more flags set. The values of the **Site Collection Flags** bits are specified as follows.

Value	Description
0x00000001	This site collection has been write-locked, and user write operations will be blocked.
0x00000002	This site collection is fully **locked**, and user read and write operations will be blocked.
0x000003FC	Unused.

Value	Description
0x00000400	The site collection has sent a notification that disk usage is near its limit.
0x00000800	Unused.
0x00001000	The number of users in this site collection is large. Special consideration can be given to query execution plans when retrieving data.
0x00002000	The site collection has disabled syndication via RSS.
0x00004000	Unused.
0x00008000	Unused.
0x00010000	Unused.
0x00020000	This site collection has been write-locked by an administrative action. Users are restricted to read-only operations.
0x00040000	Unused.
0x00080000	The site collection is restricted to users with user accounts from a particular directory path within Active Directory.
0x00100000	There are **sandboxed solutions** activated in the site collection.
0x00200000	Email has been sent out to site collection owner about the resource usage for sandboxed solutions exceeding the resource quota warning level.
0x00400000	Email has been sent out to the site collection owner about resource usage for sandboxed solutions exceeding the resource quota limit.
0x00800000	Resource usage for the sandbox solutions exceeding the resource quota limit.
0x01000000	This site collection has site collection scoped **custom actions**.
0x02000000	The Visual Upgrade site collection UI is shown. The Visual Upgrade feature allows users to choose the UI of upgraded sites. If the user chooses to update, the sites will use the new interface, which includes the ribbon; otherwise, the sites will continue to use Windows SharePoint Services 3.0 UI.
0x04000000	The Visual Upgrade Site Setting UI is shown on sites in this Site Collection. The Visual Upgrade feature allows users to choose the UI of upgraded sites. If the user chooses to update, the sites will use the new interface, which includes the ribbon; otherwise, the sites will continue to use Windows SharePoint Services 3.0 UI.
0x08000000	User-defined **workflows** are disabled for this site collection.
0xF0000000	Unused.

2.2.2.10 Site Property Flags

Site Property Flags is a 4-byte, unsigned integer bit mask that tracks property flags applied to a site. The site can have one or more **Site Property Flags** set. These flags reference implementation-specific capabilities. The values of the **Site Property Flags** bits are specified as follows.

Value	Description
0x00000001	This site allows display of implementation-specific user presence information in the front-end

Value	Description
	web server.
0x00000002	This site allows display of implementation-specific enhanced user presence information in the front-end web server.
0x00000004	**HTML** views for file dialogs MUST NOT be displayed for this site.
0x00000008	This site has disabled syndication of **list items** via RSS.
0x00000010	Unused.
0x00000020	The front-end web server for this site displays the "quick launch" navigational element.
0x00000040	The front-end web server for this site displays a hierarchical "tree view" navigational element.
0x00000080	Document parsing is disabled for this site.
0x00000100	This site has not yet been provisioned with a site template.
0x00000200	List schema information can be cached for **lists** within this site.
0x00000400	This site has at least one uniquely secured object within it.
0x00000800	Search indexing agents can index the rendered content from ASPX pages within this site.
0x00001000	Search indexing agents MUST NOT index the rendered content from ASPX pages within this site.
0x00004000	The site MUST be locked when performing field or list operations that enable cascade deletion.
0x00008000	This allows display of implementation-specific visual upgrade or site settings information on the front-end web server.
0x00010000	The site has at least one custom action that resides at the site level.
0x00020000	The site has at least one custom action that resides at the list level.
0x00040000	This site has been initialized since upgrade occurred.
0x00080000	This site SHOULD NOT be taken offline by the client.
0xFFFFE000	Unused.

2.2.2.11 UserInfo tp_flags

A 4-byte integer bit mask determining the user's options. This can have one or more flags set. The default value is 0, but it MUST NOT be NULL. The valid flags are:

Value	Description
0x00000000	None
0x00000001	The user is associated with an external application.
0x00000002	When the user accesses the front-end web server, the request MUST have an X-RequestToken HTTP Header, which is an opaque string value generated by the front-end web server.

Value	Description
0x00000004	The user will not have permission to browse the **user information list**.

2.2.2.12 View Flags

View Flags is a 4-byte, integer bit mask that corresponds to properties of a view. This bit mask can have zero or more flags set. The values of the **View Flags** bits are specified as follows.

Value	Description
0x00000001	Normal HTML-based view.
0x00000002	View has been modified by a client application such that it might not be compatible with the web interface for view modification. Implementations MUST restrict modifying any properties they do not understand.
0x00000004	Tabular view. The view will render a check box in front of each row to use for bulk updates.
0x00000008	View MUST NOT be displayed in enumerations of the views of this **list** (that is, in a view selector front-end web server element).
0x00000010	Value is unused and MUST be ignored.
0x00000020	View is read-only, and implementations MUST NOT allow any modifications to its properties.
0x00000040	If the query for this view returns no rows, implementations of the front-end web server MUST return an HTTP 410 error when displaying this view as part of an HTTP request, instead of displaying a normal empty view body.
0x00000080	Presents data in a nontabular fashion. Implementations can format results in a manner compatible with free-form presentation.
0x00000100	View suitable for displaying in an HTML-based file navigation dialog to client applications.
0x00000200	Value is unused, and implementations MUST ignore it.
0x00000400	View has functionality for aggregating data across multiple **XML documents** within an XML form library.
0x00000800	View presents a datasheet view to a rich client application.
0x00001000	Displays all items in the list recursively from the specified folder, instead of only displaying the immediate children of the current folder.
0x00002000	Requires that view's data be expanded based on a calendar recurrence (for example, having a view of list item data for the first Thursday of every month).
0x00004000	View is the system-created view of a user's items awaiting moderation in a moderated list.
0x00008000	System-created moderator's view of a moderated list, which displays list items pending approval.
0x00010000	Threaded view for legacy **discussion boards** (lists with base type 3). Implementations MUST display results in a threaded fashion, and paging of results MUST be done in terms of threads instead of by individual list items.
0x00020000	Displays HTML-based graphical charts of list item data.
0x00040000	A **personal view**, which MUST only be displayed to the user who created the view.

Value	Description
0x00080000	Displays data on a calendar based on date and time properties of the list items.
0x00100000	Default form of the specified type for the corresponding list.
0x00200000	Does not display any list items that are folders.
0x00400000	Displays list items based on the item order of the list. If this list is not an ordered list, this value MUST be 0.
0x00800000	View intended for display on mobile devices.
0x01000000	View is displayed as the **default view** of this list when a mobile view is requested.
0x02000000	Displays historical versions of list items.
0x04000000	Displays list item data in a Gantt chart.
0x08000000	View fetches the list item for the **root folder** of the view, in addition to the standard behavior of fetching all list items contained within it.
0x10000000	View is the default view presented when a view is requested for a particular content type.
0x20000000	View does not display items that have not been approved. Implementations MUST refuse to show this view to **anonymous users**.
0x40000000	View MUST NOT be displayed to any user who does not possess the UseClientIntegration right.
0x80000000	Unused flag value, which MUST be ignored by client applications.

2.2.2.13 Workdays Flag

The **Workdays Flag** is a 2-byte, bit mask that is used to specify the workdays in a week for display in a calendar. The workdays are specified as the sum or bitwise OR of the bit flags specifying each day of the week. For example, if the first day of the week is a Sunday and the workdays are Monday through Friday, the combined flags value would be 0x003E, or 62. The only valid values of the **Workdays Flag** bits are specified as follows.

Value	Description
0x0001	Saturday is a workday.
0x0002	Friday is a workday.
0x0004	Thursday is a workday.
0x0008	Wednesday is a workday.
0x0010	Tuesday is a workday.
0x0020	Monday is a workday.
0x0040	Sunday is a workday.
0xFF80X	Unused and MUST NOT be set.

2.2.2.14 WSS Rights Mask

The **WSS Rights Mask** is an 8-byte, unsigned integer that specifies the rights that can be assigned to a user or site **group**. This bit mask can have zero or more flags set.

The values of the permission mask bits are specified as follows.

Symbolic name	Value	Description
EmptyMask	0x0000000000000000	Grant no permissions.
FullMask	0x7FFFFFFFFFFFFFFF	Grant all permissions.

The **list** and document permissions (0x000000000000XXXX) are specified as follows.

Symbolic name	Value	Description
ViewListItems	0x0000000000000001	Allow viewing of list items in lists, documents in document libraries, and web discussion comments.
AddListItems	0x0000000000000002	Allow addition of list items to lists, documents to document libraries, and web discussion comments.
EditListItems	0x0000000000000004	Allow editing of list items in lists, documents in document libraries, web discussion comments, and to customize Web Part pages in document libraries.
DeleteListItems	0x0000000000000008	Allow deletion of list items from lists, documents from document libraries, and web discussion comments.
ApproveItems	0x0000000000000010	Allow approval of minor versions of a list item or document.
OpenItems	0x0000000000000020	Allow viewing the source of documents with server-side file handlers.
ViewVersions	0x0000000000000040	Allow viewing of past versions of a list item or document.
DeleteVersions	0x0000000000000080	Allow deletion of past versions of a list item or document.
CancelCheckout	0x0000000000000100	Allow discard or **check in** of a document that is checked out to another user.
ManagePersonalViews	0x0000000000000200	Allow creation, change, and deletion of personal views of lists.
	0x0000000000000400	Reserved.
ManageLists	0x0000000000000800	Allow creation and deletion of lists, addition or removal of fields to the schema of a list, and addition or removal of personal views of a list.
ViewFormPages	0x0000000000001000	Allow viewing of forms, views, and application pages, and enumerate lists.
	0x000000000000E000	Reserved.

The web level permissions (0x0000XXXXXXXX0000) are specified as follows.

Symbolic name	Value	Description
Open	0x0000000000010000	Allow access to the items contained within a site, list,

Symbolic name	Value	Description
		or folder.
ViewPages	0x0000000000020000	Allow viewing of pages in a site.
AddAndCustomizePages	0x0000000000040000	Allow addition, modification, or deletion of HTML pages or Web Part pages, and editing of the site using a compatible editing application.
ApplyThemeAndBorder	0x0000000000080000	Allow application of a theme or borders to the entire site.
ApplyStyleSheets	0x0000000000100000	Allow application of a style sheet (.css file) to the site.
ViewUsageData	0x0000000000200000	Allow viewing of reports on site usage.
CreateSSCSite	0x0000000000400000	Allow creation of a site using Self-Service Site Creation, an implementation-specific capability.
ManageSubwebs	0x0000000000800000	Allow creation of a **subsite** within the site or site collection.
CreateGroups	0x0000000001000000	Allow creation of a group of users that can be used anywhere within the site collection.
ManagePermissions	0x0000000002000000	Allow creation and modification of **permission levels** on the site and assigning permissions to users and site groups.
BrowseDirectories	0x0000000004000000	Allow enumeration of documents and folders in a site using [MS-FPSE] and WebDAV interfaces.
BrowseUserInfo	0x0000000008000000	Allow viewing the information about all users of the site.
AddDelPrivateWebParts	0x0000000010000000	Allow addition or removal of personal Web Parts on a Web Part page.
UpdatePersonalWebParts	0x0000000020000000	Allow updating of Web Parts to display personalized information.
ManageWeb	0x0000000040000000	Allow all administration tasks for the site as well as manage content.
	0x0000000F80000000	Reserved.
UseClientIntegration	0x0000001000000000	Allow use of features that launch client applications; otherwise, users can only work on documents on their local machines and upload changes to the front-end web server.
UseRemoteAPIs	0x0000002000000000	Allow use of SOAP, WebDAV, or [MS-FPSE] to access the site.
ManageAlerts	0x0000004000000000	Allow management of **alerts** for all users of the site.
CreateAlerts	0x0000008000000000	Allow creation of email alerts.
EditMyUserInfo	0x0000010000000000	Allow users to change their own user information, such as adding a picture.
	0x0000FE0000000000	Reserved.

The Special permissions (0xXXXX000000000000) are specified as follows.

Symbolic name	Value	Description
	0x3FFF000000000000	Reserved.
EnumeratePermissions	0x4000000000000000	Allow enumeration of permissions on the site, list, folder, document, or list item.
	0x8000000000000000	Reserved.

2.2.3 Enumerations

2.2.3.1 Attachments Flag

The **Attachments Flag** is a 1-byte integer flag that specifies whether an item appears to be an **attachment** or a folder related to attachments based on this document's **URL**. The following are valid values for **Attachments Flag**.

Value	Description
0	The URL does not appear to be an attachment.
1	The URL is an attachment file. The **directory name** of the document has the string "Attachments" as its next-to-last **path segment**, and a 32-bit, base-10, signed integer as the last path segment that is referring to the item ID to which this file is attached and where the permissions will be checked (for example, "Announcements/Attachments/17/file1.txt").
2	The URL is a list item attachment folder (for example, "Announcements/Attachments/17").
3	The URL is the list Attachments folder itself. The last path segment of the URL is the string "Attachments" (for example, "Announcements/Attachments").

2.2.3.2 Audit Item Type

Audit Item type is a 2-byte integer flag that specifies the type of the object in an audit specification. The following are the only valid values for **Audit Item** type.

Value	Description
1	A page or a file.
3	A **list item**.
4	A **list**.
5	A folder.
6	A site.
7	A site collection.

2.2.3.3 Calendar Type

Calendar type is a 2-byte integer value that specifies the type of calendar to use in a particular context. The only valid values of the **Calendar** type are specified as follows.

Value	Description
0	None
1	Gregorian (localized)
3	Japanese Emperor Era
4	Taiwan Calendar
5	Korean Tangun Era
6	Hijri (Arabic Lunar)
7	Thai
8	Hebrew (Lunar)
9	Gregorian (Middle East French)
10	Gregorian (Arabic)
11	Gregorian (Transliterated English)
12	Gregorian (Transliterated French)
14	Korean and Japanese Lunar
15	Chinese Lunar
16	Saka Era

2.2.3.4 Collation Order Enumeration

Collation Order Enumeration is a 2-byte integer value indicating **collation order** for textual information. The Collation Order values are mapped to Windows Collation Designator values, as specified in [Iseminger]. The only valid values of **Collation Order Enumeration** are specified as follows.

Value	Description
0	Albanian
1	Arabic
2	Chinese_PRC
3	Chinese_PRC_Stroke
4	Chinese_Taiwan_Bopomofo
5	Chinese_Taiwan_Stroke
6	Croatian

Value	Description
7	Cyrillic_General
8	Czech
9	Danish_Norwegian
10	Estonian
11	Finnish_Swedish
12	French
13	Georgian_Modern_Sort
14	German_PhoneBook
15	Greek
16	Hebrew
17	Hindi
18	Hungarian
19	Hungarian_Technical
20	Icelandic
21	Japanese
22	Japanese_Unicode
23	Korean_Wansung
24	Korean_Wansung_Unicode
25	Latin1_General
26	Latvian
27	Lithuanian
28	Lithuanian_Classic
29	Traditional_Spanish
30	Modern_Spanish
31	Polish
32	Romanian
33	Slovak
34	Slovenian
35	Thai
36	Turkish
37	Ukrainian
38	Vietnamese

Value	Description
39	Azeri_Cyrillic_90
40	Azeri_Latin_90
41	Chinese_Hong_Kong_Stroke_90
42	Divehi_90
43	Indic_General_90
44	Kazakh_90
45	Macedonian_FYROM_90
46	Syriac_90
47	Tatar_90
48	Uzbek_Latin_90

2.2.3.5 Event Host Type

Event Host type is a 4-byte, signed integer that specifies the type of object used as an **event host** for an **event receiver**. The only valid values of the **Event Host** type are specified as follows.

Value	Description
-1	The **Event Host** type is invalid.
0	The event host is a site collection.
1	The event host is a site.
2	The event host is a **list**.
3	The event host is a **list item**.
4	The event host is a content type.
5	The event host is a **workflow**.
6	The event host is a feature.

2.2.3.6 Event Receiver Type

Event Receiver type is a 32-bit signed integer that specifies the type of an **event receiver**, which specifies when the handler for the **event (2)** is invoked. The only valid values of the **Event Receiver** type are specified as follows.

Value	Description
1	The event receiver is invoked before a **list item** is added.
2	The event receiver is invoked before a list item is updated.

Value	Description
3	The event receiver is invoked before a list item is deleted.
4	The event receiver is invoked before a list item is checked in.
5	The event receiver is invoked before a list item is checked out.
6	The event receiver is invoked before a list item checkout is reverted.
7	The event receiver is invoked before an attachment to a list item is added.
8	The event receiver is invoked before an attachment to a list item is deleted.
9	The event receiver is invoked before a document is moved.
101	The event receiver is invoked before a field is added to the schema of a **list**.
102	The event receiver is invoked before a field is updated in the schema of a list.
103	The event receiver is invoked before a field is deleted from the schema of a list.
104	The event receiver is invoked before a list is added.
105	The event receiver is invoked before a list is deleted.
201	The event receiver is invoked before a site collection is deleted.
202	The event receiver is invoked before a site is deleted.
203	The event receiver is invoked before a site is moved.
204	The event receiver is invoked before a site is added.
501	The event receiver is invoked before a **workflow** is started.
10001	The event receiver is invoked after a list item is added.
10002	The event receiver is invoked after a list item is updated.
10003	The event receiver is invoked after a list item is deleted.
10004	The event receiver is invoked after a list item is checked in.
10005	The event receiver is invoked after a list item is checked out.
10006	The event receiver is invoked after a list item checkout is reverted.
10007	The event receiver is invoked after an attachment is added to a list item.
10008	The event receiver is invoked after an attachment is deleted from a list item.
10009	The event receiver is invoked after a document is moved.
10010	The event receiver is invoked after a document is transformed by the document transformation feature.
10101	The event receiver is invoked after a field is added to the schema of a list.
10102	The event receiver is invoked after a field is updated in the schema of a list.
10103	The event receiver is invoked after a field is deleted from the schema of a list.
10104	The event receiver is invoked after a list is added.

Value	Description
10105	The event receiver is invoked after a list is deleted.
10201	The event receiver is invoked after a site collection is deleted.
10202	The event receiver is invoked after a site is deleted.
10203	The event receiver is invoked after a site is moved.
10204	The event receiver is invoked after a site is provisioned.
10501	The event receiver is invoked after a workflow is started.
10502	The event receiver is invoked after a workflow is postponed.
10503	The event receiver is invoked after a workflow is completed.
20000	The event receiver is invoked when an email message is received by a list.
32766	The event receiver is context sensitive and inspects the **ContextType** (section 2.2.5.9) value to perform a corresponding action.
32767	The event receiver is used as part of a workflow.

2.2.3.7 Excluded Folder Type

Excluded Folder type is a 4-byte integer value that indicates folders that are excluded from common listings of the subfolders in a document library due to their special **roles**. The only valid values of **Excluded Folder** type are specified as follows.

Value	Description
0	No special handling.
1	Forms folder. This folder holds view pages within the document library or **list**.
2	Web images folder. This folder is named "_w" and holds image files in an image library.
3	Thumbnails folder. This folder is named "_t" and holds thumbnail images in an image library.
4	Root folder in a list or document library.

2.2.3.8 LinkDynamic Type

LinkDynamic type is a 1-byte value represented as a single, uppercase ASCII character that tracks various special link types. A **LinkDynamic Type** MUST have only one value at a time. A NULL value for **LinkDynamic** is used for a backward link. The only valid non-NULL values of **LinkDynamic** are specified as follows.

Value	Description
S	The URL is "static", which is the default and requires no special handling.
D	The URL is "dynamic", which is a link to <Site URL>/_vti_bin/shtml.dll/DirName/LeafName. Such links are used to invoke the FrontPage SmartHTML interpreter on a file.

Value	Description
L	The URL is to a layouts page (that is, it contains a path segment with the string "_layouts").
H	The URL is a history link (that is, it contains a path segment with the string "_vti_history").
G	A nonabsolute link from a ghosted document that does not fall into any other category.

2.2.3.9 LinkSecurity Type

LinkSecurity type is a 1-byte value represented as a single, uppercase ASCII character specifying the URI scheme for a link, such as HTTP or HTTPS. A **LinkSecurity** type MUST have only one value at a time. A NULL value for **LinkSecurity** is used for a backward link. The only valid non-NULL values of **LinkSecurity** are specified as follows.

Value	Description
H	The URL begins with "http://" (a nonsecure link using the http: scheme).
T	The URL begins with "shttp://" (an S-HTTP link using the Terisa's shttp: scheme, as specified in [RFC2660]).
S	The URL begins with "https://" (an SSL link using the https: or snews: scheme).
U	The URL is of another unknown scheme.

2.2.3.10 LinkType Types

LinkType type**s** is a 1-byte value represented as a single, uppercase ASCII character; it specifies type information about a link. A **LinkType** type MUST have only one value at a time. A NULL value for **LinkType** is used for a backward link. The only valid non-NULL values of a **LinkType** are specified as follows.

Value	Description
A	The link is from the ACTION attribute of an HTML form tag.
B	The link is from the attribute markup of a **bot**.
C	The link is from an autogenerated table of contents. Agents can ignore the link type when determining unreferenced files within a site.
D	The link references programmatic content, as in the HTML OBJECT or APPLET tags.
E	The link is from a cascading style sheet (CSS).
F	The link is from the SRC attribute of an HTML FRAME tag.
G	The link is to a dynamic web template for the containing document.
H	The link is from an HTML HREF attribute. This can also be used as a default link type value if a more precise type does not apply.
I	The link is to a document that the containing document includes via an include bot.
J	The link is from a field of this **list item**.

Value	Description
K	Identical to "H", except that the link contains an HTML bookmark specifier.
L	The link is a target in an HTML image map generated from an image map bot.
M	The link is to an image used in an HTML image map generated from an image map bot.
O	The link is part of a cross-page Web Part connection.
P	The link is part of the markup of a Web Part within the source of the containing document.
Q	The link references a CSS document that provides style information for the containing document.
R	The link is from the **MasterPageFile** attribute of the *@Page* directive in the containing document.
S	The link is from an HTML SRC attribute.
T	The link is to the index file used by a text search bot on this page.
V	The link is based on the properties of the document rather than anything in the **document stream**. The link type is used in tracking the link between a site and the master page URL used for the site.
X	The link is from an XML island within an HTML document.
Y	The link references an HTML document whose HTML BODY tag attributes are used as a template for the attributes of the containing document's BODY tag.
Z	The link is part of the markup of a Web Part that exists in a Web Part zone in the containing document and is consequently not stored within the source of the containing document.

2.2.3.11 List Base Type

List Base type is a 32-bit integer enumeration of possible base types for **lists**. All lists are created with one of these base types, which define implementation-specific common values for list properties. The only valid values of the **List Base** type are specified as follows.

Value	Description
0	Generic list
1	Document library
3	Discussion board list
4	Survey list
5	Issues list

2.2.3.12 List Server Template

List Server Template is a 32-bit integer enumeration of the possible values for the **List Server Template** defining the base structure of a **list**. Reserved values of the **List Server Template** are specified as follows.

Value	Description
-1	Invalid Template
100	Generic List Template
101	Document Library Template
102	Survey Template
103	Links Template
104	Announcements Template
105	Contacts Template
106	Events Template
107	Tasks Template
108	Discussion Template
109	Image Library Template
110	Data Sources Template
111	Web Template Catalog Template
112	User Info Catalog Template
113	Web Part Catalog Template
114	List Template Catalog Template
115	XML Form Template
116	Master Page Catalog Template
117	No Code Workflows Template
118	Workflow Process Template
119	Webpage Library Template
120	Custom Grid Template
130	Data Connection Library Template
140	Workflow History Template
150	Gantt Tasks Template
200	Meetings Template
201	Agenda Template
202	Meeting user Template
204	Decision (Meeting) Template
207	Meeting Objective Template
210	Textbox Template
211	Things To Bring (Meeting) Template

Value	Description
212	Homepage Library Template
301	Posts (Blog) Template
302	Comments (Blog) Template
303	Categories (Blog) Template
402	Resources List Template
403	Whereabouts List Template
404	Phone Call Memo List Template
405	Circulation List Template
420	Time Card Template
421	Holidays List Template
499	Microsoft Office IME Dictionary List Template
1100	Issue Tracking Template
1200	Administration Tasks List Template
1220	Health Rules List Template
1221	Health Reports List Template

2.2.3.13 Moderation Status

Moderation Status is a 4-byte integer indicating the moderation approval status of a **list item**. Configurations can require moderation approval to publish a list item or allow automatic approval. A published list item MUST have a **Moderation Status** of 0. The following are all possible valid values for **Moderation Status**.

Value	Description
0	The list item is approved.
1	The list item has been denied approval.
2	The list item is pending approval.
3	The list item is in the draft or checked out state.
4	The list item is scheduled for automatic approval at a future date.

2.2.3.14 Page Type

Page type is a signed 1-byte integer that is used to represent the possible page types. The reserved **Page** type values are specified as follows.

Value	Description
-1	Does not correspond to a view or a form of a **list**.
0	Default view of the corresponding list. This view is displayed whenever this list is viewed without an explicit view being specified.
1	A view of the corresponding list, but not the default view.
2	This value is only used internally within implementation-specific code and is never stored in a database.
3	This value is only used internally within implementation-specific code and is never stored in a database.
4	A display form of a list, suitable for displaying a single list item in **read-only mode**.
5	This value is only used internally within implementation-specific code and is never stored in a database.
6	An edit form for a list, suitable for presenting UI to update the properties of a list item.
7	Used to represent edit forms of a list suitable for displaying in HTML file dialogs to a client application.
8	A new form for a list, suitable for presenting UI to create a new list item.
9	This value is from a previous implementation and is no longer valid.
10	This value is only used internally within implementation-specific code and is never stored in a database.

2.2.3.15 Redirect Type

Redirect type is a 1-byte value corresponding to the type of item that a URL is redirected to. The following table contains all possible values for **Redirect** type.

Value	Description
0	Welcome page.
1	Homepage.
2	**List view** - redirect to root folder.
3	Provision - redirect to a template picker page during provisioning of a new **list**.
255	None.

2.2.3.16 Role Definition Type

Role Definition type is a 1-byte value that is used to represent the type of implementation-specific default and custom role definitions. This integer value MUST be a value enumerated in the following table.

Value	Description
0	A custom-defined role
1	Guest
2	Reader
3	Contributor
4	Web Designer
5	Administrator

2.2.3.17 Time Zone Identifier

Time Zone Identifier is a 2-byte integer value identifying a time zone. The only valid values of the Time Zone Identifier are specified as follows.

Value	Meaning
2	(GMT) Greenwich Mean Time : Dublin, Edinburgh, Lisbon, London
3	(GMT+01:00) Brussels, Copenhagen, Madrid, Paris
4	(GMT+01:00) Amsterdam, Berlin, Bern, Rome, Stockholm, Vienna
5	(GMT+02:00) Athens, Bucharest, Istanbul
6	(GMT+01:00) Belgrade, Bratislava, Budapest, Ljubljana, Prague
7	(GMT+02:00) Minsk
8	(GMT-03:00) Brasilia
9	(GMT-04:00) Atlantic Time (Canada)
10	(GMT-05:00) Eastern Time (U.S. and Canada)
11	(GMT-06:00) Central Time (U.S. and Canada)
12	(GMT-07:00) Mountain Time (U.S. and Canada)
13	(GMT-08:00) Pacific Time (U.S. and Canada)
14	(GMT-09:00) Alaska
15	(GMT-10:00) Hawaii
16	(GMT-11:00) Midway Island, Samoa
17	(GMT+12:00) Auckland, Wellington
18	(GMT+10:00) Brisbane
19	(GMT+09:30) Adelaide
20	(GMT+09:00) Osaka, Sapporo, Tokyo
21	(GMT+08:00) Kuala Lumpur, Singapore

Value	Meaning
22	(GMT+07:00) Bangkok, Hanoi, Jakarta
23	(GMT+05:30) Chennai, Kolkata, Mumbai, New Delhi
24	(GMT+04:00) Abu Dhabi, Muscat
25	(GMT+03:30) Tehran
26	(GMT+03:00) Baghdad
27	(GMT+02:00) Jerusalem
28	(GMT-03:30) Newfoundland
29	(GMT-01:00) Azores
30	(GMT-02:00) Mid-Atlantic
31	(GMT) Casablanca, Monrovia, Reykjavik
32	(GMT-03:00) Buenos Aires, Georgetown
33	(GMT-04:00) Caracas, La Paz
34	(GMT-05:00) Indiana (East)
35	(GMT-05:00) Bogota, Lima, Quito, Rio Branco
36	(GMT-06:00) Saskatchewan
37	(GMT-06:00) Guadalajara, Mexico City, Monterrey
38	(GMT-07:00) Arizona
39	(GMT-12:00) International Date Line West
40	(GMT+12:00) Fiji Is., Kamchatka, Marshall Is.
41	(GMT+11:00) Magadan, Solomon Is., New Caledonia
42	(GMT+10:00) Hobart
43	(GMT+10:00) Guam, Port Moresby
44	(GMT+09:30) Darwin
45	(GMT+08:00) Beijing, Chongqing, Hong Kong S.A.R., Urumqi
46	(GMT+06:00) Almaty, Novosibirsk
47	(GMT+05:00) Islamabad, Karachi, Toshkent
48	(GMT+04:30) Kabul
49	(GMT+02:00) Cairo
50	(GMT+02:00) Harare, Pretoria
51	(GMT+03:00) Moscow, St. Petersburg, Volgograd
53	(GMT-01:00) Cabo Verde Is.
54	(GMT+04:00) Baku

Value	Meaning
55	(GMT-06:00) Central America
56	(GMT+03:00) Nairobi
57	(GMT+01:00) Sarajevo, Skopje, Warsaw, Zagreb
58	(GMT+05:00) Ekaterinburg
59	(GMT+02:00) Helsinki, Kyiv, Riga, Sofia, Tallinn, Vilnius
60	(GMT-03:00) Greenland
61	(GMT+06:30) Yangon (Rangoon)
62	(GMT+05:45) Kathmandu
63	(GMT+08:00) Irkutsk, Ulaan Bataar
64	(GMT+07:00) Krasnoyarsk
65	(GMT-04:00) Santiago
66	(GMT+05:30) Sri Jayawardenepura
67	(GMT+13:00) Nuku'alofa
68	(GMT+10:00) Vladivostok
69	(GMT+01:00) West Central Africa
70	(GMT+09:00) Yakutsk
71	(GMT+06:00) Astana, Dhaka
72	(GMT+09:00) Seoul
73	(GMT+08:00) Perth
74	(GMT+03:00) Kuwait, Riyadh
75	(GMT+08:00) Taipei
76	(GMT+10:00) Canberra, Melbourne, Sydney
77	(GMT-07:00) Chihuahua, La Paz, Mazatlan
78	(GMT-08:00) Tijuana, Baja California
79	(GMT+02:00) Amman
80	(GMT+02:00) Beirut
81	(GMT-04:00) Manaus
82	(GMT+03:00) Tbilisi
83	(GMT+02:00) Windhoek
84	(GMT+04:00) Yerevan

2.2.3.18 Virus Status

Virus Status is a 4-byte, integer enumerated type that specifies the current virus scan status of a document. The following are valid values for **Virus Status**.

Value	Description
0	This document is reported as clean from viruses.
1	This document had a virus reported by the **virus scanner** plug-in.
2	This document had a virus reported by the virus scanner plug-in, which the scanner determines that it can remove.
3	This document had a virus previously reported, but the virus scanner determines that it successfully removed it.
4	This document had a virus reported, and the virus scanner attempted to clean it but failed.
5	This document had a virus reported, and the scanner requested that the document be deleted.
6	This document had a timeout from the virus scanner when it was last processed.

2.2.4 Binary Structures

2.2.4.1 Calendar View Options Type

Calendar View Options type is a 1-byte value that specifies calendar options for front-end web server display in the following format.

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
FirstDayOfWeek			FirstWeekOfYear		WeekUI	Unused																									

FirstDayOfWeek (3 bits): An unsigned integer specifying the first day of the week. The following are valid values for the **FirstDayOfWeek** value.

Bits	Description
000	Sunday
001	Monday
010	Tuesday
011	Wednesday
100	Thursday
101	Friday
110	Saturday

FirstWeekOfYear (2 bits): An unsigned integer specifying how the first week of the year SHOULD be handled. The following are valid values for the **FirstWeekOfYear** value.

Bits	Description
00	The year starts on January 1.
01	The year starts with the first complete week.
10	The year starts with the first week of at least four days.

WeekUI (1 bit): If this bit is set, week numbers SHOULD be displayed in the front-end web server. Otherwise, week numbers will not be displayed in the front-end web server.

Unused (2 bits): The last 2 bits of this structure are currently unused and MUST both be set to 0.

2.2.4.2 External Group Token

The **External Group Token** structure is a variable-length structure associated with a **principal** that contains a collection of the **SystemIDs** (section 2.2.1.12) for the external **groups** of which the principal is a **member**.

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
TimeTokenGenerated																															
...																															
Size																															
Magic																															
AuthenticationType																															
UserSystemIdSize																															
TokenGroupsSize																															
Magic2																															
UserSystemId (variable)																															
...																															
TokenGroups (variable)																															
...																															

TimeTokenGenerated (8 bytes): An 8-byte, unsigned integer value specifying the time that this token was generated, stored as seconds since midnight, January 1, 1899, **Coordinated Universal Time (UTC)**.

Size (4 bytes): A 4-byte, unsigned integer value specifying the length of the **External Group Token** in bytes.

Magic (4 bytes): A 4-byte, unsigned integer, which MUST use the value 0xCACBCECF.

AuthenticationType (4 bytes): A 4-byte, unsigned integer value specifying the authentication provider for the **SystemID** of the principal. The value MUST be one of the following.

Value	Description
0x00000001	Windows Integrated Authentication
0x00000003	ASP.NET Forms Authentication

UserSystemIdSize (4 bytes): A 4-byte, unsigned integer value specifying the length of the principal's serialized binary **SystemID**, in bytes.

TokenGroupsSize (4 bytes): A 4-byte, unsigned integer value specifying the length, in bytes, of the TokenGroups field containing the serialized binary **SystemIDs** for the external groups.

Magic2 (4 bytes): A 4-byte, unsigned integer, which MUST use the value 0xDADBDEDF.

UserSystemId (variable): A variable-length field containing the serialized binary **SystemID** for the principal, occupying the number of bytes specified in **UserSystemIdSize**.

TokenGroups (variable): A variable-length field containing a **Token Groups** structure, which contains the serialized binary **SystemIDs** for the external groups of which the principal is a member, occupying the number of bytes specified in **TokenGroupsSize**.

2.2.4.3 Token Group Offset and Attributes

The **Token Group Offset and Attributes** structure specifies the length and offset of the serialized binary **SystemID** (section 2.2.1.12) for a corresponding external **group** within a **Token Groups** structure.

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
Offset																															
Attributes																															

Offset (4 bytes): A 4-byte, unsigned integer value specifying the offset in bytes from the beginning of the **Token Groups** structure to the beginning of the serialized binary **SystemID** for this external group.

Attributes (4 bytes): A 4-byte, unsigned integer value specifying the length in bytes of the serialized binary **SystemID** for this external group.

2.2.4.4 Token Groups

The **Token Groups** structure contains the serialized binary **SystemIDs** (section 2.2.1.12) for a collection of external **groups**.

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
GroupCount																															
OffsetsAndAttributes (variable)																															
...																															
GroupSystemIds (variable)																															
...																															

GroupCount (4 bytes): A 4-byte, unsigned integer value specifying the number of external groups **SystemIDs** that are serialized in this structure.

OffsetsAndAttributes (variable): A variable-length serialized array containing **GroupCount** elements consisting of **Token Group Offset and Attributes** structures specifying the lengths and offsets of the serialized binary **SystemIDs** for the corresponding external groups, one for each external group's **SystemID** in the collection.

GroupSystemIds (variable): A variable-length field containing the collection of serialized binary **SystemIDs** for the external groups. Each external group's **SystemID** starts at the offset value in bytes from the beginning of the Token Groups structure and has the length in bytes specified by the Attributes value in the OffsetsAndAttributes array element corresponding to the external group.

2.2.4.5 WSS ACE

An **ACE** structure specifying the individual access rights of a **principal**.

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
PrincipalId																															
PermMask																															
...																															

PrincipalId (4 bytes): A 4-byte, signed integer specifying the user identifier (section 2.2.1.13) of the principal for this ACE.

PermMask (8 bytes): A **WSS Rights Mask** (section 2.2.2.14) containing the **list** of rights that SHOULD be granted to the principal.

2.2.4.6 WSS ACL Format

The **WSS ACL Format** structure contains an array of WSS_ACEs.

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
Magic																															
SecurityVersion																															
...																															
NumAces																															
Aces (variable)																															
...																															

Magic (4 bytes): A 4-byte, unsigned integer describing the version of the ACL. This version of the protocol MUST use the value 0xfef3.

SecurityVersion (8 bytes): An 8-byte, signed integer specifying the site collection's security version value, which was used to compute the ACL. This value is not currently used.

NumAces (4 bytes): A 4-byte, unsigned integer specifying the count of **WSS_ACEs** within this ACL.

Aces (variable): An array of **WSS_ACEs** for each of the **principals** in this ACL.

2.2.4.7 WSS External Group Map Cache Format

The **WSS External Group Map Cache Format** structure contains a cache of **WSS External Group Records** mapping external **groups** to the site groups that contain them.

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
CachedVersion																															
...																															
NumExternalGroupRecords																															
ExternalGroupRecords (variable)																															
...																															

CachedVersion (8 bytes): An 8-byte, signed integer specifying the version of this cached information.

NumExternalGroupRecords (4 bytes): A 4-byte, signed integer specifying the count of **WSS External Group Records** present in this cache.

ExternalGroupRecords (variable): A serialized collection of **WSS External Group Records**.

2.2.4.8 WSS Compressed Structures

The **WSS Compressed** structure uses the ZLIB Compressed Data Format Specification version 3.3 to compress a binary or string value to a binary format when save it in to the database.

The header of the **WSS Compressed** structure is specified as follows.

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
ID																Version															
FileHeaderSize																															
OrigSize																															
Compressed Binary string (variable)																															
...																															

ID (2 bytes): MUST be 0xA8A9.

Version (2 bytes): MUST be 0x3031 (ASCII "01")

FileHeaderSize (4 bytes): A 4-byte, unsigned integer specifying the size of the header. It is 0x0C000000.

OrigSize (4 bytes): A 4-byte, unsigned integer specifying the size of the original content. It MUST be the size of the uncompressed **stream** before compression.

Compressed Binary string (variable): The compressed string using ZLIB compression.

2.2.4.9 WSS External Group Record

The **WSS External Group Record** structure contains the mapping between an external **group** and the site groups that contain it.

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
ExternalGroupSignatureSize																															
ExternalGroupSignature (variable)																															
...																															
NumGroupIds																															
GroupIds (variable)																															
...																															

ExternalGroupSignatureSize (4 bytes): A 4-byte, signed integer specifying the size of the external group signature, in bytes.

ExternalGroupSignature (variable): An array of bytes specifying the signature for the external group using **UTF-8** encoding. The number of bytes is specified by the **ExternalGroupSignatureSize** field.

NumGroupIds (4 bytes): A 4-byte, signed integer specifying the count of site group identifiers (section 2.2.1.10) in this record.

GroupIds (variable): An array of 4-byte, signed integers specifying the site group identifiers that contain this external group. The number of elements is specified by the **NumGroupIds** field.

2.2.4.10 WSS User Token

The **WSS user Token** structure contains an array of site **group** identifiers (section 2.2.1.10).

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
Magic																															
TokenVersion																															
...																															
NumGroupIds																															
GroupIds (variable)																															
...																															

Magic (4 bytes): A 4-byte, unsigned integer specifying the version of the token format. This version of the protocol MUST use the value 0xdcd3.

TokenVersion (8 bytes): An 8-byte, signed integer specifying the site collection's security version value, which was used to compute the token. This value is not currently used and MUST be ignored.

NumGroupIds (4 bytes): A 4-byte, unsigned integer specifying the count of site group identifiers within this token.

GroupIds (variable): An array of 4-byte integers for each of the site groups the corresponding user belongs to. The number of elements in the array is specified by the **NumGroupIds** field.

2.2.5 Result Sets

The following common result sets are used by this protocol.

2.2.5.1 ACL and Permission Result Set

The ACL and Permission Result Set contains information about the permissions associated with a **security scope** in effect for a document. The ACL and Permission Result Set is defined using T-SQL syntax, as follows.

```
Acl                varbinary(max),
AnonymousPermMask bigint;
```

Acl: The binary serialization of the **WSS ACL Format** (section 2.2.4.6) **access control list (ACL)** for the security scope in effect.

AnonymousPermMask: A **WSS Rights Mask** (section 2.2.2.14) indicating the permissions granted to an anonymous user or a user who has no specific permissions on this document.

2.2.5.2 Custom Actions From Scope Result Set

This result set MUST return 1 row for each custom action retrieved. If there were no custom actions retrieved, this result set MUST NOT return any rows. This result set is defined using T-SQL syntax, as follows.

```
ScopeType           int;
ScopeId             uniqueidentifier;
Id                  uniqueidentifier;
Properties          nvarchar(max);
Version             nvarchar(64);
```

ScopeType: The custom action scope.

ScopeId: The site collection identifier (section 2.2.1.9), site identifier (section 2.2.1.11), or list identifier (section 2.2.1.5) for which the custom action resides.

Id: The custom action identifier.

Properties: The custom action data describing its functionality.

Version: The custom action version.

2.2.5.3 Domain Group Cache BEDS Update Result Set

The Domain Group Cache BEDS Update Result Set contains information to be used in recomputing the **domain group** map cache, which contains the mapping of external **groups** to the site groups that they are members of. The presence of the Domain Group Cache BEDS Update Result Set means the database's copy of the domain group map cache is out of date and MUST be recomputed to ensure that proper security checks can be made.

The Domain Group Cache BEDS Update Result Set is defined using T-SQL syntax, as follows.

```
tp_id                               int,
tp_SystemId                         varbinary(512),

GroupId                             int;
```

tp_id: The identifier of an external group which is a **member** of a Site Group.

tp_SystemId: The **SystemID** (section 2.2.1.12) of the external group.

GroupId: The site group identifier (section 2.2.1.10) of the Site Group containing the given domain group.

2.2.5.4 Domain Group Cache Versions Result Set

The Domain Group Cache Versions Result Set contains information about the version numbers associated with the Domain Group Map Caches on the front-end web server and on the back-end database server for the specified Site Collection. A Domain Group Map Cache contains a serialized representation of the External Groups that are members of Site Groups in the Site Collection. The version numbers in this result set can be used to determine whether the Domain Group Map Caches on the front-end web server or on the back-end database server are out of date and need to be refreshed. A special version number value of -2 indicates that the value MUST NOT be used for comparison.

When the Domain Group Cache Versions Result Set is returned, it MUST contain one row of version number data.

The Domain Group Cache Versions Result Set is defined using T-SQL syntax, as follows.

```
    RealVersion                     bigint,
    CachedVersion                   bigint,
    FrontEndVersion                 bigint;
```

RealVersion: The most recent version number of the domain group Map Cache information.

CachedVersion: The version number of the domain group Map Cache information on the back-end database server.

FrontEndVersion: The version number of the domain group Map Cache information on the front-end web server.

2.2.5.5 Domain Group Cache WFE Update Result Set

The Domain Group Cache WFE Update Result Set contains the binary data needed to refresh the domain group map cache. If the Domain Group Cache WFE Update Result Set is returned, it indicates that the Back-end database server domain group map cache is up-to-date, and the front-end web server cache can be refreshed if necessary.

The Domain Group Cache WFE Update Result Set is defined using T-SQL syntax, as follows.

```
    DomainGroupMapCache              varbinary(max);
```

DomainGroupMapCache: A column containing the serialized domain group map cache data. If the value of FrontEndVersion is greater than or equal to the value of CachedVersion in the results in the Domain Group Cache Versions Result Set (section 2.2.5.4), this MUST be NULL.

2.2.5.6 Document Metadata Result Set

The Document Metadata Result Set returns the metadata for a document.

The Document Metadata Result Set is defined using T-SQL syntax, as follows.

```
    Id                              uniqueidentifier,
    {FullUrl}                       nvarchar(260),
    Type                            tinyint,
    MetaInfoTimeLastModified        datetime,
    MetaInfo                        varbinary(max),
    Size                            int,
    TimeCreated                     datetime,
    TimeLastModified                datetime,
    ETagVersion                   int,
    DocFlags                      int,
    {ListType}                    int,
    {tp_Name}                     int,
    {ListTitle}                   int,
    {CacheParseId}                uniqueidentifier,
    {GhostDirName}                int,
    {GhostLeafName}               int,
    tp_Login                      nvarchar(255),
    CheckoutDate                  datetime,
    CheckoutExpires               datetime,
    VirusStatus                   int,
    VirusInfo                     nvarchar(255),
    SetupPathVersion              tinyint,
    SetupPath                     nvarchar(255),
    SetupPathUser                 nvarchar(255),
    NextToLastTimeModified        datetime,
    UIVersion                     int,
    CheckinComment                nvarchar(1023),
    WelcomePageUrl                nvarchar(260),
    WelcomePageParameters         nvarchar(max),
    {tp_Flags}                    int,
    Acl                           varbinary(max),
    AnonymousPermMask             bigint,
    DraftOwnerId                  int,
    Level                         tinyint,
    ParentVersion                 int,
    TransformerId                 uniqueidentifier,
    ParentLeafName                nvarchar(128),
    ProgId                        nvarchar(255),
    DoclibRowId                   int,
    {tp_DefaultWorkflowId}        int,
    ListId                        uniqueidentifier,
    ItemChildCount                int,
    FolderChildCount              int,
    MetaInfoVersion               int,
    {CurrentVerMetaInfo}          varbinary(max),
    ContentVersion                int,
    {ContentVersionDirty}         bit;
```

Id: The **document identifier** (section 2.2.1.2) of the requested document.

{FullUrl}: The full **store-relative form** URL for the document being requested.

Type: The **Document Store** type (section 2.2.2.4) of this document.

MetaInfoTimeLastModified: A time stamp in **UTC** format specifying the last time the **MetaInfo** value of this document was changed, which is useful for providing minimal metadata returns to clients. This value MUST be NULL if the **MetaInfo** column value of the document has never been changed.

MetaInfo: A METADICT for this document. (The METADICT format is specified in [MS-FPSE] section 2.2.2.2.11). This value MUST be NULL if the document does not have any METADICT associated with it.

Size: The number of bytes in the document stream of the document. This value can be NULL if the document is a folder or site **Document Store** type.

TimeCreated: A time stamp in UTC format that specifies when this document was created.

TimeLastModified: A time stamp in UTC format that specifies when the document was last saved. This value does not necessarily correspond to the actual time when the document was last modified.

ETagVersion: A counter incremented any time a change is made to this document and used for internal conflict detection.

DocFlags: A **Doc Flags** (section 2.2.2.3) value describing the document. This value can be NULL.

{ListType}: This value MUST be NULL.

{tp_Name}: This value MUST be NULL.

{ListTitle}: This value MUST be NULL.

{CacheParseId}: Used for concurrency detection if two different requests attempt to perform dependency update or undirtying on a document at the same time. If *@CheckCacheParseId* is set to "1", this field MUST reflect the value stored for the document. Otherwise, it MUST return NULL.

{GhostDirName}: A placeholder for a **directory name**. This value MUST be NULL.

{GhostLeafName}: A placeholder for a **leaf name**. This value MUST be NULL.

tp_Login: If this document exists and is currently checked out, then this is the **login name** of the user to whom it is checked out. In all other cases, this is NULL.

CheckoutDate: A time stamp in UTC format specifying when this document was checked out. This value MUST be NULL if the document has never been checked out.

CheckoutExpires: A time stamp in UTC format that specifies when the **short-term lock** for this document will expire. If this date is in the past, the document SHOULD be treated as unlocked. This value can be NULL if no short-term lock has been placed on the document.

VirusStatus: An enumerated type specifying the current virus state of this document. This value can be NULL if it has not been processed by a virus scanner. Valid values are listed in **Virus Status** (section 2.2.3.18).

VirusInfo: A string containing a provider specific message returned by the virus scanner when it last processed the document. This value can be NULL if it has not been processed by a virus scanner.

SetupPathVersion: For a ghosted document, this governs the **setup path** location with which the **SetupPath** fragment is relative. This value MUST NOT be NULL. The following values are valid.

Value	Description
2	The **SetupPath** is relative to the install location of Windows SharePoint Services 2.0 on the front-end web server (for example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\60).
3	The **SetupPath** is relative to the install location of Windows SharePoint Services 3.0 on the front-end web server (for example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\12).
4	The **SetupPath** is relative to the install location of Microsoft SharePoint Foundation 2010 on the front-end web server (for example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\14).

SetupPath: For a document that is now or once was ghosted, this contains the setup path fragment relative to the base setup path described earlier by the **SetupPathVersion** value, where the content **stream** of this document can be found. This value can be NULL.

SetupPathUser: If this document is now or once was ghosted, this contains the login name of the user that created the ghosted document. This value can be NULL.

NextToLastTimeModified: The value of **TimeLastModified** from the previous time when the document was last saved. If the **NextToLastTimeModified** value matches the **TimeLastModified** value when the rename occurred, and the client has a document that it has successfully fixed up, the client can safely submit the document to the front-end web server despite what appears to be an intervening edit to the document. This value can be NULL.

UIVersion: The UI version number for the document. This value MUST be NULL in the case of a document that does not exist.

CheckinComment: An optional description provided when a document is checked in or published, which could be displayed in version management UI. This value can be NULL.

WelcomePageUrl: If this document is a folder, this specifies an optional page to redirect to when the folder is requested with an **HTTP GET** operation. The URL is relative to the URL of the folder itself and MUST be subsumed by that folder. Attempts to break out of the folder such as "../../somepage.aspx" are not valid. This value can be NULL.

WelcomePageParameters: Contains optional **URL** parameters to specify as part of the WelcomePageUrl value. These parameters start at either the query string signifier '?' or the bookmark signifier '#'. This value can be NULL.

{tp_Flags}: A **List Flags** (section 2.2.2.5) value describing the **list** that contains this document.

Acl: The binary serialization of the **WSS ACL Format** (section 2.2.4.6) ACL for this document. This is either explicitly defined or inherited from the parent object of the document. This value can be NULL if a WSS ACL is not defined for the document.

AnonymousPermMask: A **WSS Rights Mask** (section 2.2.2.14) that indicates the permissions granted to a user that is anonymous, or has no specific permissions, to this document. This value can be NULL if anonymous access to the document is not allowed.

DraftOwnerId: The user identifier (section 2.2.1.13) of the user that published this document as a draft. This value MUST only be non-NULL if the requested document exists and is a draft **version**.

Level: A **Publishing Level** type (section 2.2.2.6) value specifying the publishing status of this document.

ParentVersion: If the document is a transformed version of another document, this is the user interface (UI) version value from the original document. This value MUST be NULL if the document is not the product of a document transformation.

TransformerId: If the document is a transformed version of another document, this is the GUID of the agent that performed the transformation. This value MUST be NULL if the document is not the product of a document transformation.

ParentLeafName: If the document is a transformed version of another document, this is the leaf name of the original document, if the original document is in the same folder as the transformed document. If either is moved, the relationship is broken. This value MUST be NULL if the document is not the product of a document transformation.

ProgId: Designates a preferred application to open the document. The **ProgId** is used to distinguish between different applications that save files with a given **file extension** (that is, different editing applications for HTML or XML files). This value MUST be NULL if the parser did not specify a **ProgId** when the document was saved.

DoclibRowId: The document library row identifier for this document. If the requested document is not contained in a list, this value MUST be NULL.

{tp_DefaultWorkflowId}: The workflow identifier (section 2.2.1.16) corresponding to the **workflow** to be invoked if this document is in a moderated list and this document is submitted for approval as part of a **check in**. This value MUST be NULL.

ListId: The list identifier (section 2.2.1.5) of the list that contains the requested document. If the document is not contained in a list, this value MUST be NULL.

ItemChildCount: The number of non-folder (those whose **Document Store** type is not folder) children of this document.

FolderChildCount: The number of folder (those whose **Document Store** type is folder) children of this document.

MetaInfoVersion: A counter incremented any time a change is made to the METADICT for this document and used for internal conflict detection.

{CurrentVerMetaInfo}: This value is not used and MUST be NULL.

ContentVersion: A counter incremented any time a change is made to the binary contents of this document; used for internal conflict detection. This value MUST be NULL if the document does not exist.

{ContentVersionIsDirty}: This value specifies whether the **ContentVersion** value can be used for conflict detection. If the **ContentVersion** value can be used for conflict detection or the **ContentVersion** value is NULL, this value MUST be 0; otherwise it MUST be 1.

2.2.5.7 Document Version Metadata Result Set

The Document Version Metadata Result Set contains the metadata about the requested version of the document.

The Document Version Metadata Result Set MUST only be returned if *@Version* is not negative. If the document exists, the Document Version Metadata Result Set MUST contain one row; otherwise, it MUST contain no rows.

The Document Version Metadata Result Set is defined using T-SQL syntax, as follows.

```
Id                              uniqueidentifier,
{FullUrl}                       nvarchar(260),
Type                            tinyint,
MetaInfoTimeLastModified        datetime,
MetaInfo                        varbinary(max),
Size                            int,
TimeCreated                     datetime,
TimeLastModified                datetime,
ETagVersion                     int,
DocFlags                        int,
{ListType}                      int,
{tp_Name}                       nvarchar(38),
{ListTitle}                     nvarchar(255),
{CacheParseId}                  uniqueidentifier,

{GhostDirName}                  nvarchar(256),
{GhostLeafName}                 nvarchar(128),
tp_Login                        nvarchar(255),
CheckoutDate                    datetime,
CheckoutExpires                 datetime,
VirusStatus                     int,
VirusInfo                       nvarchar(255),
SetupPathVersion                tinyint,
SetupPath                       nvarchar(255),
SetupPathUser                   nvarchar(255),
NextToLastTimeModified          datetime,
UIVersion                       int,
CheckinComment                  nvarchar(1023),
WelcomePageUrl                  nvarchar(260),
WelcomePageParameters           nvarchar(max),
{tp_Flags}                      bigint,
Acl                             varbinary(max),
AnonymousPermMask               int,
DraftOwnerId                    int,
Level                          tinyint,
ParentVersion                  int,
TransformerId                  uniqueidentifier,
ParentLeafName                 nvarchar(128),
ProgId                         nvarchar(255),
DoclibRowId                    int,
{tp_DefaultWorkflowId}         uniqueidentifier,
ListId                         uniqueidentifier;
ItemChildCount                 int;
FolderChildCount               int;
MetaInfoVersion                int;
{MetaInfo}                     varbinary(MAX),
ContentVersion                 int,
{ContentVersionDirty}          bit;
```

Id: The **document identifier** (section 2.2.1.2) of the requested document.

{FullUrl}: The full store-relative form URL for the document being requested.

Type: The **Document Store** type (section 2.2.2.4) of this document.

MetaInfoTimeLastModified: A time stamp in **UTC** format specifying the last time the metainfo value of this document was changed. This value MUST be NULL if the Metainfo column value of the document has never been changed.

MetaInfo: A METADICT for this **document version**. The METADICT format is specified in [MS-FPSE] section 2.2.2.2.11. This value MUST be NULL if the document does not have any METADICT associated with it.

Size: The number of bytes in the document stream of the document version requested. This value can be NULL if the document is a folder or site **Document Store** type.

TimeCreated: A time stamp in UTC format specifying when this document was created.

TimeLastModified: A time stamp in UTC format specifying when the document was last saved. This corresponds to the TimeCreated or TimeLastModified of the document version requested.

ETagVersion: A counter incremented any time a change is made to this document and used for internal conflict detection.

DocFlags: A **Doc Flags** (section 2.2.2.3) value describing this document. This value can be NULL if the document is a folder **Document Store** type.

{ListType}: This value MUST be NULL.

{tp_Name}: This value MUST be NULL.

{ListTitle}: This value MUST be NULL.

{CacheParseId}: Used for concurrency detection if two different requests attempt to perform dependency update or undirtying on a document at the same time. If *@CheckCacheParseId* is set to "1", this field MUST reflect the value stored for the document. Otherwise, it MUST return NULL.

{GhostDirName}: A placeholder for a directory name. This value MUST be NULL.

{GhostLeafName}: A placeholder for a leaf name. This value MUST be NULL.

tp_Login: If this document exists and is currently checked out, then this is the login name of the user to whom it is checked out. In all other cases, this is NULL.

CheckoutDate: A time stamp in UTC format specifying when this document was checked out. This value can be NULL if the document is checked out.

CheckoutExpires: A time stamp in UTC format specifying when the **short-term lock** for this document will expire. If this date is in the past, the document SHOULD be treated as unlocked. This value can be NULL if no short-term lock has been placed on the document.

VirusStatus: An enumerated type specifying the current virus state of this document version. This value can be NULL if it has not been processed by a virus scanner. Valid values are listed in **Virus Status** (section 2.2.3.18).

VirusInfo: A string containing a provider specific message returned by the virus scanner when it last processed the document. This value can be NULL if it has not been processed by a virus scanner.

SetupPathVersion: For a ghosted document, this governs the setup path location that the **SetupPath** fragment is relative to. This value MUST NOT be NULL. The following are values are valid.

Value	Description
2	The **SetupPath** is relative to the install location of Windows SharePoint Services 2.0 on the front-end web server (for example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\60).
3	The **SetupPath** is relative to the install location of Windows SharePoint Services 3.0 on the front-end web server (for example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\12).
4	The **SetupPath** is relative to the install location of Microsoft SharePoint Foundation 2010 on the front-end web server (for example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\14).

SetupPath: For a document that is now or once was ghosted, this contains the setup path fragment relative to the base setup path described earlier by the **SetupPathVersion** value, where the content **stream** of this document can be found. This value can be NULL.

SetupPathUser: If this document is now or once was ghosted, then this contains the login name of the user that created the ghosted document. This value can be NULL.

NextToLastTimeModified: The value of **TimeLastModified** from the previous time when the document was last saved. If the **NextToLastTimeModified** value matches the **TimeLastModified** value when the rename occurred and the client has a document that it has successfully fixed up, the client can safely submit the document to the front-end web server despite what appears to be an intervening edit to the document. This value can be NULL.

UIVersion: The user interface (UI) version number for the document. This value can be produced as the integer counter described earlier (the first **Version** field). This value MUST match *@Version*.

CheckinComment: An optional description provided when a document is checked in, which could be displayed in version management UI. This value can be NULL.

WelcomePageUrl: If this document is a folder, then this specifies an optional page to redirect to when the folder is requested with an HTTP GET operation. The URL is relative to the URL of the folder itself and MUST be subsumed by that folder. Attempts to break out of the folder such as "../../somepage.aspx" are not valid. This value can be NULL.

WelcomePageParameters: Contains optional URL parameters to specify as part of the WelcomePageUrl value. These parameters start at either the query string signifier "?" or the bookmark signifier "#". This value can be NULL.

{tp_Flags}: A **List Flags** (section 2.2.2.5) value describing the **list** that contains this document. This value MUST be NULL.

Acl: The binary serialization of the **WSS ACL Format** (section 2.2.4.6) ACL for this document. This is either explicitly defined or inherited from the parent object of the document.

AnonymousPermMask: A **WSS Rights Mask** (section 2.2.2.14) that indicates the permissions granted to a user that is anonymous, or has no specific permissions, to this document. This value can be NULL if anonymous access to the document is not allowed.

DraftOwnerId: The user identifier (section 2.2.1.13) of the user that published this document as a draft. This value is only non-NULL if the requested document is a draft version.

Level: A **Publishing Level** type value specifying the publishing status of this document. This value MUST match *@Level*.

ParentVersion: If the document is a transformed version of another document (see Doc Flags value 0x00000400), then this is the **UIVersion** value from the parent document. This value MUST be NULL if the document is not a child of a document transformation.

TransformerId: If the document is a transformed version of another document, then this is the GUID of the agent that performed the transformation. This value MUST be NULL if the document is not the product of a document transformation.

ParentLeafName: If the document is a transformed version of another document, then this is the leaf name of the original document, if the original document is in the same folder as the transformed document. If either is moved, the relationship is broken. This value MUST be NULL if the document is not the product of a document transformation.

ProgId: Designates a preferred application to open the document. The **ProgId** is used to distinguish between different applications that save files with a given file extension (that is, different editing applications for HTML or XML files). This value MUST be NULL if the parser did not specify a **ProgId** when the document was saved.

DoclibRowId: The document library row identifier for this document. If the requested document is not contained in a list, this value MUST be NULL.

{tp_DefaultWorkflowId}: A placeholder for the workflow identifier corresponding to a **workflow** invoked as part of a check in on a moderated list. This value MUST be NULL.

ListId: The list identifier (section 2.2.1.5) of the list containing the requested document. If the document is not contained in a list, this value MUST be NULL.

ItemChildCount: The number of non-folder (those whose **Document Store** type is not folder) children of this document.

FolderChildCount: The number of folder (those whose **Document Store** type is folder) children of this document.

MetaInfoVersion: A counter incremented any time a change is made to the metainfo for this document and used for internal conflict detection.

{MetaInfo}: A METADICT for this document. The METADICT format is specified in [MS-FPSE] section 2.2.2.2.11. This value MUST be NULL if the document does not have any METADICT associated with it.

ContentVersion: A counter incremented any time a change is made to the binary contents of this document; used for internal conflict detection. This value MUST be NULL if the document does not exist.

{ContentVersionIsDirty}: This value specifies whether the **ContentVersion** value can be used for conflict detection. If the **ContentVersion** value can be used for conflict detection or the **ContentVersion** value is NULL, this value MUST be 0; otherwise it MUST be 1.

2.2.5.8 Empty Result Set

The Empty Result Set is defined using T-SQL syntax, as follows.

```
    {No column name}     Not Applicable
```

2.2.5.9 Event Receivers Result Set

The Event Receivers Result Set contains information about the **event receivers** defined for an **Event Host**.

The Event Receivers Result Set MUST contain one row for each event receiver registered for the **Event Host**.

The Event Receivers Result Set is defined using T-SQL syntax, as follows.

```
    Id                              uniqueidentifier,
    Name                            nvarchar(256),
    SiteId                          uniqueidentifier,
    WebId                           uniqueidentifier,
    HostId                          uniqueidentifier,
    HostType                        int,
    ItemId                          int,
    DirName                         nvarchar(256),
    LeafName                        nvarchar(128),
    Synchronization                 int,
    Type                            int,
    SequenceNumber                  int,
    Assembly                        nvarchar(256),
    Class                           nvarchar(256),
    SolutionId                      varbinary(512),
    Data                            nvarchar(256),
    Filter                          nvarchar(256),
    SourceId                        varbinary(512),
    SourceType                      int,
    Credential                      int,
    ContextType                     varbinary(16),
    ContextEventType                varbinary(16),
    ContextId                       varbinary(16),
    ContextObjectId                 varbinary(16),
    ContextCollectionId             varbinary(16),
    Hash                         nvarchar(50),
    ValidatorsHash                  char(64),
    ValidationErrorUrl           nvarchar(4000),
    ValidationErrorMessage       nvarchar(4000)
```

Id: The event receiver identifier of the event receiver.

Name: The **display name** of the event receiver.

SiteId: The site collection identifier (section 2.2.1.9) for a site collection that contains the event host on which the event receiver is registered.

WebId: The site identifier (section 2.2.1.11) for the site that contains the event host on which the event receiver is registered.

HostId: The **document identifier** (section 2.2.1.2) for the event host on which the event receiver is registered.

HostType: The **Event Host** type (section 2.2.3.5) of the event host on which the event receiver is registered.

ItemId: Reserved. The server MUST default this value to NULL, and a client MUST NOT use this value.

DirName: Reserved. The server MUST default this value to NULL, and a client MUST NOT use this value.

LeafName: Reserved. The server MUST default this value to NULL, and a client MUST NOT use this value.

Synchronization: Contains the execution behavior of the event receiver. MUST be one of the following values.

Value	Description
0	If the **Event Receiver** type (section 2.2.3.6) specified by the **Type** column is less than 10001, the execution behavior of the event receiver MUST be synchronous.
1	The execution behavior of the event receiver MUST be synchronous.
2	The execution behavior of the event receiver MUST be asynchronous.

Type: The **Event Receiver** type of the event receiver.

SequenceNumber: An ordinal value that determines the relative order in which the event receiver is triggered. **SequenceNumber** MUST be greater than or equal to 0 and less than or equal to 65535.

Assembly: The name of the .NET assembly that contains the implementation of the event receiver.

Class: The name of the .NET class definition that contains the implementation of the event receiver.

SolutionId: The sandboxed solution identifier of the sandboxed solution.

Data: Additional data persisted on behalf of the event receiver implementation, to be passed to the event receiver.

Filter: Reserved. The server MUST default this value to NULL, and a client MUST NOT use this value.

SourceId: The **feature identifier** (section 2.2.1.4) or **content type identifier** (section 2.2.1.1) of the feature or content type that registered the event receiver. If the event receiver was not registered by a feature or content type, the value MUST be NULL.

SourceType: An integer value that specifies the source of the registration for the event receiver. The **SourceType** value MUST be one of the following values.

Value	Description
0	The event receiver does not come from a specially treated source.
1	A content type registered the event receiver.
2	A feature registered the event receiver.

Credential: Reserved. The server MUST default this value to NULL, and a client MUST NOT use this value.

ContextType: The **workflow identifier** (section 2.2.1.16) of the **workflow** that the **event (2)** relates to, if any.

ContextEventType: Reserved. The server MUST default this value to NULL, and a client MUST NOT use this value.

ContextId: The workflow identifier for the registered workflow, if any.

ContextObjectId: The document identifier for the object instance that the Workflow is registered against, if any.

ContextCollectionId: A workflow identifier for the **workflow instance** that manages the event receiver, if any.

Hash: The implementation-specific **hash** of the content of the **sandboxed solution**.

ValidatorsHash: The implementation-specific hash of the sandboxed solution validator that validated the sandboxed solution.

ValidationErrorUrl: The URL to render the sandboxed solution validator validation failed. If validation succeeded, it MUST be NULL

ValidationErrorMessage: If the sandboxed solution validator validation failed, this stores an error message string. Otherwise, it MUST be NULL.

2.2.5.10 Individual URL Security Result Set

The Individual URL Security Result Set contains security information about a document.

The Individual URL Security Result Set is defined using T-SQL syntax, as follows.

```
{ListId}                        uniqueidentifier,
Acl                             varbinary(max),
AnonymousPermMask               bigint,
{IsAttachment}                  bit,
{NeedManageListRight}           bit,
{BaseType}                      int,
{ExcludedType}                  int,
{ListFlags}                     bigint,
{Level}                         tinyint,
{DraftOwnerId}                  int,
{DocType}                       tinyint,
{bNeedNoThrottle}               bit,
{ItemCount}                     int,
{DoclibRowId}                   int,
{DocFlags}                      int;
```

{ListID}: The **list identifier** (section 2.2.1.5) of the **list** or document library containing the document location.

Acl: The WSS ACL for the security scope associated with the document location.

AnonymousPermMask: The **WSS Rights Mask** (section 2.2.2.14) that applies to an anonymous user, or a user with no assigned permissions, in the security scope associated with the document location.

{IsAttachment}: A bit flag specifying whether the document location is an attachment or an attachment folder. This value MUST be set to 1 if the document location is an attachment or attachment folder; otherwise, it MUST be 0.

{NeedManageListRight}: A bit flag specifying whether the current user needs the **ManageLists** bit of the **WSS Rights Mask** set to write to the document location. This value MUST be set to 1 if the current user MUST have the **ManageLists** bit of the **WSS Rights Mask** set in the security scope of the document location in order to write the document; otherwise, it MUST be 0.

{BaseType}: The **List Base** type (section 2.2.3.11) of the list or document library containing the document location.

{ExcludedType}: Contains an enumerated value specifying whether the document location is within a special folder type in the containing list or document library. The following values are valid.

Value	Description
0	The document location is not contained in a special folder.
1	The document location is, or is contained within, a forms folder: a folder named "Forms" within a document library.
2	The document location is, or is contained within, a web image folder: a folder named "_w" within a document library.
3	The document location is, or is contained within, a thumbnail folder: a folder named "_t" within a document library.
4	The document location is at the **root folder** of the list or document library.

{ListFlags}: The **List Flags** (section 2.2.2.5) of the list or document library containing the document location.

{Level}: The **Publishing Level** type (section 2.2.2.6) value specified for the document, or if the parameter was NULL, the **Publishing Level** type of the **current version** of the document for the current user.

{DraftOwnerId}: The user identifier (section 2.2.1.13) of the **principal** that published this document as a draft. This value MUST be NULL if the document does not exist or does not have a draft version.

{DocType}: The **Document Store** type (section 2.2.2.4) of this document.

{bNeedNoThrottle}: If the list or document library containing the document location is a list or document library that can ignore throttle, then this value MUST be 1. Otherwise, it MUST be 0.

{ItemCount}: The number of the Items in list or document library containing the document location is a list or document library.

{DoclibRowId}: The row identifier of the document in the containing list or document library.

{DocFlags}: A **Doc Flags** (section 2.2.2.3) value specifying information about the document.

2.2.5.11 Link Information Result Set

The Link Information Result Set returns information about each **forward link** from the document and backward link to the document within the site collection.

The Link Information Result Set is defined using T-SQL syntax, as follows.

```
    LinkDirName                    nvarchar(256),
    LinkLeafName                   nvarchar(128),
    LinkType                       tinyint,
    LinkSecurity                   tinyint,
    LinkDynamic                    tinyint,
    LinkServerRel                  bit,
    LinkStatus                     tinyint,
    PointsToDir                    bit,
    WebPartId                      uniqueidentifier,
    LinkNumber                     int,
    WebId                          uniqueidentifier,
    Search                         nvarchar(max),
    FieldId                        uniqueidentifier;
```

LinkDirName: Contains the directory name of the link target.

LinkLeafName: Contains the leaf name of the link target.

LinkType: The **LinkType** value of the link. This value MUST be NULL for a backward link.

LinkSecurity: The **LinkSecurity** value of the forward link, which specifies whether the scheme of the link is HTTP or HTTPS. This value MUST be NULL for a backward link.

LinkDynamic: The **LinkDynamic** value of the forward link, which specifies whether the link is to a special type of target. This value MUST be NULL for a backward link.

LinkServerRel: A bit flag that specifies whether the link URL is **server-relative URL** or not. A value of 1 specifies a server-relative URL. This value MUST be NULL for a backward link.

LinkStatus: The **Document Store** type (section 2.2.2.4) value of the document targeted by a forward link. This value MUST be 128 for a backward link. If the forward link target is a document that does not exist, or if this link refers to a target that exists outside the specified site collection, or if it refers to a location that could not be verified, this value MUST be NULL.

PointsToDir: A bit flag specifying whether the target of the forward link was a directory and has been modified to target a welcome page. This value MUST be NULL for a backward link. For a forward link, if the link target is a directory where a welcome page is specified, the link MUST be changed to the URL of the welcome page and **PointsToDir** MUST be set to 1 so that the link can be distinguished from an explicit link to the welcome page; otherwise this value MUST be 0.

WebPartId: This value MUST be NULL.

LinkNumber: This value MUST be NULL.

WebId: This value MUST be NULL.

Search: This value MUST be NULL.

FieldId: This value MUST be NULL.

2.2.5.12 List Metadata Result Set

The List Metadata Result Set contains the metadata associated with a **list**.

The List Metadata Result Set is defined using T-SQL syntax, as follows.

```
tp_Id                                 uniqueidentifier,
tp_Title                              nvarchar(255),
tp_Modified                           datetime,
tp_Created                            datetime,
tp_LastDeleted                        datetime,
tp_Version                            int,
tp_BaseType                           int,
tp_FeatureId                          uniqueidentifier,
tp_ServerTemplate                     int,
DirName                               nvarchar(256),
LeafName                              nvarchar(128),
DirName                               nvarchar(256),
LeafName                              nvarchar(128),
tp_ReadSecurity                       int,
tp_WriteSecurity                      int,
tp_Description                        nvarchar(max),
{tp_Fields}                           varbinary(max),
tp_Direction                          int,
AnonymousPermMask                     bigint,
tp_Flags                              bigint,
tp_ThumbnailSize                      int,
tp_WebImageWidth                     int,
tp_WebImageHeight                    int,
tp_ImageUrl                          nvarchar(255),
tp_ItemCount                         int,
tp_Author                            int,
tp_HasInternalFGP                    bit,
tp_ScopeId                           uniqueidentifier,
Acl                                  varbinary(max),
tp_EventSinkAssembly                 nvarchar(255),
tp_EventSinkClass                    nvarchar(255),
tp_EventSinkData                     nvarchar(255),
tp_EmailAlias                        nvarchar(128),
tp_WebFullUrl                        nvarchar(256),
tp_WebId                             uniqueidentifier,
tp_WebTitle                          nvarchar(255),
tp_Web                               int,
tp_WebLanguage                       int,
tp_WebCollation                      smallint,
tp_SendToLocation                    nvarchar(512),
{tp_MaxMajorVersionCount}            int,
{tp_MaxMajorwithMinorVersionCount}   int,
tp_MaxRowOrdinal                     int,
tp_ListDataDirty                     int,
tp_DefaultWorkflowId                 uniqueidentifier,
{HasBackLookupRels}                  bit,
tp_ContentTypes                      varbinary(max),
tp_Subscribed                        bit,
{ChildCount}                         int,
tp_NoThrottleListOperations          bit,
tp_TitleResource                     nvarchar(256),
tp_DescriptionResource               nvarchar(256),
tp_ValidationFormula                 nvarchar(1024),
tp_ValidationMessage                 nvarchar(1024);
```

tp_Id: The list identifier (section 2.2.1.5) of the list.

tp_Title: The title of this list for display in the front-end web server.

tp_Modified: A time stamp in **UTC** format specifying when this list was last modified.

tp_Created: A time stamp in UTC format specifying when this list was created.

tp_LastDeleted: A time stamp in UTC format specifying when an item was last deleted from this list. This value MUST default to NULL if no item has been deleted.

tp_Version: A counter incremented any time a change is made to the schema or other properties of this list and used for internal conflict detection.

tp_BaseType: The **List Base** type (section 2.2.3.11) of this list. The following values are valid.

Value	Description
0	"Generic" list
1	Document library
3	"**Discussion board**" list
4	"Survey" list
5	"Issues" list

tp_FeatureId: The feature identifier (section 2.2.1.4) for the feature that defines the base schema of this list.

tp_ServerTemplate: The **list template identifier** that defines the base structure of this list.

DirName: The directory name of the location that contains this list.

LeafName: The leaf name of the location that contains this list.

DirName: The directory name of the default **document template** in the list. This value can be NULL if a document template is not defined for this list.

LeafName: The leaf name of the default document template in the list. This value can be NULL if a document template is not defined for this list.

tp_ReadSecurity: This signifies special restrictions that can be placed on list item access. The following values are valid.

Value	Description
1	No special restrictions.
2	Users can see only their own list items. The front-end web server MUST NOT display list items to users without the **ManageLists** permissions unless the list item was created by that user (for example, **tp_Author** = *@UserId*).

tp_WriteSecurity: This signifies special restrictions that can be placed on list item update. The following values are valid.

Value	Description
1	No special restrictions.
2	Users will see only their own list items. The front-end web server MUST NOT allow users without the **ManageLists** permission to update a list item unless the list item was created by that user (for example, **tp_Author** = *@UserId*).
4	Users will not update any list items in this list. The front-end web server MUST NOT allow users without the **ManageLists** permission to add or update list items in this list.

tp_Description: The description of this list for display in the front-end web server.

{tp_Fields}: This field MUST be NULL if the site or list has been flagged to cache all schema data; otherwise, it contains a **WSS Compressed** structure (section 2.2.4.8). Uncompressed it contains an implementation-specific version string followed by an XML representation of the **field definitions**. The field definitions include display and interaction options. The XML MUST conform to the **FieldDefinitionDatabaseWithVersion** complex type, as specified in section 2.2.8.3.5.

tp_Direction: An enumerated value specifying the direction of text flow for front-end web server elements presented by this list. The following values are valid.

Value	Description
0	No explicit direction is specified.
1	Text flow SHOULD be left to right.
2	Text flow SHOULD be right to left.

AnonymousPermMask: A **WSS Rights Mask** (section 2.2.2.14) that indicates the permissions granted to a user that is anonymous, or has no specific permissions, on this list.

tp_Flags: A **List Flags** (section 2.2.2.5) value describing this list.

tp_ThumbnailSize: The width, in pixels, specified for use when creating thumbnail images of list items within this list. This value MUST be NULL for lists that do not have a **List Server Template** (section 2.2.3.12) value of 109 (Image Library Template). Thumbnail images are generated by the front-end web server for documents and have implementation-specific capabilities.

tp_WebImageWidth: The width, in pixels, specified for use when creating web images of list items within this list. This value MUST be NULL for lists that do not have a **List Server Template** value of 109 (Image Library Template). web images are generated by the front-end web server for documents and have implementation-specific capabilities.

tp_WebImageHeight: The height, in pixels, specified for use when creating web images of list items within this list. This value MUST be NULL for lists that do not have a **List Server Template** value of 109 (Image Library Template).

tp_ImageUrl: The URL of the image used to represent this list.

tp_ItemCount: The number of list items that are stored within this list.

tp_Author: The identifier of the user who is listed as creating this list.

tp_HasInternalFGP: This flag is set to 1 if there have ever been list items for this list that have had a unique access control list (ACL) applied. Otherwise it is set to 0.

tp_ScopeId: Unique identifier of the security scope for this list. This indicates the specific **WSS ACL Format** (section 2.2.4.6) access control list (ACL) to use for calculating the permission settings on this list.

Acl: The binary serialization of the **WSS ACL Format** ACL for this list. This MUST be used for the permissions for this list if the **tp_ScopeId** value of this list is set to unique security scope.

tp_EventSinkAssembly: The name of the .NET assembly that implements the **event sink** associated with this list. This value MUST default to NULL if no event sink has been associated with the list.

tp_EventSinkClass: The name of the .Net assembly class that implements the event sink associated with this list. This value MUST default to NULL if no event sink has been associated with the list.

tp_EventSinkData: Additional data persisted on behalf of the event sink implementation, to be passed to the event sink associated with this list. This value MUST default to NULL if no event sink has been associated with the list.

tp_EmailAlias: The **email address** of the list. This email address is used to allow files to be sent directly to this list through an implementation-specific email handling feature. This value MUST default to NULL if no email inserts server has been associated with the list.

tp_WebFullUrl: The complete store-relative form URL to the site that contains this list.

tp_WebId: The site identifier (section 2.2.1.11) of the site that contains the list.

tp_WebTitle: The title, for display in the front-end web server, of the site that contains this list.

tp_Web: The identifier of the **site template** for the site that contains this list.

tp_WebLanguage: The **language code identifier (LCID)** of the display language of the site that contains this list.

tp_WebCollation: The **collation order** for information in the site that contains this list.

tp_SendToLocation: Contains an implementation-specific string of the URL used to copy list items to alternative locations. This value MUST be NULL if no "Send To Location" has been associated with the list. The "Send To Location" is an implementation-specific **feature** that allows users to manually save copies of **list items** and **documents** to a remote location.

{tp_MaxMajorVersionCount}: If the list has versioning enabled, this field contains the number of **major versions** that will be retained for this document. All versions more than **tp_MaxMajorVersionCount** removed from the current version of the document are automatically removed at version creation time. A value of 0 specifies that versions SHOULD NOT automatically be removed for this list.

{tp_MaxMajorwithMinorVersionCount}: If the list has versioning enabled, this field contains the number of major versions that will have their associated minor versions retained for this document. All versions more than **tp_MaxMajorVersionCount** removed from the current version of the document are automatically removed at version creation time. A value of 0 specifies that versions SHOULD NOT automatically be removed for this list.

tp_MaxRowOrdinal: This specifies the maximum row ordinal used to store list items for this list. This value indicates an implementation-specific calculation for storage of list items within lists.

tp_ListDataDirty: This is set to 1 if the list items in this list require dependency update processing before their next access (for example, updating document link information by parsing each document).

tp_DefaultWorkflowId: The workflow identifier (section 2.2.1.16) corresponding to the **workflow** invoked if the document is in a moderated list and the document is submitted for approval as part of a check-in. If the document does not exist, or is not contained in a list with a configured approval Workflow, this value MUST be NULL.

{HasBackLookupRels}: A bit flag specifying whether this list has any relationship lookup fields with relationship delete behavior that cascades or relationship delete behavior that is restricted.

tp_ContentTypes: WSS Compressed structure (section 2.2.4.8). Uncompressed it will get XML data specifying the content types registered for this list.

tp_Subscribed: Set to 1 if an **alert** for changes to this list has been created in the past, signifying that additional processing needs to be performed.

{ChildCount}: If the document is contained within a list, this MUST be the sum total of list items and folders within the list. Otherwise, this MUST be the sum total of documents and folders that are contained within directory specified by the **DirName** column.

tp**_NoThrottleListOperations:** If this list is exempt from resource throttling operations, then this value MUST be 1. Otherwise, it MUST be 0.

tp_DescriptionResource: The resource token or resource identifier of the description for the site.

tp_ValidationFormula: A string representing the data validation criteria used to perform custom validation rules prior to the list being updated.

tp_ValidationMessage: A string containing text to display in the user interface when the list fails validation based on the data validation criteria represented by **tp_ValidationFormula**.

2.2.5.13 List Web Parts Result Set

The List Web Parts Result Set contains information about the Web parts related to the **lists** associated with a specified document.

The List Web Parts Result Set is defined using T-SQL syntax, as follows.

```
    tp_ListID                      uniqueidentifier,
    tp_Type                        tinyint,
    tp_Id                          uniqueidentifier,
    tp_Flags                       int,
    tp_DisplayName                 nvarchar(255),
    tp_PageUrl                      nvarchar(255),
    tp_BaseViewId                   tinyint,
    tp_View                         varbinary(max),
    tp_Level                        tinyint,
    tp_ContentTypeId                varbinary(512),
    tp_PageUrlID                    uniqueidentifier,
    tp_AllUsersProperties           varbinary(max),
    tp_PerUserProperties            varbinary(max),
    tp_WebPartIdProperty            nvarchar(255),
    tp_Cache                        varbinary(max)
```

tp_ListID: The list identifier (section 2.2.1.5) of the list containing the Web part.

tp_Type: The **Page** type (section 2.2.3.14) of the Web part.

tp_Id: The web part identifier (section 2.2.1.15) of the Web part.

tp_Flags: A **View Flags** (section 2.2.2.12) value specifying view-related settings for this Web part.

tp_DisplayName: The display name specified for the Web part, if any. This value can be an empty string.

tp_PageUrl: The store-relative form URL to the site that contains this Web part.

tp_BaseViewId: The view identifier for the view where this Web part is used.

tp_View: Contains implementation-specific XML used when processing this Web part. This data is stored as a **WSS Compressed** structure (section 2.2.4.8), and is compressed by the algorithm specified in [RFC1950].

tp_Level: A **Publishing Level** type (section 2.2.2.6) value specifying the publishing status of the Web part. This value MUST be NULL if the Web part does not exist.

tp_ContentTypeId: A binary content type identifier value specifying the valid content types that this Web part can be used to view. If this Web part is not restricted to a particular content type, this value MUST be 0.

tp_PageUrlID: The **document identifier** (section 2.2.1.2) with which the web part is associated.

tp_AllUsersProperties: A list of the **XML** properties which are common for all users of the web part. This value can be NULL.

tp_PerUserProperties: A list of the XML properties which are specific to a particular user of the web part. This value can be NULL.

tp_WebPartIdProperty: A string which identifies the web part within the page.

tp_Cache: The private data that is cached for the web part. This value can be NULL.

2.2.5.14 NULL Individual URL Security Result Set

The NULL Individual URL Security Result Set indicates that a specified document is not found.

The NULL Individual URL Security Result Set MUST return a single row and is defined using T-SQL syntax, as follows.

```
    {ListId}                    uniqueidentifier = NULL,
    {Acl}                       varbinary(max) = NULL,
    {AnonymousPermMask}         bigint = NULL,
    {IsAttachment}              bit = 0,
    {NeedManageListRight}       bit = 0,
    {BaseType}                int = NULL,
    {ExcludedType}            int = NULL,
    {ListFlags}               bigint = NULL,
    {Level}                   tinyint = NULL,
    {DraftOwnerId}            int = NULL,
    {DocType}                 tinyint = NULL,
    {bNeedNoThrottle}         bit = NULL,
    {ItemCount}               int = NULL,
    {DoclibRowId}             int = 0,
    {DocFlags}                int = 0;
```

{ListID}: This MUST be NULL.

{Acl}: This MUST be NULL.

{AnonymousPermMask}: This MUST be NULL.

{IsAttachment}: This MUST be 0.

{NeedManagedListRight}: This MUST be 0.

{BaseType}: This MUST be NULL.

{ExcludedType}: This MUST be NULL.

{ListFlags}: This MUST be NULL.

{Level}: This MUST be NULL.

{DraftOwnerId}: This MUST be NULL.

{DocType}: This MUST be NULL.

{bNeedNoThrottle}: This MUST be NULL.

{ItemCount}: This MUST be NULL.

{DoclibRowId}: This MUST be 0.

{DocFlags}: This MUST be 0.

2.2.5.15 NULL Unique Permissions Result Set

The NULL Unique Permissions Result Set indicates that unique security scopes are not found in the specified location. When returned, the result set MUST contain a single row of data. The NULL Unique Permissions Result Set is defined using T-SQL syntax, as follows.

```
    {ScopeId}                     binary(1),
    {Acl}                         varbinary(max),
    {AnonymousPermMask}           binary(1);
```

{ScopeId}: This MUST be 0x00. This field can be cast as a **uniqueidentifier** with a value of 00000000-0000-0000-0000-000000000000.

{Acl}: This MUST be NULL.

{AnonymousPermMask}: This MUST be 0x00. This field can be cast as a **bigint**.

2.2.5.16 Object ID Result Set

The **Object ID Result Set** contains the GUID of the matching object, if any. The **Object ID Result Set** is defined using T-SQL syntax, as follows.

```
    Id                         uniqueidentifier
```

Id: Contains the GUID of the matching object.

2.2.5.17 Principal User Information Result Set

The Principal user Information Result Set returns information about a **principal**. The Principal user Information Result Set is defined using T-SQL syntax, as follows.

```
    tp_Id                          int,
    tp_SystemID                    varbinary(512),
    tp_Title                       nvarchar(255),
    tp_Login                       nvarchar(255),
    tp_Email                       nvarchar(255),
    tp_Notes                       nvarchar(1023),
    tp_SiteAdmin                   bit,
    tp_DomainGroup                 bit,
    tp_Flags                       int;
```

tp_Id: The user identifier (section 2.2.1.13) of the principal. MUST NOT be NULL.

tp_SystemID: The **SystemID** (section 2.2.1.12) of the principal. This parameter MUST NOT be NULL.

tp_Title: Contains the display name of the principal. This parameter MUST NOT be NULL.

tp_Login: Contains the login name of the principal. This parameter MUST NOT be NULL.

tp_Email: Contains the email address of the principal. This parameter MUST NOT be NULL.

tp_Notes: Contains notes associated with the principal. This parameter MUST NOT be NULL.

tp_SiteAdmin: Set to 1 if the principal is a **site collection administrator**; otherwise 0. This parameter MUST NOT be NULL.

tp_DomainGroup: Set to 1 if the principal is a domain group; otherwise 0. This parameter MUST NOT be NULL.

tp_Flags: The **UserInfo Flags** (section 2.2.2.11) value of the principal. This parameter MUST NOT be NULL.

2.2.5.18 Server Time Result Set

The Server Time Result Set returns the current time from the back-end database server in **UTC**. The Server Time Result Set is defined using T-SQL syntax, as follows.

```
    {CurrentTime}                  datetime;
```

{CurrentTime}: The current time from the back-end database server, in UTC format.

2.2.5.19 Single Doc Link Information Result Set

The Link Info Result Set returns information about each forward link from, and each backward link to, a specified document or document version.

The Link Info Result Set MUST be ordered by the DocId column and is defined using T-SQL syntax, as follows.

```
DocId                          uniqueidentifier,
LinkDirName                    nvarchar(256),
LinkLeafName                   nvarchar(128),
LinkType                       tinyint,
LinkSecurity                   tinyint,
LinkDynamic                    tinyint,
LinkServerRel                  bit,
LinkStatus                     tinyint,
PointsToDir                    bit,
WebPartId                      uniqueidentifier,
LinkNumber                     int,
WebId                          uniqueidentifier,
Search                         nvarchar(max),
FieldId                        uniqueidentifier;
```

DocId: The **document identifier** (section 2.2.1.2) of the specified document.

LinkDirName: Contains the directory name of the link target.

LinkLeafName: Contains the leaf name of the link target.

LinkType: The **LinkType** value of the link. See **LinkType Types** (section 2.2.3.10) for a list of valid **LinkType** values. This value MUST be NULL for a backward link.

LinkSecurity: The **LinkSecurity** value of the forward link, which specifies whether the scheme of the link is HTTP or HTTPS. This value MUST be NULL for a backward link.

LinkDynamic: The **LinkDynamic** value of the forward link, which specifies whether the link is to a special type of target. This value MUST be NULL for a backward link.

LinkServerRel: A bit flag which specifies whether the link URL is server-relative or not. A value of 1 specifies a server-relative URL. This value MUST be NULL for a backward link.

LinkStatus: The **Document Store** type (section 2.2.2.4) value of the document targeted by a forward link. This value MUST be 128 for a backward link. If the forward link target is a document that does not exist, or if this link refers to a target that exists outside the specified site collection, or if it refers to a location that could not be verified, this value MUST be NULL.

PointsToDir: A bit flag specifying whether the target of the forward link was a directory and has been modified to target a welcome page. This value MUST be NULL for a backward link. For a forward link, if the link target is a directory where a welcome page is specified, the link MUST be changed to the URL of the welcome page and **PointsToDir** MUST be set to 1 so that the link can be distinguished from an explicit link to the welcome page; otherwise, this value MUST be 0.

WebPartId: This value MUST be NULL.

LinkNumber: This value MUST be NULL.

WebId: This value MUST be NULL.

Search: This value MUST be NULL.

FieldId: This value MUST be NULL.

2.2.5.20 Site Audit Mask Result Set

The Site Audit Mask Result Set contains the context-sensitive identifier for a specified object, and the **Audit Flags** (section 2.2.2.1) set and inherited on that object. The Site Audit Mask Result Set is defined using T-SQL syntax, as follows.

```
{Id}                          uniqueidentifier,
{AuditFlags}                  int,
{InheritAuditFlags}           int,
{SiteGlobalAuditMask}         int;
```

{Id}: The identifier for the object. The **Audit Item** type (section 2.2.3.2) of the object determines the type of identifier used for this value (for example, a site identifier (section 2.2.1.11) or a **list identifier** (section 2.2.1.5)). If an object of the specified **Audit Item** type at the specified location is found, the value MUST NOT be NULL; otherwise, it MUST be NULL.

{AuditFlags}: An **Audit Flags** value specifying the operations to be audited that are set directly on the specified object. If an object of the specified **Audit Item** type at the specified location is not found, the value MUST be NULL.

{InheritAuditFlags}: An **Audit Flags** value specifying the operations to be audited on the specified object that are inherited from a contained document. If an object of the specified **Audit Item** type at the specified location is not found, the value MUST be NULL.

{SiteGlobalAuditMask}: An **Audit Flags** value specifying the operations to be audited on the specified object that are applied globally within the specified site collection.

2.2.5.21 Site Feature List Result Set

The Site Feature List Result Set contains the **list** of feature identifiers (section 2.2.1.4) of a site.

The Site Feature List Result Set is defined using T-SQL syntax, as follows.

```
FeatureId uniqueidentifier;
```

FeatureId: A feature identifier for a feature of a site.

2.2.5.22 Site Metadata Result Set

The Site Metadata Result Set contains metadata for a site or sites.

The Site Metadata Result Set is defined using T-SQL syntax, as follows.

```
Id                            uniqueidentifier,
Title                         nvarchar(255),
Description                   nvarchar(max),
MetaInfoVersion               int,
WebTemplate                   int,
Language                      int,
Locale                        int,
Collation                     smallint,
TimeZone                      smallint,
Time24                        bit,
CalendarType                  smallint,
AdjustHijriDays               smallint,
AltCalendarType               tinyint,
CalendarViewOptions           tinyint,
WorkDays                      smallint,
WorkDayStartHour               smallint,
WorkDayEndHour                 smallint,
ProvisionConfig                smallint,
Flags                          int,
Author                         int,
AlternateCSSUrl                nvarchar(260),
CustomizedCss                  nvarchar(260),
CustomJSUrl                    nvarchar(260),
AlternateHeaderUrl             nvarchar(260),
SecurityProvider               uniqueidentifier,
MasterUrl                      nvarchar(260),
CustomMasterUrl                nvarchar(260),
{SiteLogoUrl}                  nvarchar(260),
{SiteLogoDescription}          nvarchar(255),
AllowMUI                       bit,
TitleResources                 nvarchar(256),
DescriptionResource            nvarchar(256),
AlternateMUICultures           nvarchar(max),
OverwriteMUICultures           bit,
UIVersion                      tinyint,
ClientTag                      smallint,
AnonymousPermMask              bigint,
{SiteFlags}                    int,
{SitePortalURL}                nvarchar(260),
{SitePortalName}               nvarchar(255),
MeetingCount                   smallint,
DefTheme                       nvarchar(64),
CachedNav                      varbinary(max),
CachedInheritedNav             varbinary(max),
CachedNavDirty                 int,
CachedDataVersion              int,
NavParentWebId                 uniqueidentifier,
FirstUniqueAncestorWebId       uniqueidentifier,
ScopeId                        uniqueidentifier,
DbNow                          datetime,
Acl                            varbinary(max),
{RequestAccessEmail}           nvarchar(255),
Ancestry                       varbinary(max),
ProductVersion                 smallint,
tp_Id                          int,
tp_SiteAdmin                   bit,
tp_IsActive                    bit,
tp_Login                       nvarchar(255),
tp_Email                       nvarchar(255),
tp_Title                       nvarchar(255),
tp_Notes                       nvarchar(1023),
tp_ExternalTokenLastUpdated    datetime,
tp_Token                       varbinary(max),

tp_Flags                       int,
UserId                         int,
{SiteSecurityVersion}          bigint,
tp_Locale                      int,
tp_TimeZone                    smallint,
tp_Time24                      bit,
tp_CalendarType                smallint,
tp_AdjustHijriDays             smallint,
tp_AltCalendarType             tinyint,
tp_CalendarViewOptions         tinyint,
tp_WorkDays                    smallint,
tp_WorkDayStartHour            smallint,
tp_WorkDayEndHour              smallint,
{SiteHashKey}                  binary(16),
{UserInfoListId}               uniqueidentifier,
{RootWebId}                    uniqueidentifier,
{SiteLastContentChange}        datetime,
{SiteLastSecurityChange}       datetime,
{RbsCollectionId}              int;
```

Id: The site identifier (section 2.2.1.11) for the site.

Title: The title of the site for display in the front-end web server.

Description: The description of the site for display in the front-end web server.

MetaInfoVersion: A counter incremented any time a change is made to the metainfo for this site and used for internal conflict detection.

WebTemplate: The identifier for the site template used in the **site definition** to define the base structure of this site.

Language: The LCID associated with the site that is used to determine current UI culture, which determines which language resources to use to display messages on the front-end web server.

Locale: The LCID associated with the site that is used to determine the current culture for regional language specific data formatting such as currency or date and time settings.

Collation: The **Collation Order** of the site which indicates an additional sorting order that SHOULD be processed by the back-end database server. The collation method is an implementation-specific capability of the front-end web server and back-end database server.

TimeZone: The **Time Zone Identifier** (section 2.2.3.17) for the time zone to be used when displaying time values for this site.

Time24: If set to "1", use a 24-hour time format when displaying time values for this site; otherwise, a 12-hour time format can be used.

CalendarType: The **Calendar** type (section 2.2.3.3) that is to be used when processing date values for this site.

AdjustHijriDays: If the **Calendar** type value is "6", this specifies the number of days to extend or reduce the current month in Hijri calendars for this site.

AltCalendarType: The **Calendar** type of an alternate calendar for processing date values for this site. If NULL, only the **CalendarType** value is used for this site.

CalendarViewOptions: A **Calendar View Options** type that specifies the calendar display options setting for this site.

WorkDays: A set of **Workdays Flags** that specify the weekdays defined as the workweek for this site.

WorkDayStartHour: The start time of the workday in minutes after midnight for this site.

WorkDayEndHour: The end time of the workday in minutes after midnight for this site.

ProvisionConfig: An identifier of the site template used to provision this site. The following reserved site template identifiers are defined.

Value	Description
-1	This site has not had any site template provisioned.
0	This site has the implementation-specific default site template applied.
1	This site has the Team Collaboration site template applied.
2	This site has the **Meeting Workspace site** template applied.
3	This site has the Central Administration site template applied.

Value	Description
4	This site has the Wiki site template applied.
9	This site has the Blog site template applied.

Flags: A **Site Property Flags** value describing the configuration of this site.

Author: The user identifier (section 2.2.1.13) of the user who is listed as creating this site.

AlternateCSSUrl: The URL for a custom CSS sheet file registered on the site for use in pages of the site.

CustomizedCss: This contains an implementation-specific list of custom CSS files associated with this site.

CustomJSUrl: The URL for a custom JavaScript file registered on the site for use in pages of the site.

AlternateHeaderUrl: The URL for a custom header HTML page registered on the site for use in pages of the site when rendered on the front-end web server.

SecurityProvider: **COM** class identifier (CLSID) of the **external security provider** for this site. This is NULL for sites using the native security implementation.

MasterUrl: The URL for the **master page** registered on the site for use in pages of the site when rendered on the front-end web server.

CustomMasterUrl: The URL for an alternate master page registered on the site for use in pages of the site rendered on the front-end web server.

{SiteLogoUrl}: The URL of an image that represents the site for display in the front-end web server.

{SiteLogoDescription}: The description of the image that represents the site for display in the front-end web server as an ALT tag on the image.

AllowMUI: A bit which indicates whether the Multilingual user Interface feature is enabled.

TitleResource*:* The resource token or resource identifier of the title for the site whose metadata is to be updated.

DescriptionResource: The resource token or resource identifier of the description for the site whose metadata is to be updated.

AlternateMUICultures: The string that contains the distinct language identifier(s) for all the alternate language(s) of the site. For example, the language identifier for English is 1033. Every element is separated by semicolon from the next.

OverwriteMUICultures: A bit which specifies whether the changes made to user-specified text in the default language SHOULD automatically overwrite the existing translations made in all alternate languages.

UIVersion: A number that represents the visual state of the site.

ClientTag: A number that represents the state of the application files of the site.

AnonymousPermMask: A **WSS Rights Mask** (section 2.2.2.14) that indicates the permissions granted to a user that is anonymous, or has no specific permissions, to this site. The value MUST be NULL for sites that do not exist.

{SiteFlags}: A **Site Collection Flags** (section 2.2.2.9) value that indicates the settings for the site collection that contains this site.

{SitePortalURL}: The URL for a different site registered on the site for use in **navigation structures** as a parent location.

{SitePortalName}: The display name of a different site registered on the site for use in navigation structures as a parent location.

MeetingCount: If this site is a meeting workspace (that is, its site template identifier in **ProvisionConfig** is set to 2), this value indicates the number of meetings that are configured.

DefTheme: The name of a theme that is used as part of the display of the site.

CachedNav: An implementation-specific serialization of navigation structure information to display in front-end web server elements associated with this site.

CachedInheritedNav: This contains an implementation-specific serialization of navigation structure information to display in front-end web server elements associated with the root navigation elements for this site.

CachedNavDirty: If this value is not set to "0", the cached navigation information for this site MUST be regenerated.

CachedDataVersion: A counter incremented on every change to the cached navigation information, used for internal conflict detection.

NavParentWebId: The site identifier of the site to be treated as the parent of this site when displaying navigation elements in the front-end web server.

FirstUniqueAncestorWebId: The site identifier of the closest site in this site's ancestor chain that does not inherit security settings from its **parent site**.

ScopeId: The scope identifier (section 2.2.1.8) of the security scope containing the site.

DbNow: The current time in **UTC** format on the back-end database server.

Acl: The binary serialization of the **WSS ACL Format** (section 2.2.4.6) ACL for this site. This is either explicitly defined or inherited from the parent object of the site.

{RequestAccessEmail}: The email address for a user who has the authority to grant access to the site, if any. This can be used by the front-end web server to generate an email to request access for a user who does not have access to the site.

Ancestry: An implementation-specific serialization of the navigation structure information for the site.

ProductVersion: The implementation-specific version identifier used to create the site.

tp_Id: The user identifier (section 2.2.1.13) of the current user.

tp_SiteAdmin: Set to "1" if the current user is an administrator of the site.

tp_IsActive: Set to "1" if the current user is an active user in the site collection containing this site.

tp_Login: The login information of the current user.

tp_Email: The email address of the current user.

tp_Title: The display name of the current user.

tp_Notes: Notes about the current user.

tp_ExternalTokenLastUpdated: A time stamp in UTC format specifying the time when the **External Group Token** (section 2.2.4.2) for the current user was last updated.

tp_Flags: A **WSS user Flags** value of the specified user

tp_Token: A **WSS user Token** (section 2.2.4.10) value specifying the site **group** membership of the current user.

UserId: The site membership identifier of the current user. This is NULL if the current user has not been added to the set of members of this site.

{SiteSecurityVersion}: The current security information version of the site collection.

tp_Locale: An LCID referring to the **locale** value to be used when displaying messages in the front-end web server for the current user. If this value is NULL, the site setting for Locale is used instead.

tp_TimeZone: The **Time Zone Identifier** for the time zone to be used when displaying time values for the current user. If this value is NULL, the site setting for **TimeZone** is used instead.

tp_Time24: A bit flag that specifies whether to use a 24-hour time format when displaying time values to the current user. If this parameter is set to "1", the 24-hour time format is be used; otherwise, the 12-hour time format is be used. If this value is NULL, the site setting for **Time24** MUST be used instead.

tp_CalendarType: The **Calendar** type to be used when processing date values for the current user. If this value is NULL, the site setting for **CalendarType** MUST be used instead.

tp_AdjustHijriDays: If the **tp_CalendarType** value is "6", this specifies the number of days to extend or reduce the current month in Hijri calendars for the current user. If this value is NULL, the site setting for **AdjustHijriDays** MUST be used instead.

tp_AltCalendarType: The **Calendar** type of an alternate calendar to be used when displaying calendars in views for the current user. If this value is NULL, the site setting for **AltCalendarType** MUST be used instead.

tp_CalendarViewOptions: A **Calendar View Options** type specifying the calendar options for the current user. If this value is NULL, the site setting for **CalendarViewOptions** MUST be used instead.

tp_WorkDays: A set of **Workdays Flags** specifying the weekdays defined as the workweek for the current user. If this value is NULL, the site setting for **WorkDays** MUST be used instead.

tp_WorkDayStartHour: The start time of the work day in minutes after midnight for the current user. If this value is NULL, the site setting for **WorkDayStartHour** MUST be used instead.

tp_WorkDayEndHour: The end time of the workday in minutes after midnight for the current user. If this value is NULL, the site setting for **WorkDayEndHour** MUST be used instead.

{SiteHashKey}: Contains binary information used when generating digital signatures of information associated with this site collection.

{UserInfoListId}: The list identifier (section 2.2.1.5) of the user information list for the site collection containing this site.

{RootWebId}: The site identifier for the root site of the site collection containing this site.

{SiteLastContentChange}: A time stamp in UTC format describing the time content was last changed within this site collection.

{SiteLastSecurityChange}: A time stamp in UTC format describing the time security settings were last changed within this site collection.

{RbsCollectionId}: The identifier for the remote blob storage collection for the site collection, or zero if remote blob storage is not configured for this database.

2.2.5.23 Site MetaInfo Result Set

The Site MetaInfo Result Set returns metainfo about a site.

The Site MetaInfo Result Set is defined using T-SQL syntax, as follows.

```
MetaInfo                        varbinary(max);
```

MetaInfo: Site metainfo in the form of a METADICT. This value can be NULL. The METADICT format is specified in the [MS-FPSE] METADICT section.

2.2.5.24 Unique Permissions Result Set

The Unique Permissions Result Set returns the ACL information for all the unique security scopes of folders and **list items** contained in a specified location.

The Unique Permissions Result Set is defined using T-SQL syntax, as follows.

```
ScopeId                         uniqueidentifier,
Acl                             varbinary(max),
AnonymousPermMask               bigint;
```

ScopeId: The scope identifier (section 2.2.1.8) of the security scope.

Acl: The binary serialization of the **WSS ACL Format** (section 2.2.4.6) ACL of the security scope.

AnonymousPermMask: A **WSS Rights Mask** (section 2.2.2.14) that indicates the permissions granted to an anonymous user, or a user who has no specific permissions on this security scope.

2.2.5.25 URL Result Set

The URL Result Set returns a list of URLs, one for the root folder of each site. The URL Result Set is defined using T-SQL syntax as follows.

```
FullUrl                         nvarchar (256)
```

FullUrl: The store-relative form URL of the site.

2.2.5.26 List Related Fields Result Set

The List Related fields Result Set returns information about all the relationship lookup fields whose target **list** is the specified list. The T-SQL syntax for the result set is as follows:

```
tp_WebId                   uniqueidentifier NOT NULL,
tp_Id                      uniqueidentifier NOT NULL,
FieldId                    uniqueidentifier NOT NULL,
DeleteBehavior             tinyint NOT NULL
```

tp_WebId: The site identifier (section 2.2.1.11) of the site that contains the relationship lookup field.

tp_Id: The list identifier (section 2.2.1.5) of the list that contains the relationship lookup field.

FieldId: The **field identifier** of the relationship lookup field.

DeleteBehavior: The relationship delete behavior on the relationship lookup field

2.2.6 SQL Structures

2.2.6.1 Configuration Object

A Configuration Object encapsulates as a **group** of application settings a set of metadata used to identify and manipulate those settings. Configuration Objects have the following fields.

2.2.6.1.1 Class

Applications use a Configuration Object **Class** to help distinguish the schemas of the data stored in different objects. **Class** is a complex data type with the following fields, defined in T-SQL syntax.

```
ClassId                         uniqueidentifier NOT NULL,
BaseClassId                     uniqueidentifier NOT NULL,
FullName                        nvarchar(256)    NOT NULL
```

ClassId: This MUST be a GUID that is different from all other **ClassIds** registered in the Configuration Database (section 3.1.1).

BaseClassId: A class is said to derive from the class referenced by **BaseClassId**. This allows a class hierarchy to be built. Many **ClassIds** can share a single **BaseClassId**. The **BaseClassId** of a Configuration Object **Class** MUST have the same value as a **ClassId** in the Configuration Database (section 3.1.1). If a Configuration Object **Class** has no base class, its **BaseClassId** MUST be identical to its **ClassId**.

FullName: An identifier that can be used by an application to associate a **ClassId** with a human-readable or machine-readable string.

The following classes are used during the execution of this protocol.

Class	ClassId
Alternate URL Collection	9920F486-2FF4-4d10-9532-E01979826585
Content Database	3D4F5451-1735-48bb-B920-76C1EC240B1D
Database Service Instance	3112E92F-B97D-481e-8CEB-03FDE15ED1A7
Farm	674DA553-EA77-44A3-B9F8-3F70D786DE6A
Server	E77AAF47-3CAC-4001-BC6B-5BCCB6486318
Web Application	113FB569-7520-4651-8FC4-E9F4F5887618
Web Service	45AD2BF2-4E3E-46A1-B477-126944C0ACEF

2.2.6.1.2 Id

A Configuration Object's **Id** is used to identify a single instance of a Configuration Object. As such, it MUST be different from the IDs of all other Configuration Objects in the Configuration Database (section 3.1.1). **Id** is defined in T-SQL format as follows.

```
Id                              uniqueidentifier NOT NULL
```

2.2.6.1.3 Name

Configuration Objects can also be identified by name. Unlike **Id**, a Configuration Object's name is only used to distinguish instances of Configuration Objects with the same parent and class. The combination of name, parent, and class MUST be unique among the set of Configuration Objects in a single Configuration Database (section 3.1.1). Name is defined in T-SQL format as follows.

```
    Name                        nvarchar(128) NOT NULL
```

2.2.6.1.4 Parent

Configuration Objects are organized in an ancestry hierarchy. Each Configuration Object MUST have one and only one parent, which MUST be a Configuration Object registered in the same Configuration Database (section 3.1.1). The set of Configuration Objects with a given parent are known as the parent's children. A Configuration Object's descendants are the set of Configuration Objects including the parent's children, the children's children, and so on.

The Configuration Object at the root of an ancestry hierarchy MUST be its own parent.

The Configuration Database MUST delete a Configuration Object when its parent is deleted.

A Configuration Object defines its parent through its own **ParentId** property, which is the Id of its parent. **ParentId** is defined in T-SQL format as follows.

```
    ParentId                        uniqueidentifier NOT NULL
```

2.2.6.1.5 Status

The **Status** property of a Configuration Object MUST conform to the data type defined in Configuration Object **Status** (section 2.2.2.2).

```
    Status         int NOT NULL
```

2.2.6.1.6 Version

Version is used to identify when a **Configuration Object** has been created, modified, or deleted. **Version** is defined in T-SQL format as follows.

```
    Version                         rowversion NOT NULL
```

2.2.6.1.7 Properties

Any configuration settings needed by an application that are not stored in one of the other Configuration Object fields can be stored in the properties field, which is defined using T-SQL format as follows.

```
    Properties                      nvarchar(max)
```

Successful execution requires parsing the properties of the following classes. Each of these classes stores its properties in an XML format. Because only a portion of each class' properties field is needed for successful execution, the values needed are identified using queries from the XML Path Language specified in [XPATH]. The schemata of the contents of other Configuration Objects' **Property Bags** will differ based on the needs of each application and are not specified in this document. Unless otherwise noted in the description column, each XPath query resolves to one and only one element value.

2.2.6.1.7.1 Alternate URL Collection

An **Alternate URL Collection Configuration Object** stores a **list** of absolute URLs. The parent of an **Alternate URL Collection** MUST be a **Farm Configuration Object**.

Property	XPath Query	Description
Alternate URL	/object/fld[attribute::name='m_Urls']/fld/object/sFld[attribute::name='m_RequestUri']	When executed against the properties of an **Alternate URL Collection** Configuration Object, this query MUST return one or more values containing absolute URLs. The URLs MUST contain only the portion of the incoming URL beginning with the scheme component and ending with the authority component (for example, "http://example.com:80"). The URLs MUST be unique within a Configuration Database (section 3.1.1).

2.2.6.1.7.2 Content Database

The Content Database Configuration Object stores information needed to establish a connection to a content database.

Property	XPath Query	Description
Username	/object/fld[attribute::name='m_Username']	If the value returned by this XPath query is not NULL or empty, the client uses SQL authentication to connect to the content database. The client MUST pass the value returned by the XPath query as the SQL authentication username.
Password	/object/fld[attribute::name='m_Password']	If the value returned by this XPath query is not NULL or empty, the client MUST include this value in the **user name** portion of the content database connection string.

2.2.6.1.7.3 Web Application

The **Web Application** Configuration Object stores a **list** of content databases used by the **web application** as well as information needed to parse URLs. The parent of a **Web Application** Configuration Object MUST be a **Web Service** Configuration Object.

Property	XPath Query	Description
Web Application Alternate URL Collection	/object/fld[attribute::name='m_AlternateUrlCollection']	This value is a string representation of a Configuration Object **ID**. It is used to associate a web application with an **Alternate URL Collection**. Each Alternate URL Collection MUST be referenced by zero or one web applications.
Prefix Name	/object/fld[attribute::name='m_Prefixes']/object/fld/fld/object/sFld[attribute::name='m_Name']	The value returned by this query MUST be a valid **URL Path** segment.
Prefix Type	/object/fld[attribute::name='m_Prefixes']/object/fld/fld/object/sFld[attribute::name='m_Type']	The value returned by this query MUST be the literal "ExplicitInclusion" or the literal "WildcardInclusion". ExplicitInclusion specifies that a path-based site collection can be created under the specified URL path. WildcardInclusion specifies that multiple path-based site collections can be created under the specified URL path.

2.2.6.2 Dependencies

In addition to the ancestry hierarchy defined in **Parent** (section 2.2.6.1.4), Configuration Objects can also define **Dependencies**. **Dependencies** can be used by an application to discover the **list** of Configuration Objects that depend on a given Configuration Object.

A **Dependency** is a complex type with the following fields, defined in T-SQL format as follows.

```
ObjectId                       uniqueidentifier NOT NULL
```

```
    DependantId                    uniqueidentifier NOT NULL
```

ObjectId: The **ID** of the Configuration Object that depends on another.

DependantId: The **ID** of the Configuration Object upon which the Configuration Object identified by **ObjectId** depends.

2.2.7 Tables and Views

2.2.7.1 AllDocs Table

The **AllDocs** table stores data for all documents in the content database. The **AllDocs** table is defined using T-SQL syntax as follows.

```
TABLE AllDocs(
    Id                             uniqueidentifier  NOT NULL,
    SiteId                         uniqueidentifier  NOT NULL,
    DirName                        nvarchar(256)     NOT NULL,
    LeafName                       nvarchar(128)     NOT NULL,
    WebId                          uniqueidentifier  NOT NULL,
    ListId                         uniqueidentifier  NULL,
    DoclibRowId                    int               NULL,
    Type                           tinyint           NOT NULL,
    SortBehavior                   tinyint           NOT NULL DEFAULT 0,
    Size                           int               NULL,
    ETagVersion                    AS
        CASE
            WHEN InternalVersion IS NULL
            THEN NULL
            ELSE ((InternalVersion + (BumpVersion * 256)) / 256)
        END,
    EffectiveVersion               AS
        CASE
            WHEN InternalVersion IS NULL
            THEN NULL
            ELSE InternalVersion + (BumpVersion * 256)

        END,
    InternalVersion                int               NULL,
    BumpVersion                    tinyint           NOT NULL DEFAULT 0,
    UIVersion                      int               NOT NULL DEFAULT 512,
    Dirty                          bit               NULL,
    ListDataDirty                  bit               NOT NULL DEFAULT 0,
    CacheParseId                   uniqueidentifier  NULL,
    DocFlags                       int               NULL,
    ThicketFlag                    bit               NULL DEFAULT 0,
    CharSet                        int               NULL,
    ProgId                         nvarchar(255)     NULL,
    TimeCreated                    datetime          NOT NULL,
    TimeLastModified               datetime          NOT NULL,
    NextToLastTimeModified         datetime          NULL,
    MetaInfoTimeLastModified       datetime          NULL,
    TimeLastWritten                datetime          NULL,
    DeleteTransactionId            varbinary(16)     NOT NULL DEFAULT 0x,
    SetupPathVersion               tinyint           NOT NULL DEFAULT 4,
    SetupPath                      nvarchar(255)     NULL,
    SetupPathUser                  nvarchar(255)     NULL,
    CheckoutUserId                 int               NULL,
    CheckoutDate                   datetime          NULL,
    CheckoutExpires                datetime          NULL,
    VersionCreatedSinceSTCheckout  bit               NOT NULL DEFAULT 0,
    LTCheckoutUserId               AS
        CASE WHEN DocFlags & 32 = 32
        THEN CheckoutUserId
        ELSE NULL
        END,
    VirusVendorID                int               NULL,
    VirusStatus                  int               NULL,
    VirusInfo                    nvarchar(255)     NULL,
    MetaInfo                     varbinary(max)    NULL,
    MetaInfoSize                 int               NULL,
    MetaInfoVersion              int               NOT NULL DEFAULT 1,
    UnVersionedMetaInfo          varbinary(max)    NULL,
    UnVersionedMetaInfoSize      int               NULL,
    UnVersionedMetaInfoVersion   int               NULL,
    WelcomePageUrl               nvarchar(260)     NULL,
    WelcomePageParameters        nvarchar(max)     NULL,
    IsCurrentVersion             bit               NOT NULL DEFAULT 1,
    Level                        tinyint           NOT NULL DEFAULT 1,
    CheckinComment               nvarchar(1023)    NULL,
    AuditFlags                   int               NULL,
    InheritAuditFlags            int               NULL,
    DraftOwnerId                 int               NULL,
    UIVersionString              AS
        CAST(UIVersion/512 AS nvarchar) + '.' + CAST(UIVersion%512 AS nvarchar),
    ParentId                     uniqueidentifier  NOT NULL,
    HasStream                    AS
        WHEN Type = 0 AND
             ((DocFlags & 256) = 256) AND
             SetupPath IS NULL OR (SetupPath IS NOT NULL AND
             ((DocFlags & 64) = 64))
        THEN 1
        ELSE 0
        END,
    ScopeId                      uniqueidentifier  NOT NULL,
    BuildDependencySet           varbinary(max)    NULL,
    ParentVersion                int               NULL,
    ParentVersionString          AS
        CAST(ParentVersion/512 AS nvarchar) + '.' +
        CAST(ParentVersion%512 AS nvarchar),
    TransformerId                uniqueidentifier  NULL,
    ParentLeafName               nvarchar(128)     NULL,
    IsCheckoutToLocal            AS
        CASE
            WHEN DocFlags & 512 = 512
            THEN 1

            ELSE 0
        END,
    CtoOffset                    smallint          NULL,
    Extension                    AS
        CASE
            WHEN
                CHARINDEX(N'.', LeafName COLLATE Latin1_General_BIN) > 0
                    THEN RIGHT(LeafName,
                        CHARINDEX(N'.', REVERSE(LeafName)
                        COLLATE Latin1_General_BIN)-1)
            ELSE N''
        END,
    ExtensionForFile             AS
        CASE
            WHEN Type = 0 AND
                CHARINDEX(N'.', LeafName COLLATE Latin1_General_BIN) > 0
                    THEN RIGHT(LeafName,
                        CHARINDEX(N'.', REVERSE(LeafName)
                        COLLATE Latin1_General_BIN)-1)
            ELSE N''
        END,
    ItemChildCount               int               NOT NULL DEFAULT 0,
    FolderChildCount             int               NOT NULL DEFAULT 0,
    FileFormatMetaInfo           varbinary(max)    NULL,
    FileFormatMetaInfoSize       int               NOT NULL DEFAULT 0,
    FFMConsistent                bit               NULL,
    ContentVersion                  int               NOT NULL DEFAULT 0,
    ListSchemaVersion               int               NULL,
    ClientId                        varbinary(16)     NULL);
```

Id: The **document identifier** (section 2.2.1.2) of the document.

SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the document.

DirName: The directory name of the document location.

LeafName: The leaf name of the document location.

WebId: The site identifier (section 2.2.1.11) of the site containing the document.

ListId: The **list identifier** (section 2.2.1.5) of the **list** containing the document, if any, or NULL if the document is not contained in a list.

DoclibRowId: The row identifier for the document within the containing document library or list, if applicable.

Type: An integer identifier specifying the document's **Document Store** type (section 2.2.2.4).

SortBehavior: An integer value specifying how an item SHOULD be sorted within a view. If this parameter is set to 1, then the item MUST be sorted like a folder. If this parameter is set to 0, it MUST be sorted like a file. If the document is a site, the value MUST be 2. If this parameter is set to 128, it MUST be sorted like a backward link. Other values are invalid.

Size: The size of the document stream in bytes. This parameter can be set to NULL or to 0 for items such as sites, folders, document libraries, and lists.

ETagVersion: A counter incremented any time a change is made to this document and used for conflict detection. This takes into account the pending version update in **BumpVersion**. This is a version number separate from the user-visible versioning system in **UIVersion** and **UIVersionString**.

EffectiveVersion: A counter incremented any time a change is made to the document and is used only internally. This takes into account the pending version update in **BumpVersion**.

InternalVersion: A counter incremented any time a change is made to the document and is used only internally to join to other tables. This does not take into account the pending version update in **BumpVersion**.

BumpVersion: A counter incremented any time the **InternalVersion** of a document needs to be updated. It stores the pending version update to the document.

UIVersion: A UI version number associated with the document. **UIVersion** defaults to 512, which corresponds to a displayed version of 1.0.

Dirty: Set to 1 if the document has dependencies that require further processing.

ListDataDirty: Set to 1 if the document requires subsequent fix-up of link information.

CacheParseId: An implementation-specific identifier for an internal dependency update process. Used to manage bulk updates to documents when processed for dependency fix-up.

DocFlags: A **Doc Flags** (section 2.2.2.3) value specifying information about the document.

ThicketFlag: If set to 1, then this indicates that the document is an auxiliary thicket file, one of a set of files supporting a thicket.

CharSet: An optional **character set** associated with the document. Any **Windows code page** identifier is valid for **CharSet**. This value can be NULL, indicating that no Windows code page is specified for the document.

ProgId: An optional preferred application to open this document. The **ProgId** is used to distinguish between different applications that save files with a given file extension (that is, different editing applications for HTML or XML files).

TimeCreated: A time stamp in **UTC** format specifying when this document was created.

TimeLastModified: A time stamp in UTC format. The value specifies when the document was last saved. This corresponds to the actual time when the document was last modified.

NextToLastTimeModified: The value of **TimeLastModified** from the previous time that the document was saved.

MetaInfoTimeLastModified: A time stamp in UTC format specifying when the metadata information for the document was last changed.

TimeLastWritten: A time stamp in UTC format specifying when any changes were made to the document stream. This does not reflect changes made strictly to the metainfo or other item properties.

DeleteTransactionId: The transaction identifier for the implementation-specific deleted items **Recycle Bin**. A value of 0x specifies that this document is not marked as deleted. In the **Docs View** (section 2.2.7.4), **DeleteTransactionId** MUST equal 0x, because the **Docs View** only displays items that are not marked as deleted.

SetupPathVersion: For a ghosted document, this parameter governs the **SetupPath** fragment relative to the setup path location. The following values are valid.

Value	Description
NULL	This is NOT a ghosted document.
2	The **SetupPath** is relative to the install location of Windows SharePoint Services 2.0 on the front-end web server (for example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\60).
3	The **SetupPath** is relative to the install location of Windows SharePoint Services 3.0 on the front-end web server (for example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\12).
4	The **SetupPath** is relative to the install location of Microsoft SharePoint Foundation 2010 on the front-end web server (for example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\14).

SetupPath: For a document that is now or once was ghosted, this contains the setup path fragment relative to the base setup path specified by the **SetupPathVersion** value, where the content **stream** of this document can be found. This value MUST be NULL if the document was never ghosted.

SetupPathUser: If this document is now or once was ghosted, then this contains the login name of the user who created the ghosted document. This value is undefined for documents that were never ghosted.

CheckoutUserId: If the document is checked out, then this parameter contains the user identifier (section 2.2.1.13) of the user who has the document checked out. Otherwise, this parameter is NULL.

CheckoutDate: A time stamp in UTC format indicating when this document was checked out.

CheckoutExpires: A time stamp in UTC format indicating when the **short-term lock** for this document will expire.

VersionCreatedSinceSTCheckout: If this parameter is 1, then the document version has been incremented since the document last had a short-term lock established. This is used to prevent more than one new version of the document from being created while a short-term lock is established.

LTCheckoutUserId: If the document is currently checked out, then this parameter is a calculated column containing the value of **CheckoutUserId**. Otherwise, this parameter is NULL.

VirusVendorID: The identifier of the virus scanner that processed this document. This value MUST be NULL if this document has not been processed by a virus scanner.

VirusStatus: An enumerated type specifying the current virus check status of this document. This value MUST be NULL if the document has not been processed by a virus scanner. See **Virus Status** (section 2.2.3.18) for a list of valid values.

VirusInfo: A string containing a provider-specific message returned by the virus scanner when it last processed the document. This value MUST be NULL if the document has not been processed by a virus scanner.

MetaInfo: A METADICT for the document. The METADICT format is specified in [MS-FPSE] section 2.2.2.2.11.

MetaInfoSize: The size, in bytes, of the document metadata info.

MetaInfoVersion: An integer value that tracks the version of the document metadata info.

UnVersionedMetaInfo: A METADICT holding all version-independent metadata for the document.

UnVersionedMetaInfoSize: The size in bytes of **UnVersionedMetaInfo**.

UnVersionedMetaInfoVersion: An integer value that tracks the version of the **UnVersionedMetaInfo** metadata.

WelcomePageUrl: If the document is a folder, then this optionally specifies a page to redirect to when the folder is requested with an HTTP GET operation. The URL is relative to the URL of the folder itself and MUST be subsumed by that folder. Attempts to break out of the folder, such as "../../default.aspx", are not valid.

WelcomePageParameters: This parameter MUST contain any URL parameters configured for the welcome page. This value contains a query string starting with "?" or a hash parameter starting with "#".

IsCurrentVersion: If this value is set to 1, then this row is for a current version of the document (out of all the rows corresponding to the given **DirName**, **LeafName**, each with a different **Publishing Level** type (section 2.2.2.6) value). Otherwise, this row is for a historical version of the document.

Level: A **Publishing Level** type value specifying the publishing level of this version of the document as checked out, draft, or published.

CheckinComment: An optional comment string associated with the document when it was last checked in or published.

AuditFlags: An **Audit Flags** (section 2.2.2.1) value determining the operations to be tracked on this document.

InheritAuditFlags: An **Audit Flags** value determining the operations to be tracked on this document's children.

DraftOwnerId: If the current publishing level of the document is a draft, then this is the user identifier (section 2.2.1.13) of the user who has that draft version checked out. Otherwise, this MUST be set to NULL.

UIVersionString: A calculated column presenting the value of the **UIVersion** column as a displayed version string, in the following format: major version value, followed by '.', followed by the minor version value. For example, **UIVersion** values 512, 513, 514, and 1024 correspond to **UIVersionString** values of 1.0, 1.1, 1.2, and 2.0, respectively.

ParentId: The document identifier (section 2.2.1.2) of the document's parent container. For example, the document identifier of the folder or document library containing this subfolder or document. This value MUST NOT be NULL, because every item but one has a parent container. A special empty document identifier, '00000000-0000-0000-0000-000000000000', marks the parent container of the topmost item (the root site) in the site collection.

HasStream: A calculated bit indicating whether the document has an associated document stream. This MUST be set to "1" if:

- The **SetupPath** is NULL and the document's **Document Store** type is 0 (file) and it is allowed to contain a stream.
- The **SetupPath** is not NULL and the document content is unghosted.

If neither condition applies, then the value MUST be "0".

ScopeId: The scope identifier (section 2.2.1.8) of the scope of the document.

BuildDependencySet: A binary array holding implementation-specific information about dependent documents. NULL indicates that there is no information about dependencies. A **BuildDependencySet** of size 0 indicates a document with no dependencies.

ParentVersion: If the document is a transformed version of another document, then this is the **UIVersion** value from the original document. This value MUST be NULL if the document is not the product of a document transformation.

ParentVersionString: A calculated column presenting the value of the **ParentVersion** column as a displayed version string. See **UIVersionString** for displayed version format and examples.

TransformerId: If the document is a transformed version of another document, then this MUST be the GUID of the agent that performed the transformation. Otherwise, **TransformerId** MUST be NULL.

ParentLeafName: If the document is a transformed version of another document, then this is the leaf name of the original document, if the original document is in the same folder as the transformed document. This value MUST be NULL if the document is not the product of a document transformation.

IsCheckoutToLocal: A calculated bit indicating whether the document is checked out to a client's local disk. This parameter MUST be set to "1" if the **DocFlags** bit array includes 512 (long-term check-out to local); otherwise, it MUST be set to "0".

CtoOffset: A content type order offset. This is an implementation-specific value used by list and document library views to allow custom ordering of the menu items under the "New" button. This value can be NULL.

Extension: A calculated field holding the file extension part, if applicable, from the **LeafName** value. If the **LeafName** value contains a "." character, **Extension** MUST hold the string following the last "."; otherwise, **Extension** MUST be set to the empty string. For example, if the full leaf name is 'document.docx.bac', **Extension** MUST be "bac".

ExtensionForFile: A calculated field holding the file extension part, if applicable, from the **LeafName** value. If the document's **Document Store** type is set to "0", indicating that it is a file type,

ExtensionForFile MUST be identical to **Extension**; otherwise, **ExtensionForFile** value MUST be an empty string.

ItemChildCount: The number of nonfolder children of this document.

FolderChildCount: The number of folder children of this document.

FileFormatMetaInfo: A METADICT holding the file format metadata info of this document.

FileFormatMetaInfoSize: The size, in bytes, of the file format metadata info.

FFMConsistent: A bit flag specifying whether the file format metadata info is in a consistent state.

ContentVersion: A counter incremented any time a change is made to the binary contents of this document; used for internal conflict detection.

ListSchemaVersion: A counter that keeps track of the schema version of the list containing this document for a document that can be parsed.

ClientId: An identifier assigned by an external client to keep track of this document.

2.2.7.2 AllDocStreams Table

The **AllDocStreams** table stores the document stream and related data for documents with content **streams** stored in the content database on the BEDS. The **AllDocStreams** table is defined using T-SQL syntax, as follows.

```
TABLE AllDocStreams(
      Id                              uniqueidentifier   NOT NULL,
      SiteId                          uniqueidentifier   NOT NULL,
      InternalVersion                 int NOT NULL,
      Content                         varbinary(max)              NULL,
      RbsId                           varbinary(800) NULL
);
```

Id: The **document identifier** (section 2.2.1.2) of the document.

SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the document.

Size: The size of the document stream, in bytes. For a ghosted document, this can be NULL.

InternalVersion: An integer value specifying the implementation-related **internal version number** of this version of the document.

Content: The document stream with the content of the document. For a ghosted document, this can be NULL.

RbsId: If remote blob storage is enabled and the document's content is contained in a remote data store, this MUST be the remote blob storage identifier for the document's content. If remote blob storage is disabled or the document's content is not contained in a remote data store, this MUST contain NULL. For additional information regarding remote blob storage, see [MS-WSSO] section 2.1.2.3.8.

2.2.7.3 AllUserData Table

The **AllUserData** table stores data for all **list** and document library list items. The table provides a fixed number of generic columns in various data types, affording storage for application-defined variable schemas. A list item can be represented by more than one row in this table, if its list's schema requires more entries of a particular data type than are available in a single row. Application-defined metadata for documents in document libraries also resides in **AllUserData**, and it is accessed via joins with the **Docs View** (section 2.2.7.4).

The table is defined using T-SQL syntax, as follows.

```
TABLE AllUserData (
tp_Id                          int                  NOT NULL,
tp_ListId                      uniqueidentifier     NOT NULL,
tp_SiteId                      uniqueidentifier     NOT NULL,
tp_RowOrdinal                  tinyint                  NOT NULL    DEFAULT ((0)),
tp_Version                     int                  NOT NULL,
tp_Author                      int                  NULL,
tp_Editor                      int                  NULL,
tp_Modified                    datetime             NULL,
tp_Created                     datetime             NULL,
tp_Ordering                    varchar(512)         NULL,
tp_ThreadIndex                 varbinary(512)       NULL,
tp_HasAttachment               bit                  NOT NULL    DEFAULT ((0)),
tp_ModerationStatus            int                  NOT NULL    DEFAULT ((0)),
tp_IsCurrent                   bit                  NOT NULL    DEFAULT ((1)),
tp_ItemOrder                   float                NULL,
tp_InstanceID                  int                  NULL,
tp_GUID                        uniqueidentifier     NOT NULL    DEFAULT (newid()),
tp_CopySource                  nvarchar(260)        NULL        DEFAULT (NULL),
tp_HasCopyDestinations         bit                  NULL        DEFAULT ((0)),
tp_AuditFlags                  int                  NULL,
tp_InheritAuditFlags           int                  NULL,
tp_Size                        int                  NOT NULL    DEFAULT ((0)),
tp_WorkflowVersion             int                  NULL,
tp_WorkflowInstanceID          uniqueidentifier     NULL,
tp_ParentId                    uniqueidentifier     NOT NULL,
tp_DocId                       uniqueidentifier     NOT NULL,
tp_DeleteTransactionId         varbinary(16)        NOT NULL    DEFAULT (0x),
tp_ContentTypeId               varbinary(512)       NULL,
nvarchar1                      nvarchar(255)        NULL,
nvarchar2                      nvarchar(255)        NULL,
nvarchar3                      nvarchar(255)        NULL,
nvarchar4                      nvarchar(255)        NULL,
nvarchar5                      nvarchar(255)        NULL,
nvarchar6                      nvarchar(255)        NULL,
nvarchar7                      nvarchar(255)        NULL,

nvarchar8                      nvarchar(255)        NULL,
ntext1                         nvarchar(max)                NULL,
ntext2                         nvarchar(max)                NULL,
ntext3                         nvarchar(max)                NULL,
ntext4                         nvarchar(max)                NULL,
sql_variant1                   sql_variant          NULL,
nvarchar9                      nvarchar(255)        NULL,
nvarchar10                     nvarchar(255)        NULL,
nvarchar11                     nvarchar(255)        NULL,
nvarchar12                     nvarchar(255)        NULL,
nvarchar13                     nvarchar(255)        NULL,
nvarchar14                     nvarchar(255)        NULL,
nvarchar15                     nvarchar(255)        NULL,
nvarchar16                     nvarchar(255)        NULL,
ntext5                         nvarchar(max)                NULL,
ntext6                         nvarchar(max)                NULL,
ntext7                         nvarchar(max)                NULL,
ntext8                         nvarchar(max)                NULL,
sql_variant2                   sql_variant          NULL,
nvarchar17                     nvarchar(255)        NULL,
nvarchar18                     nvarchar(255)        NULL,
nvarchar19                     nvarchar(255)        NULL,
nvarchar20                     nvarchar(255)        NULL,
nvarchar21                     nvarchar(255)        NULL,
nvarchar22                     nvarchar(255)        NULL,
nvarchar23                     nvarchar(255)        NULL,
nvarchar24                     nvarchar(255)        NULL,
ntext9                      nvarchar(max)               NULL,
ntext10                     nvarchar(max)               NULL,
ntext11                     nvarchar(max)               NULL,
ntext12                     nvarchar(max)               NULL,
sql_variant3                sql_variant         NULL,
nvarchar25                  nvarchar(255)       NULL,
nvarchar26                  nvarchar(255)       NULL,
nvarchar27                  nvarchar(255)       NULL,
nvarchar28                  nvarchar(255)       NULL,
nvarchar29                  nvarchar(255)       NULL,
nvarchar30                  nvarchar(255)       NULL,
nvarchar31                  nvarchar(255)       NULL,
nvarchar32                  nvarchar(255)       NULL,
ntext13                     nvarchar(max)               NULL,
ntext14                     nvarchar(max)               NULL,
ntext15                     nvarchar(max)               NULL,
ntext16                     nvarchar(max)               NULL,
sql_variant4                sql_variant         NULL,
nvarchar33                  nvarchar(255)       NULL,
nvarchar34                  nvarchar(255)       NULL,
nvarchar35                  nvarchar(255)       NULL,
nvarchar36                  nvarchar(255)       NULL,
nvarchar37                  nvarchar(255)       NULL,
nvarchar38                  nvarchar(255)       NULL,
nvarchar39                  nvarchar(255)       NULL,
nvarchar40                  nvarchar(255)       NULL,
ntext17                     nvarchar(max)               NULL,
ntext18                     nvarchar(max)               NULL,
ntext19                     nvarchar(max)               NULL,
ntext20                     nvarchar(max)               NULL,
sql_variant5                sql_variant         NULL,
nvarchar41                  nvarchar(255)       NULL,
nvarchar42                  nvarchar(255)       NULL,
nvarchar43                  nvarchar(255)       NULL,
nvarchar44                  nvarchar(255)       NULL,
nvarchar45                  nvarchar(255)       NULL,
nvarchar46                  nvarchar(255)       NULL,
nvarchar47                  nvarchar(255)       NULL,
nvarchar48                  nvarchar(255)       NULL,
ntext21                     nvarchar(max)               NULL,
ntext22                     nvarchar(max)               NULL,
ntext23                     nvarchar(max)               NULL,

ntext24                     nvarchar(max)               NULL,
sql_variant6                sql_variant         NULL,
nvarchar49                  nvarchar(255)       NULL,
nvarchar50                  nvarchar(255)       NULL,
nvarchar51                  nvarchar(255)       NULL,
nvarchar52                  nvarchar(255)       NULL,
nvarchar53                  nvarchar(255)       NULL,
nvarchar54                  nvarchar(255)       NULL,
nvarchar55                  nvarchar(255)       NULL,
nvarchar56                  nvarchar(255)       NULL,
ntext25                     nvarchar(max)               NULL,
ntext26                     nvarchar(max)               NULL,
ntext27                     nvarchar(max)               NULL,
ntext28                     nvarchar(max)               NULL,
sql_variant7                sql_variant         NULL,
nvarchar57                  nvarchar(255)       NULL,
nvarchar58                  nvarchar(255)       NULL,
nvarchar59                  nvarchar(255)       NULL,
nvarchar60                  nvarchar(255)       NULL,
nvarchar61                  nvarchar(255)       NULL,
nvarchar62                  nvarchar(255)       NULL,
nvarchar63                  nvarchar(255)       NULL,
nvarchar64                  nvarchar(255)       NULL,
ntext29                     nvarchar(max)               NULL,
ntext30                     nvarchar(max)               NULL,
ntext31                     nvarchar(max)               NULL,
ntext32                        nvarchar(max)              NULL,
sql_variant8                   sql_variant        NULL,
int1                           int                NULL,
int2                           int                NULL,
int3                           int                NULL,
int4                           int                NULL,
int5                           int                NULL,
int6                           int                NULL,
int7                           int                NULL,
int8                           int                NULL,
int9                           int                NULL,
int10                          int                NULL,
int11                          int                NULL,
int12                          int                NULL,
int13                          int                NULL,
int14                          int                NULL,
int15                          int                NULL,
int16                          int                NULL,
float1                         float              NULL,
float2                         float              NULL,
float3                         float              NULL,
float4                         float              NULL,
float5                         float              NULL,
float6                         float              NULL,
float7                         float              NULL,
float8                         float              NULL,
float9                         float              NULL,
float10                        float              NULL,
float11                        float              NULL,
float12                        float              NULL,
datetime1                      datetime           NULL,
datetime2                      datetime           NULL,
datetime3                      datetime           NULL,
datetime4                      datetime           NULL,
datetime5                      datetime           NULL,
datetime6                      datetime           NULL,
datetime7                      datetime           NULL,
datetime8                      datetime           NULL,
bit1                           bit                NULL,
bit2                           bit                NULL,
bit3                           bit                NULL,
bit4                           bit                NULL,
bit5                           bit                NULL,

bit6                           bit                NULL,
bit7                           bit                NULL,
bit8                           bit                NULL,
bit9                           bit                NULL,
bit10                          bit                NULL,
bit11                          bit                NULL,
bit12                          bit                NULL,
bit13                          bit                NULL,
bit14                          bit                NULL,
bit15                          bit                NULL,
bit16                          bit                NULL,
uniqueidentifier1              uniqueidentifier   NULL,
tp_Level                       tinyint            NOT NULL   DEFAULT ((1)),
tp_IsCurrentVersion            bit                NOT NULL   DEFAULT (CONVERT
                                                                     ([bit],(1),0)),
tp_UIVersion                   int                NOT NULL   CONSTRAINT
                                       [AllUserData_DEFAULT_UIVersionDEFAULT((512)),
tp_CalculatedVersion           int                NOT NULL   CONSTRAINT
                                       [AllUserData_DEFAULT_CalculatedVersionDEFAULT((0)),
tp_UIVersionString             AS     ((CONVERT([nvarchar],[tp_UIVersion]/(512),0)+'.')
                                      + CONVERT([nvarchar],[tp_UIVersion]%(512),0)),
tp_DraftOwnerId                int                NULL       DEFAULT   (NULL),
tp_CheckoutUserId              int                NULL       DEFAULT   (NULL)
);
```

tp_Id: The identifier for the list item, uniquely identifying it within the **AllUserData** table.

tp_ListId: The list identifier (section 2.2.1.5) of the list or document library containing the list item.

tp_SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the list item.

tp_RowOrdinal: The zero-based ordinal index of this row in the set of rows representing the list item. Additional rows are used to represent list items that have more application-defined columns of one or more data types than can fit in a single row in the **AllUserData** table.

tp_Version: A counter incremented any time a change is made to the list item, used for internal conflict detection. Due to the mapping of application properties to the generic columns schema in this table, changes to application schema as well as property values can affect a version increment.

tp_Author: The user identifier (section 2.2.1.13) for the user who created the list item.

tp_Editor: The user identifier for the user who last edited the list item.

tp_Modified: A date and time value in **UTC** format specifying when this list item was last modified.

tp_Created: A date and time value in UTC format specifying when this list item was created.

tp_Ordering: A concatenation of time stamp values in yyyyMMddHHmmss format, specifying the threading structure of the list items in a legacy discussion board list (a list with a **List Base** type (section 2.2.3.11) of 3). For list items in all other types of lists, this parameter MUST be NULL.

tp_ThreadIndex: A binary structure specifying the list item's position within a legacy discussion board list (a list with a **List Base** type of 3). For list items in all other types of list, this parameter MUST be NULL.

tp_HasAttachment: A bit set to 1 if the list item has an attachment associated with it; otherwise, it is set to 0.

tp_ModerationStatus: A **Moderation Status** (section 2.2.3.13) value indicating the current moderation approval status of the list item.

tp_IsCurrent: A bit set to 1 if this is a current version of this list item; otherwise, it is set to 0.

tp_ItemOrder: A value used to calculate the relative order in which to view the list item when displayed with other list items from the same list.

tp_InstanceID: If this list item is associated with a particular instance of a recurring meeting, this is the integer ID of that instance. For all other list items, this MUST be NULL.

tp_GUID: A value uniquely identifying this list item.

tp_CopySource: The URL used as a source for the list item. If this list item was not copied from a source list item, this value MUST be NULL.

tp_HasCopyDestinations: This bit is set to 1 if destination locations for the list item to be copied to have been set. If the list item does not have a destination location set, this value MUST be 0.

tp_AuditFlags: An **Audit Flags** (section 2.2.2.1) value determining the operations to be tracked on this list item.

tp_InheritAuditFlags: An **Audit Flags** value for the operations to be tracked on this list item, as determined from parent container **Audit Flags** settings.

tp_Size: The sum of the size, in bytes, of the content of application-schema columns in the list item. This does not include the size of the **stream** for list items that have an associated stream.

tp_WorkflowVersion: If the list item is part of a **workflow**, this stores an integer denoting the state of this list item within that workflow. Otherwise, this value MUST be NULL.

tp_WorkflowInstanceID: A workflow identifier (section 2.2.1.16) value for the currently active workflow instance on this list item. If the list item is not part of a workflow, this value MUST be NULL.

tp_ParentId : The **document identifier** (section 2.2.1.2) of the item's parent container. For example, the document identifier of the folder or document library containing this subfolder or item. This value MUST NOT be NULL, because every item but one has a parent container. A special empty document identifier, "00000000-0000-0000-0000-000000000000", marks the parent container of the topmost item (the root site) in the site collection.

tp_DocId: The document identifier of the list item.

tp_DeleteTransactionId: An identifier for use with an implementation-specific deleted items recycle bin. This MUST equal 0x if the list item is nondeleted.

tp_ContentTypeId: The binary identifier of the content type associated with the list item.

The next seven columns are duplicated a number of times within the table definition. This is indicated using a suffix "#", which is replaced with a numeral in the column names, described as follows. The number of times each column is duplicated varies and is indicated for each column. Each **group** of columns is dedicated to hold application-defined fields of a different data type, described as follows.

nvarchar#: Columns for application-defined fields that hold values of type **nvarchar**. The 64 columns are named nvarchar1 to nvarchar64. If the column does not contain data, this value MUST be NULL.

ntext#: Columns for application-defined fields that hold values of type **nvarchar(max)**. The 32 columns are named ntext1 to ntext32. If the column does not contain data, this value MUST be NULL.

sql_variant#: Columns for application-defined fields that hold values of type **sql_variant**. The 8 columns are named sql_variant1 to sql_variant8. If the column does not contain data, this value MUST be NULL.

int#: Columns for application-defined fields that hold values of type **int**. The 16 columns are named int1 to int16. If the column does not contain data, this value MUST be NULL.

float#: Columns for application-defined fields that hold values of type **float**. The 12 columns are named float1 to float12. If the column does not contain data, this value MUST be NULL.

datetime#: Columns for application-defined fields that hold values of type **datetime**. The 8 columns are named datetime1 to datetime8. If the column does not contain data, this value MUST be NULL.

bit#: Columns for application-defined fields that hold values of type **bit**. The 16 columns are named bit1 to bit16. If the column does not contain data, this value MUST be NULL.

uniqueidentifier1: A column for an application-defined field of type **uniqueidentifier**. If the column does not contain data, this value MUST be NULL.

tp_Level: A **publishing level** value specifying the publishing status of this version of the list item.

tp_IsCurrentVersion: A **bit** indicating whether this row corresponds to a current version or an historical version of the list item. This value MUST be 1 if this row contains a current version. Otherwise, it MUST be 0.

tp_UIVersion: The UI version number associated with this list item. The default value of **tp_UIVersion** is 512, which corresponds to a displayed version of 1.0.

tp_CalculatedVersion: This contains the UI version number if this is an historical version of the list item. This MUST be 0 if the list item is a current version.

tp_UIVersionString: A calculated column containing the value of the **tp_UIVersion** column as a displayed version string.

tp_DraftOwnerId: The identifier of the user who created this list item as a draft. This value is non-NULL only if the list item exists and is a draft version.

tp_CheckoutUserId: The identifier of the user who checked out this list item. This value is non-NULL only if the list item exists and is checked out.

2.2.7.4 Docs View

The **Docs** View reflects all rows in the table **AllDocs** (section 2.2.7.1) that are not deleted (that is, all table rows where DeleteTransactionId is equal to 0x). **Docs** View records include items that have a URL, such as files, folders, **lists**, document libraries, and sites, referred to collectively as documents.

See **AllDocs** (section 2.2.7.1) for the T-SQL syntax description of the structure of this view.

2.2.7.5 GroupMembership Table

The **GroupMembership** table stores a **list** of the **principals** that are members of each site **group** in a site collection. The GroupMembership table is defined using T-SQL syntax, as follows.

```
TABLE GroupMembership(
      SiteId                         uniqueidentifier   NOT NULL,
      GroupId                        int                NOT NULL,
      MemberId                       int                NOT NULL
);
```

SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the principals and site groups.

GroupId: The group identifier of the site group that the principal specified by **MemberId** is a **member** of.

MemberId: The user identifier (section 2.2.1.13) of a principal that is a member of the site group specified by **GroupId**.

2.2.7.6 Sec_SiteGroupsView

The **Sec_SiteGroupsView** is a view into site **group** information with one site group per row. Site groups available through this view are owned by a user or domain group, or by a site group. If a user or domain group owns the site group, the **OwnerIsUser** bit is set to 1, and the **GroupID**, **GroupName**, **GroupDescription**, **GroupSiteID**, **GroupOwner** and **GroupOwnerIsUser** fields are set to NULL.

If the site group is owned by a site group, the **OwnerIsUser** bit is set to 0, and the **UserID**, **UserSID**, **UserName**, **UserLogin**, **UserEmail**, **UserNotes**, **UserSiteAdmin** and **UserDomainGroup** are set to NULL. The **Sec_SiteGroupsView** table is defined using T-SQL syntax, as follows.

```
VIEW Sec_SiteGroupsView(
      ID                                   int,
      Title                                nvarchar(255),
      Description                          nvarchar(512),
      SiteWebId                            uniqueidentifier,
      Owner                                int,
      OwnerIsUser                          bit,
      OwnerGlobal                          bit,
      Type                                tinyint,
      Global                              bit,
      SiteID                              uniqueidentifier,
      UserID                              int,
      UserSID                             varbinary(512),
      UserName                            nvarchar(255),
      UserLogin                           nvarchar(255),
      UserEmail                           nvarchar(255),
      UserNotes                           nvarchar(1023),
      UserSiteAdmin                       bit,
      UserDomainGroup                     bit,
      UserFlags                           int,
      GroupID                             int,
      GroupName                           nvarchar(255),
      GroupDescription                    nvarchar(512),
      GroupSiteID                         uniqueidentifier,
      GroupOwner                          int,
      GroupOwnerIsUser                    bit,
      DLAlias                             nvarchar(128),
      DLErrorMessage                      nvarchar(512),
      DLFlags                             int,
      DLJobId                             int,
      DLArchives                          varchar(4000),
      RequestEmail                        nvarchar(255),
      Flags                               int
);
```

ID: The identifier for the site group. This parameter MUST NOT be NULL.

Title: The user-friendly display name for the site group. This parameter MUST NOT be NULL.

Description: The site group description.

SiteWebId: The site collection identifier (section 2.2.1.9) of the site collection that contains the site group information. This value is the same as returned in **SiteId**. For non-user-owned groups, this column name will be **WebId**. This parameter MUST NOT be NULL.

Owner: The user identifier (section 2.2.1.13) or site group identifier (section 2.2.1.10) of the site group owner. This parameter MUST NOT be NULL.

OwnerIsUser: A bit flag specifying whether the site group owner is a user or a site group. When the value in the **Owner** field is a user identifier for a user or a domain group, the **OwnerIsUser** flag MUST be set to 1. When the value in the **Owner** field is a site group identifier, the **OwnerIsUser** flag MUST be set to 0. This parameter MUST NOT be NULL.

OwnerGlobal: A bit flag specifying whether the view contains ownership information for a user or domain group, or for a site group.

When the owner is a site group, the **OwnerGlobal** flag MUST be set to 1, and the **GroupID**, **GroupName**, **GroupDescription**, **GroupSiteID**, **GroupOwner** and **GroupOwnerIsUser** fields MUST be populated with information from the site group owning this site group, and the **UserID**, **UserSID**, **UserName**, **UserLogin**, **UserEmail**, **UserNotes**, **UserSiteAdmin** and **UserDomainGroup** MUST be NULL.

When the owner is a user or domain group, the **OwnerGlobal** flag MUST be set to 0, the **UserID**, **UserSID**, **UserName**, **UserLogin**, **UserEmail**, **UserNotes**, **UserSiteAdmin** and **UserDomainGroup** MUST be populated with information from the user or domain group owning this site group, and the **GroupID**, **GroupName**, **GroupDescription**, **GroupSiteID**, **GroupOwner** and **GroupOwnerIsUser** MUST be NULL.

Type: This value MUST be 0.

Global: This value MUST be 1.

SiteID: The site collection identifier (section 2.2.1.9) of the site collection. This parameter MUST NOT be NULL.

UserID: The user identifier of the owner of the site group. This MUST be NULL when the site group owner is a site group.

UserSID: The **SystemID** (section 2.2.1.12) of the owner of the site group. This MUST be NULL when the site group owner is a site group.

UserName: The user-friendly name of the owner of the site group. This MUST be NULL when the site group owner is a site group.

UserLogin: The login name of the owner of the site group. This MUST be NULL when the site group owner is a site group.

UserEmail: The email address of the owner of the site group. This MUST be NULL when the site group owner is a site group.

UserNotes: The notes associated with the owner of the site group. This MUST be NULL when the site group owner is a site group.

UserSiteAdmin: A bit flag specifying whether the site group owner is a site collection administrator. When the site group owner is a site collection administrator, the **UserSiteAdmin** flag MUST be set to 1. If the user or domain group owner of the site group is not a site collection administrator, the **UserSiteAdmin** flag MUST be set to 0. This MUST be NULL when the site group owner is a site group.

UserDomainGroup: A bit flag specifying whether the site group owner is a domain group. When the site group owner is a domain group, the flag MUST be set to 1. When the site group owner is a user or domain group, the flag MUST be set to 0. This flag MUST be NULL when the site group owner is a site group.

UserFlags: Contains the **UserInfo Flags** (section 2.2.2.11) value describing the owner of the site group. This MUST be NULL when the site group owner is a site group.

GroupID: The site group identifier (section 2.2.1.10) of the owner of this site group. This MUST be NULL when the site group owner is a user or domain group.

GroupName: The user-friendly name of the owner of this site group. This MUST be NULL when the site group owner is a user or domain group.

GroupDescription: The description of the owner of this site group. This MUST be NULL when the site group owner is a user or domain group.

GroupSiteID: The site collection identifier (section 2.2.1.9) of the site collection containing the owner of this site group. This MUST be NULL when the site group owner is a user or domain group.

GroupOwner: The user identifier (section 2.2.1.13) or site group identifier of the owner of the site group that owns this site group. This MUST be NULL when the site group owner is a user or domain group, because only an owner that is a site group can have a group owner.

GroupOwnerIsUser: A bit flag specifying whether the site group that owns this site group is in turn owned by a user or domain group, or by a site group. When the owner of the site group that owns this site group is a user or domain group, the **GroupOwnerIsUser** flag MUST be set to 1. This flag MUST be set to 0 when the site group owner is owned by a site group. This MUST be NULL when the site group owner is a user or domain group.

DLAlias: The email **distribution list** address for the site group. This value MUST be NULL if the group has no email distribution list address.

DLErrorMessage: Contains the most recent error message returned by an asynchronous email distribution list operation, if any.

DLFlags: Contains a bit field of status flags for the email distribution list associated with this site group. This parameter MUST be one of the values listed in the following table.

Value	Description
0x00000000	None.
0x00000001	**Pending** - If this bit is set, the email distribution list associated with this site group currently has a pending asynchronous operation.
0x00000002	**Dirty** - If this bit is set, the email distribution list associated with this group is **dirty**. The email distribution list is considered dirty when A) an asynchronous operation is pending, and B) the site group has been modified since the asynchronous operation was initiated. A dirty email distribution list needs to have its status and membership synchronized as soon as the pending operation completes (whether it succeeds or fails).
0x00000004	**PendingJobIsRename** - The type of the currently pending asynchronous email distribution list operation, if any. When this bit is set, the pending operation is a Rename. When the bit is not set, it is a Create operation.

DLJobId: Contains the job ID of the currently pending asynchronous email distribution list operation or 0 if there is no pending operation.

DLArchives: An array of bytes containing a **list** of pairs of list identifiers (section 2.2.1.5) defining additional lists, which are recipients of email sent to the email distribution list via the address in DLAlias. Each Identifier MUST be stored as a string, with commas separating the list's parent site **document identifier** (section 2.2.1.2) and the list identifier, and with semicolons following each pair.

RequestEmail: The email address used to send a request to join or depart a site group.

Flags: Contains the membership permissions bit field for the site group. This parameter MUST NOT be NULL and MUST be one of the values listed in the following table.

Value	Description
0x00000000	Allow anyone to view the membership of the site group.
0x00000001	Only allow members of the site group to view the membership.
0x00000002	Allow members of the site group to edit the membership of the site group.
0x00000004	Allow users to request membership in this site group, and allow users to request to leave the site group. All requests MUST be sent to the email address specified by RequestEmail.
0x00000008	Automatically accept user requests to join or leave the site group. This bit MUST be set only when the bit 0x00000004 is also set.

2.2.7.7 Sites Table

The **Sites** table stores information about site collections. The table is defined using T-SQL syntax, as follows.

```
TABLE Sites(
      FullUrl                       nvarchar(255) DEFAULT N'',
      Id                            uniqueidentifier,
      NextUserOrGroupId             int,
      OwnerID                       int,
      SecondaryContactID              int,
      Subscribed                      bit,
      TimeCreated                     datetime,
      UsersCount                      int            DEFAULT ((1)),
      BWUsed                          bigint,
      DiskUsed                        bigint,
      SecondStageDiskUsed             bigint,
      QuotaTemplateID                 smallint,
      DiskQuota                       bigint,
      UserQuota                       int,
      DiskWarning                     bigint,
      DiskWarned                      datetime,
      CurrentResourceUsage            float DEFAULT 0 NOT NULL,
      AverageResourceUsage            float DEFAULT 0 NOT NULL,
      ResourceUsageWarning            float DEFAULT 0 NOT NULL,
      ResourceUsageMaximum            float DEFAULT 0 NOT NULL,
      BitFlags                        int,
      SecurityVersion                 bigint,
      CertificationDate               datetime,
      DeadWebNotifyCount              smallint,
      PortalURL                       nvarchar(260),
      PortalName                      nvarchar(255),
      LastContentChange               datetime       DEFAULT (getutcdate()),
      LastSecurityChange              datetime       DEFAULT (getutcdate()),
      AuditFlags                      int,
      InheritAuditFlags               int,
      UserInfoListId                  uniqueidentifier,
      UserIsActiveFieldRowOrdinal     int,
      UserIsActiveFieldColumnName     nvarchar(64),
      UserAccountDirectoryPath        nvarchar(255),
      RootWebId                       uniqueidentifier,
      HashKey                         binary(16),
      DomainGroupMapVersion           bigint,
      DomainGroupMapCacheVersion      bigint         DEFAULT ((-1)),
      DomainGroupMapCache             varbinary(max),
      HostHeader                      nvarchar(128),
      SubscriptionId                  varbinary(16),
      RbsCollectionId                 int            DEFAULT 0 NOT NULL,
      LastSMRequest                   datetime       DEFAULT NULL

);
```

FullUrl: The absolute URL of the site collection.

Id: The site collection identifier (section 2.2.1.9) of the site collection. Used only during upgrade.

NextUserOrGroupId: An integer value that is incremented when a new user or site **group** is added to the site collection. Indicates that the value of the next user or site group identifier (section 2.2.1.10) to be used.

OwnerID: The user identifier (section 2.2.1.13) of the user who owns the site collection.

SecondaryContactID: The user identifier of the user who is the secondary **contact** of the site collection.

Subscribed: A bit set to 1 to indicate that the site collection has been subscribed for implementation-specific notifications.

TimeCreated: The date and time in **UTC** format when the site collection was created.

UsersCount: The number of users in the site collection.

BWUsed: The number of sites in the site collection actively used. Serves an implementation-specific, usage-reporting feature.

DiskUsed: The size of disk space used to store content in the site collection, in bytes.

SecondStageDiskUsed: The size of disk space used to store the second-stage trash bin items for the site collection, in bytes.

QuotaTemplateID: An identifier of a quota template used to set the disk quota for the site collection.

DiskQuota: The maximum size, in bytes, of disk space that can be allocated by the site collection. A value of 0 indicates that no limit is set.

UserQuota: The maximum number of users that the site collection can contain. A value of 0 indicates that no limit is set.

DiskWarning: The size, in bytes, of disk space that can be allocated on the site collection before the disk warning email is sent. A value of 0 indicates that no warning email will be sent.

DiskWarned: A date and time value in UTC format set to the last time a warning was sent to the site collection owner about disk usage, if any.

CurrentResourceUsage: A float that specifies the current resource usage of the site collection up to the current date.

AverageResourceUsage: A float that specifies the average resource usage of the site collection over the configured period of days.

ResourceUsageWarning: A float which specifies the current resource usage warning setting. If the current resource usage exceeds it, a warning notification will be sent.

ResourceUsageMaximum: A float which specifies the max resource usage of the site collection.

BitFlags: A **Site Collection Flags** (section 2.2.2.9) value describing the site collection.

SecurityVersion: A version number incremented when changes are made to the site collection's permissions.

CertificationDate: The date and time, in UTC format, when the site collection was last confirmed by its owner as being used.

DeadWebNotifyCount: The number of times that a notification was sent to the site collection owner that the site collection will be deleted if the owner does not certify it as being used. (See **CertificationDate**.)

PortalURL: The URL of an external website designated as the sites' portal. This is an implementation-specific navigation links feature.

PortalName: The name of the external website referenced in the **PortalURL** column. Used for display in implementation-specific navigation features.

LastContentChange: The date and time value, in UTC format, that the content of the site collection was last changed.

LastSecurityChange: The date and time value, in UTC format, when the permissions on the site collection were last changed.

AuditFlags: An **Audit Flags** (section 2.2.2.1) value specifying the operations to be audited that are set directly on the site collection.

InheritAuditFlags: An **Audit Flags** value specifying the operations to be audited on the specified object that are inherited.

UserInfoListId: The list identifier (section 2.2.1.5) of a **list** holding user information for the site collection.

UserIsActiveFieldRowOrdinal: The ordinal of the column in the list identified by **UserInfoListId** that tracks whether a user is an active user in the site collection.

UserIsActiveFieldColumnName: The name of the column in the list identified by **UserInfoListId** that tracks whether a user is an active user in the site collection.

UserAccountDirectoryPath: A provider-specific user account path. The value is used for validation of added user accounts if the **Site Collection Flags** bit 0x00080000 is set in the **BitFlags** column.

RootWebId: The site identifier (section 2.2.1.11) of the site configured as the root site in the site collection.

HashKey: A hash key associated with the site collection.

DomainGroupMapVersion: The version of the domain group map.

DomainGroupMapCacheVersion: The version of the domain group map cache.

DomainGroupMapCache: A binary serialization of the domain group map cache, containing a mapping of external groups to the site groups of which they are members.

HostHeader: When used in Host Header mode, this contains the Host Header string associated with the site collection's **web application**.

SubscriptionId: The GUID of the implementation-specific subscription feature for the site collection.

RbsCollectionId: The identifier of the remote blob storage collection associated with the site collection, or zero if remote blob storage is not configured for this database.

LastSMRequest: The date and time value, in UTC format, when the storage metrics information for the site collection was last requested.

2.2.7.8 UserData View

The **UserData** View reflects all **list items** in the **AllUserData** table (section 2.2.7.3) that are current versions and not marked as deleted (that is, all table rows where **tp_IsCurrentVersion** is 1, **tp_CalculatedVersion** is 0, and **tp_DeleteTransactionId** is 0x).

See **AllUserData** (section 2.2.7.3) for the T-SQL syntax description of the structure of this view.

2.2.7.9 UserDataVersioned View

The **UserDataVersioned** View reflects all **list items** in the **AllUserData** table (section 2.2.7.3) that are not deleted (that is, all table rows where **tp_DeleteTransactionId** is 0x).

See **AllUserData** (section 2.2.7.3) for the T-SQL syntax description of the structure of this view.

2.2.7.10 UserInfo Table

The **UserInfo** table stores descriptive properties and security information about **principals** with access to a site collection. The **UserInfo** table is defined using T-SQL syntax, as follows.

```
TABLE UserInfo (
tp_SiteID                     uniqueidentifier   NOT NULL,
tp_Id                         int                NOT NULL,
tp_DomainGroup                bit                NOT NULL,
tp_SystemID                    varbinary(512)     NOT NULL,
tp_Deleted                     int                NOT NULL,
tp_SiteAdmin                   bit                NOT NULL,
tp_IsActive                    bit                NOT NULL DEFAULT (1),
tp_Login                       nvarchar(255)      NOT NULL,
tp_Title                       nvarchar(255)      NOT NULL,
tp_Email                       nvarchar(255)      NOT NULL,
tp_Notes                       nvarchar(1023)     NOT NULL,
tp_Token                       varbinary(max)     NULL,
tp_ExternalToken               varbinary(max)     NULL,
tp_ExternalTokenLastUpdated    datetime           NULL,
tp_Locale                      int                NULL,
tp_CalendarType                smallint           NULL,
tp_AdjustHijriDays             smallint           NULL,
tp_TimeZone                    smallint           NULL,
tp_Time24                      bit                NULL,
tp_AltCalendarType             uniqueidentifier   NULL,
tp_CalendarViewOptions         uniqueidentifier   NULL,
tp_WorkDays                    smallint           NULL,
tp_WorkDayStartHour            smallint           NULL,
tp_WorkDayEndHour              smallint           NULL,
tp_Mobile                      nvarchar(127)      NULL,
tp_Flags                       int                NOT NULL DEFAULT (0),
);
```

tp_SiteID: The site collection identifier (section 2.2.1.9) of the site collection associated with the principal.

tp_Id: The user identifier (section 2.2.1.13) of the principal, which MUST be unique within both this table and the **group** identifiers for site groups. Certain user identifiers are predefined, such as the system account (1073741823), the site collection owner, and the secondary site collection **contact**. User identifier values of 0 and -1 MUST NOT be used in the table.

tp_DomainGroup: A bit set to 1 if the principal is a domain group; otherwise, 0, specifying that the principal is a user.

tp_SystemID: The **SystemID** (section 2.2.1.12) of the principal.

tp_Deleted: A value specifying whether the principal is marked as deleted. If **tp_Deleted** is 0, the principal is not deleted. If **tp_Deleted** is not 0, the principal is marked as deleted, and the value is the user identifier that was associated with this principal. The deleted state can be used for user information, rather than dropping entries from the table, to preserve **list item** ownership information. A user or domain group with the **tp_Deleted** value set to nonzero can be restored by setting the **tp_Deleted** value to 0 and updating other fields as necessary.

tp_SiteAdmin: A bit set to 1 if the principal is a site collection administrator for the site collection.

tp_IsActive: A bit set to 1 if the principal is an active user in the site collection.

tp_Login: The login name for the principal. This value MUST NOT be empty.

tp_Title: The display name for the principal. This value MUST NOT be empty.

tp_Email: The email address for the principal. This value can be empty.

tp_Notes: A descriptive text about the principal. This value can be empty.

tp_Token: A **WSS user Token** (section 2.2.4.10) value specifying the site group membership of the principal. This can be NULL, indicating that the principal is not a **member** of a site group.

tp_ExternalToken: An **External Group Token** (section 2.2.4.2) value encoding information on external group membership derived from an external authentication role provider. This value can be

NULL, indicating that this user has never visited any site in the site collection. If this value is NULL, the value in **tp_ExternalTokenLastUpdated** MUST also be NULL.

tp_ExternalTokenLastUpdated: A **datetime** containing a time stamp in **UTC** format specifying the time when the value in **tp_ExternalToken** for the principal was last updated. This can be NULL if the value in **tp_ExternalToken** has never been updated.

tp_Locale: An LCID specifying the preferred locale value to be used when displaying messages in the front-end web server for the principal. This can be NULL, specifying that the system or site default locale can be used instead.

tp_CalendarType: The **Calendar** type (section 2.2.3.3) to be used when processing date values for the principal. This can be NULL, specifying that the site default **Calendar** type can be used instead.

tp_AdjustHijriDays: If the **tp_CalendarType** value is "6", this specifies the number of days to extend or reduce the current month in Hijri calendars for the principal. Otherwise, this value MUST be NULL and MUST be ignored.

tp_TimeZone: The Time Zone Identifier (section 2.2.3.17) for the time zone to be used when displaying time values for the principal. This can be NULL, specifying that the system or site default time zone can be used instead.

tp_Time24: A bit flag specifying whether a 24-hour time format can be used when displaying time values to the principal. If **tp_Time24** is 1, the 24-hour time format will be used; otherwise, the 12-hour time format will be used. This can be NULL, specifying that the system or site default can be used instead.

tp_AltCalendarType: The **Calendar** type of an alternate calendar to be used when displaying calendars in views for the principal. This can be NULL, specifying that the **tp_CalendarType** value can be used instead.

tp_CalendarViewOptions: A **Calendar View Options** type that specifies the calendar options for the principal. This can be NULL, specifying that the site default calendar view options can be used instead.

tp_WorkDays: A set of **Workdays Flags** that specify the weekdays defined as the work week for the principal. This can be NULL, specifying that the site default **Workdays** value can be used instead.

tp_WorkDayStartHour: The start time of the workday in minutes after midnight for the principal. This can be NULL, specifying that the site default workday start hour can be used instead.

tp_WorkDayEndHour: The end time of the workday in minutes after midnight for the principal. This can be NULL, specifying that the site default workday end hour can be used instead.

tp_Mobile: The mobile phone number of the **security principal**.

tp_flags: A 4-byte integer bit mask determining the user's options. See section 2.2.2.11.

2.2.7.11 Versions

The **Versions** table stores information used for object versioning during upgrade operations. The table is defined using T-SQL syntax, as follows.

```
TABLE Versions(
      VersionId                    uniqueidentifier NOT NULL,
      Version                      nvarchar(64) NOT NULL,
      Id                           int IDENTITY(1,1) NOT NULL,
      UserName                     nvarchar(255) NULL,
      TimeStamp                    datetime NULL,
      FinalizeTimeStamp            datetime NULL,
      Mode                         int NULL,
      ModeStack                      int NULL,
      Updates                        int NOT NULL DEFAULT((0)),
      Notes                          nvarchar (1024) NULL
);
```

VersionId: A GUID that identifies the version.

Version: The actual version string in the format "<major>.<minor>.<build>.<iteration>" (for example, "3.0.106.0").

Id: An integer value that is an SQL identity column.

UserName: A string containing the user name adding or updating the version.

TimeStamp: A time stamp in **UTC** format indicating when the version was added or updated.

FinalizeTimeStamp: Unused.

Mode: Unused.

ModeStack: Unused.

Updates: An integer value tracking the number of updates applied to a particular version. This value MUST be initialized to 1 when inserted and incremented each time the version is updated.

Notes: A string containing optional notes associated with the version.

2.2.8 XML Structures

No common XML Structures are defined in this protocol.

2.2.8.1 Namespaces

2.2.8.2 Simple Types

2.2.8.2.1 FALSE_Case_Insensitive_Else_Anything

This type is used to specify a **Boolean** value.

```
      <xs:simpleType name="FALSE_Case_Insensitive_Else_Anything">
        <xs:restriction base="xs:string">
          <xs:pattern value="[Ff][Aa][Ll][Ss][Ee]|.*" />
        </xs:restriction>
      </xs:simpleType>
```

2.2.8.2.2 FieldInternalType

The **FieldInternalType** type is used to specify a field internal type.

Note In the following XML, the **value** attribute is broken into multiple lines only for readability.

```
      <xs:simpleType name="FieldInternalType">
        <xs:restriction base="xs:string">
            <xs:pattern value="[iI][nN][tT][eE][gG][eE][rR]|[tT][eE][xX]
    [tT]|[nN][oO][tT][eE]|[dD][aA][tT][eE][tT][iI][mM][eE]|[cC][oO][uU][nN]
    [tT][eE][rR]|[cC][hH][oO][iI][cC][eE]|[lL][oO][oO][kK][uU][pP]|[bB][oO]
    [oO][lL][eE][aA][nN]|[nN][uU][mM][bB][eE][rR]|[cC][uU][rR][rR][eE][nN]
    [cC][yY]|[uU][rR][lL]|[cC][oO][mM][pP][uU][tT][eE][dD]|[tT][hH][rR][eE]
    [aA][dD][iI][nN][gG]|[gG][uU][iI][dD]|[mM][uU][lL][tT][iI][cC][hH][oO]
    [iI][cC][eE]|[gG][rR][iI][dD][cC][hH][oO][iI][cC][eE]|[cC][aA][lL][cC]
    [uU][lL][aA][tT][eE][dD]|[fF][iI][lL][eE]|[aA][tT][tT][aA][cC][hH][mM]
    [eE][nN][tT][sS]|[uU][sS][eE][rR]|[rR][eE][cC][uU][rR][rR][eE][nN][cC]
    [eE]|[cC][rR][oO][sS][sS][pP][rR][oO][jJ][eE][cC][tT][lL][iI][nN][kK]|
    [mM][oO][dD][sS][tT][aA][tT]|[eE][rR][rR][oO][rR]|[cC][oO][nN][tT][eE]
    [nN][tT][tT][yY][pP][eE][iI][dD]|[pP][aA][gG][eE][sS][eE][pP][aA][rR]
    [aA][tT][oO][rR]|[tT][hH][rR][eE][aA][dD][iI][nN][dD][eE][xX]|[wW][oO]
    [rR][kK][fF][lL][oO][wW][sS][tT][aA][tT][uU][sS]|[aA][lL][lL][dD][aA]
    [yY][eE][vV][eE][nN][tT]|[wW][oO][rR][kK][fF][lL][oO][wW][eE][vV][eE]
    [nN][tT][tT][yY][pP][eE]" />
        </xs:restriction>
      </xs:simpleType>
```

While the XSD pattern is normative, the values **are:**

- Integer
- Text
- Note
- DateTime
- Counter
- Choice
- Lookup
- Boolean
- Number
- Currency
- Url
- Computed
- Threading
- GUID
- MultiChoice
- GridChoice
- Calculated
- File
- Attachments
- User
- Recurrence
- CrossProjectLink
- ModStat

- Error
- ContentTypeId
- PageSeparator
- ThreadIndex
- WorkflowStatus
- AllDayEvent
- WorkFlowEventType

2.2.8.2.3 FieldAggregationAttribute

The **FieldAggregationAttribute** type is used to specify how values are promoted from an XML or site template document.

```
<xs:simpleType name="FieldAggregationAttribute">
    <xs:restriction base="xs:string">
        <xs:enumeration value="average" />
        <xs:enumeration value="count" />
        <xs:enumeration value="first" />
        <xs:enumeration value="last" />
        <xs:enumeration value="max" />
        <xs:enumeration value="merge" />
        <xs:enumeration value="min" />
        <xs:enumeration value="plaintext" />
        <xs:enumeration value="signature" />
        <xs:enumeration value="sum" />
    </xs:restriction>
</xs:simpleType>
```

The meanings of the values are specified in the following table. In the table, a matching node means an element or an attribute that matches a specified XPath query. The value of an element node is the concatenation of all character data directly contained by the element.

Value	Description
average	The average of the matching nodes, whose values can be interpreted as a floating point numbers. Empty string if no values match.
count	Count of the matching nodes.
first	The value of the first matching node.
last	The value of the last matching node.
max	The value of the largest matching node interpreted as a floating point number.
merge	The value of the nodes, in order, delimited by a newline character.
min	The value of the smallest matching node interpreted as a floating point number.
plaintext	If no nodes match, or if the first matching node is not an element, an empty string. Otherwise, renders the first matching node and its child elements with only character data shown.
signature	TRUE if the first matching node is an element and has child elements. FALSE if the first matching node is an element with no child elements. Otherwise, an empty string.

Value	Description
sum	The sum of the nodes that match and whose values can be interpreted as a floating point numbers.

2.2.8.2.4 FieldRefType

The **FieldRefType** type is used to specify the type of relationship given by a field reference definition within the field.

```
<xs:simpleType name="FieldRefType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Recurrence" />
    <xs:enumeration value="EventType" />
    <xs:enumeration value="UID" />
    <xs:enumeration value="RecurrenceId" />
    <xs:enumeration value="EventCancel" />
    <xs:enumeration value="StartDate" />
    <xs:enumeration value="EndDate" />
    <xs:enumeration value="RecurData" />
    <xs:enumeration value="Duration" />
    <xs:enumeration value="TimeZone" />
    <xs:enumeration value="XMLTZone" />
    <xs:enumeration value="CPLink" />
    <xs:enumeration value="LinkURL" />
    <xs:enumeration value="MasterSeriesItemID" />
    <xs:enumeration value="AllDayEvent" />
  </xs:restriction>
</xs:simpleType>
```

The meanings of the values are specified in the following table.

Value	Meaning
Recurrence	Specifies a recurring event.
EventType	Specifies an event type.
UID	Specifies a **unique identifier (UID)** for the event.
RecurrenceId	Specifies a recurrence Id for the event.
EventCancel	Specifies a cancelled event

Value	Meaning
StartDate	Specifies a start date for the event.
EndDate	Specifies an end date for the event.
RecurData	Specifies recurrence data for the event.
Duration	Specifies event duration.
TimeZone	Specifies a time zone for the event.
XMLTZone	Specifies a time zone in XML format.
CPLink	Specifies a cross program link.
LinkURL	Specifies a link **Uniform Resource Locator (URL)**.
MasterSeriesItemID	Specified the ID of the master recurrence record.
AllDayEvent	Specifies an all-day event.

2.2.8.2.5 FieldRichTextMode

The **FieldRichTextMode** type is used to specify a formatting mode for a rich text field.

```
<xs:simpleType name="FieldRichTextMode">
  <xs:restriction base="xs:string">
```

```
        <xs:enumeration value="Compatible"/>
        <xs:enumeration value="FullHtml"/>
        <xs:enumeration value="HtmlAsXml"/>
        <xs:enumeration value="ThemeHtml"/>
    </xs:restriction>
  </xs:simpleType>
```

The meanings of the values are specified in the following table.

Value	Description
Compatible	Specifies support for basic formatting, such as bold formatting.
FullHtml	Specifies support for more advanced formatting options, such as pictures, tables, and hyperlinks.
HtmlAsXml	Specifies support for **HTML** that is well-formed XML.
ThemeHtml	Specifies support for formatting with in-line style specifications. See [HTML] section 14.2.2.

2.2.8.2.6 IMEMode

The **IMEMode** type is used to specify a bias for an Input Method Editor (IME).

```
  <xs:simpleType name="IMEMode">
    <xs:restriction base="xs:string">
      <xs:enumeration value="inactive" />
      <xs:enumeration value="auto" />
      <xs:enumeration value="active" />
      <xs:enumeration value="disabled" />
    </xs:restriction>
  </xs:simpleType>
```

The meanings of the values are specified in the following table.

Value	Description
inactive	All characters are entered without the IME. Users can still activate the IME.
auto	Default. IME is not affected. This value has the same effect as not specifying the **ime-mode** attribute.
active	All characters are entered through the IME. Users can still deactivate the IME.
disabled	The IME is completely disabled. Users cannot activate the IME if the control has focus.

The reader MUST accept any value. The writer SHOULD write this attribute as one of the valid values for the **ime-mode** attribute to be applied to an <input> tag. See [HTML], section 17.4.

2.2.8.2.7 IntPositive

This type is used to specify a positive integer.

```
  <xs:simpleType name="IntPositive">
    <xs:restriction base="xs:int">
```

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

2.2.8.2.8 JoinType

The **JoinType** type specifies how to handle **Lookup** fields for which the target of the lookup does not exist.

```
  <xs:simpleType name="JoinType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="INNER" />
      <xs:enumeration value="LEFT OUTER" />
    </xs:restriction>
  </xs:simpleType>
```

The meanings of the values are specified in the following table.

Value	Description
INNER	Items whose target does not exist are omitted.
LEFT OUTER	Items whose target does not exist are included.

2.2.8.2.9 TextDirection

The **TextDirection** type is used to specify the preferred direction of displaying text.

```
  <xs:simpleType name="TextDirection">
    <xs:restriction base="xs:string">
      <xs:enumeration value="ltr" />
      <xs:enumeration value="rtl" />

      <xs:enumeration value="none" />
      <xs:enumeration value="None" />
    </xs:restriction>
  </xs:simpleType>
```

The meanings of the values are specified in the following table.

Value	Description
ltr	Specifies a left-to-right text direction.
rtl	Specifies a right-to-left text direction.
none	Specifies no hint for a text direction, and that the direction will follow the context of the page.
None	Same as none.

2.2.8.2.10 TRUE_If_Present

This type is used to specify a Boolean value.

```
      <xs:simpleType name="TRUE_If_Present">
        <xs:restriction base="xs:string"/>
      </xs:simpleType>
```

2.2.8.2.11 TRUEFALSE

This type is used to specify a Boolean value.

```
      <xs:simpleType name="TRUEFALSE">
        <xs:restriction base="xs:string">
          <xs:pattern value="[Tt][Rr][Uu][Ee]|[Ff][Aa][Ll][Ss][Ee]"/>
        </xs:restriction>
      </xs:simpleType>
```

2.2.8.2.12 TRUE_Case_Sensitive_Else_Anything

This type is used to specify a Boolean value.

```
      <xs:simpleType name="TRUE_Case_Sensitive_Else_Anything">
        <xs:restriction base="xs:string">
          <xs:pattern value="TRUE|.*" />
        </xs:restriction>
      </xs:simpleType>
```

2.2.8.2.13 UniqueIdentifierWithBracesUppercase

This type is used to specify a GUID.

```
    <xs:simpleType name="UniqueIdentifierWithBracesUppercase">
      <xs:restriction base="xs:string">
        <xs:pattern value="\{?[0-9A-F]{8}\-?[0-9A-F]{4}\-?[0-9A-F]{4}\-?[0-9A-F]{4}\-?[0-9A-
    F]{12}\}?"/>
      </xs:restriction>
    </xs:simpleType>
```

2.2.8.2.14 UniqueIdentifierWithOrWithoutBraces

This type is used to specify a GUID.

```
      <xs:simpleType name="UniqueIdentifierWithOrWithoutBraces">
        <xs:restriction base="xs:string">
          <xs:pattern value="\{?[0-9a-fA-F]{8}\-[0-9a-fA-F]{4}\-[0-9a-fA-F]{4}\-[0-9a-fA-F]{4}\-
    [0-9a-fA-F]{12}\}?"/>
        </xs:restriction>
      </xs:simpleType>
```

2.2.8.2.15 RelationshipDeleteBehaviorAttribute

The **RelationshipDeleteBehaviorAttribute** describes the relationship between a **list item** and the target list item of a **Lookup** field.

```
      <xs:simpleType name="RelationshipDeleteBehaviorAttribute">
        <xs:restriction base="xs:string">
          <xs:enumeration value="Cascade" />
          <xs:enumeration value="Restrict" />
          <xs:enumeration value="None" />
        </xs:restriction>
      </xs:simpleType>
```

The meanings of the values are specified in the following table.

Value	Description
Cascade	Specifies that when a list item in the target **list** is deleted, all related items in the list that contains the **Lookup** field are also deleted.
Restrict	Specifies that deleting a list item in the target list is prevented if there are any related items in the list that contains the **Lookup** field.
None	Specifies no restrictions.

2.2.8.3 Complex Types

2.2.8.3.1 CHOICEDEFINITION

The **CHOICEDEFINITION** type contains a choice for a **Choice** or **MultiChoice** field.

2.2.8.3.1.1 Schema

```
      <xs:complexType name="CHOICEDEFINITION" mixed="true">
          <xs:attribute name="JumpTo" type="xs:string" />
      </xs:complexType>
```

2.2.8.3.1.2 Attributes

JumpTo: Specifies the **Name** of the next field in a survey list to display when this choice is selected.

2.2.8.3.1.3 Child Elements

<Content>: Specifies a value for a choice in a **Choice** or **MultiChoice** field.

2.2.8.3.2 CHOICEDEFINITIONS

The CHOICEDEFINITIONS type contains a collection of choices for a **Choice** or **MultiChoice** field.

2.2.8.3.2.1 Schema

```
    <xs:complexType name="CHOICEDEFINITIONS">
        <xs:sequence>
          <xs:element name="CHOICE" type="CHOICEDEFINITION" minOccurs="0" maxOccurs="unbounded"
    />
        </xs:sequence>
    </xs:complexType>
```

2.2.8.3.2.2 Attributes

None.

2.2.8.3.2.3 Child Elements

CHOICE: Specifies a choice in a **Choice** or **MultiChoice** field.

2.2.8.3.3 FieldDefinition

A field definition describes the structure and format of a field that is used within a **list** or content type.

2.2.8.3.3.1 Schema

```
      <xs:complexType name="FieldDefinition" mixed="true">
        <xs:all>
          <xs:element name="CHOICES" type="CHOICEDEFINITIONS" minOccurs="0" maxOccurs="1" />
          <xs:element name="Customization" minOccurs="0" maxOccurs="1">
              <xs:complexType>
                  <xs:sequence>
                      <xs:any minOccurs="1" maxOccurs="1" namespace="##any"
processContents="skip" />
                  </xs:sequence>
              </xs:complexType>
          </xs:element>
          <xs:element name="Default" type="xs:string" minOccurs="0" maxOccurs="1" />
          <xs:element name="DefaultFormula" type="xs:string" minOccurs="0" maxOccurs="1" />
          <xs:element name="DisplayBidiPattern" minOccurs="0" maxOccurs="1">
            <xs:complexType>
              <xs:sequence>
                <xs:any minOccurs="0" maxOccurs="unbounded" namespace="##any"
processContents="skip" />
              </xs:sequence>
              <xs:anyAttribute processContents="skip" />
            </xs:complexType>
          </xs:element>
          <xs:element name="DisplayPattern" minOccurs="0" maxOccurs="1">
            <xs:complexType>
              <xs:sequence>
                <xs:any minOccurs="0" maxOccurs="unbounded" namespace="##any"
processContents="skip" />
              </xs:sequence>
              <xs:anyAttribute processContents="skip" />
            </xs:complexType>
          </xs:element>
          <xs:element name="FieldRefs" minOccurs="0" maxOccurs="1">
            <xs:complexType mixed="true">

              <xs:sequence>
                <xs:element name="FieldRef" type="FieldRefDefinitionField" minOccurs="0"
maxOccurs="unbounded" />
              </xs:sequence>
            </xs:complexType>
          </xs:element>
          <xs:element name="Formula" type="xs:string" minOccurs="0" maxOccurs="1" />
          <xs:element name="FormulaDisplayNames" type="xs:string" minOccurs="0" maxOccurs="1" />
          <xs:element name="MAPPINGS" type="MAPPINGDEFINITIONS" minOccurs="0" maxOccurs="1" />
          <xs:element name="ParserRefs" type="FieldParserRefs" minOccurs="0" maxOccurs="1" />
          <xs:element name="Validation" minOccurs="0" maxOccurs="1">
            <xs:complexType>
              <xs:sequence>
                <xs:any minOccurs="0" maxOccurs="unbounded" namespace="##any"
processContents="skip" />
              </xs:sequence>
              <xs:anyAttribute processContents="skip" />
            </xs:complexType>
          </xs:element>
        </xs:all>
        <xs:attribute name="Aggregation" type="FieldAggregationAttribute" default="first"/>
        <xs:attribute name="aggregation" type="xs:string" />
        <xs:attribute name="AllowDeletion" type="TRUEFALSE" default="TRUE" />
        <xs:attribute name="AllowHyperlink" type="TRUEFALSE" default="FALSE" />
        <xs:attribute name="AllowMultiVote" type="TRUEFALSE" default="FALSE" />
        <xs:attribute name="AppendOnly" type="TRUEFALSE" default="FALSE" />
```

```
<xs:attribute name="AuthoringInfo" type="xs:string" default=""/>
<xs:attribute name="BaseRenderingType" type="FieldInternalType" />
<xs:attribute name="BaseType" type="FieldInternalType" default="Text" />
<xs:attribute name="Calculated" type="xs:string" />
<xs:attribute name="CalType" type="xs:int" />
<xs:attribute name="CalendarType" type="xs:int" />
<xs:attribute name="CAMLRendering" type="TRUEFALSE" default="FALSE" />
<xs:attribute name="CanToggleHidden" type="TRUEFALSE" default="FALSE" />
<xs:attribute name="CountRelated" type="TRUE_If_Present" default="FALSE" />
<xs:attribute name="ClassInfo" type="xs:string" default="" />
<xs:attribute name="ColName" type="xs:string" />
<xs:attribute name="ColName2" type="xs:string" />
<xs:attribute name="Commas" type="TRUEFALSE" />
<xs:attribute name="Customization" type="xs:string" />
<xs:attribute name="Decimals" type="xs:int" default="-1"/>
<xs:attribute name="DefaultListField" type="TRUEFALSE" default="FALSE" />
<xs:attribute name="DefaultURLDesc" type="xs:string" />
<xs:attribute name="Description" type="xs:string" />
<xs:attribute name="Direction" type="TextDirection" default="none" />
<xs:attribute name="Dir" type="xs:string" />
<xs:attribute name="DisplaceOnUpgrade" type="TRUEFALSE" default="FALSE" />
<xs:attribute name="DisplayImage" type="xs:string" />
<xs:attribute name="DisplayName" type="xs:string" />
<xs:attribute name="DisplayNameSrcField" type="xs:string" />
<xs:attribute name="DisplaySize" type="xs:int" />
<xs:attribute name="Div" type="xs:string" default="1.0" />
<xs:attribute name="EnableLookup" type="TRUEFALSE" default="FALSE" />
<xs:attribute name="EnforceUniqueValues" type="TRUEFALSE" default="FALSE" />
<xs:attribute name="ExceptionImage" type="xs:string" />
<xs:attribute name="FieldRef" type="xs:string" />
<xs:attribute name="FillInChoice" type="TRUEFALSE" default="FALSE" />
<xs:attribute name="Filterable" type="TRUEFALSE" default="TRUE" />
<xs:attribute name="FilterableNoRecurrence" type="TRUEFALSE" default="FALSE" />
<xs:attribute name="ForcedDisplay" type="xs:string" />
<xs:attribute name="ForcePromoteDemote" type="TRUE_If_Present" />
<xs:attribute name="Format" type="xs:string" />
<xs:attribute name="FromBaseType" type="TRUEFALSE" default="FALSE" />
<xs:attribute name="GridEndNum" type="xs:int" />
<xs:attribute name="GridNATxt" type="xs:string" default="" />
<xs:attribute name="GridStartNum" type="IntPositive" />
<xs:attribute name="GridTxtRng1" type="xs:string" default="" />
<xs:attribute name="GridTxtRng2" type="xs:string" default="" />
<xs:attribute name="GridTxtRng3" type="xs:string" default="" />

<xs:attribute name="Group" type="xs:string" />
<xs:attribute name="HeaderImage" type="xs:string" />
<xs:attribute name="Height" type="xs:int" />
<xs:attribute name="Hidden" type="TRUEFALSE" default="FALSE" />
<xs:attribute name="ID" type="UniqueIdentifierWithOrWithoutBraces" />
<xs:attribute name="Id" type="xs:string" />
<xs:attribute name="IMEMode" type="IMEMode" />
<xs:attribute name="Indexed" type="TRUEFALSE" default="FALSE" />
<xs:attribute name="ImnHeader" type="xs:string" />
<xs:attribute name="IsolateStyles" type="TRUEFALSE" default="FALSE" />
<xs:attribute name="IsRelationship" type="TRUEFALSE" default="FALSE" />
<xs:attribute name="JoinColName" type="xs:string" default="tp_ID" />
<xs:attribute name="JoinRowOrdinal" type="xs:int" fixed="0" />
<xs:attribute name="JoinType" type="JoinType" default="LEFT OUTER" />
<xs:attribute name="JumpTo" type="xs:string" />
<xs:attribute name="JumpToFillinChoice" type="xs:string" />
<xs:attribute name="JumpToNo" type="xs:string" />
<xs:attribute name="JumpToYes" type="xs:string" />
<xs:attribute name="LCID" type="xs:int" />
<xs:attribute name="LinkToItem" type="TRUE_Case_Sensitive_Else_Anything" />
<xs:attribute name="LinkToItemAllowed" type="xs:string" />
<xs:attribute name="List" type="xs:string" />
<xs:attribute name="ListItemMenu" type="TRUE_Case_Sensitive_Else_Anything" />
<xs:attribute name="ListItemMenuAllowed type="xs:string" />
<xs:attribute name="Max" type="xs:float" />
```

```
    <xs:attribute name="MaxLength" type="xs:int" />
    <xs:attribute name="maxLength" type="xs:string" />
    <xs:attribute name="Min" type="xs:string" />
    <xs:attribute name="Mult" type="xs:string" />
    <xs:attribute name="Name" type="xs:string" use="required" />
    <xs:attribute name="NegativeFormat" type="xs:string" />
    <xs:attribute name="node" type="xs:string" />
    <xs:attribute name="Node" type="xs:string" />
    <xs:attribute name="NoEditFormBreak" type="TRUEFALSE" default="FALSE" />
    <xs:attribute name="NumLines" type="xs:string" default="6" />
    <xs:attribute name="OverwriteInChildScopes" type="TRUEFALSE" default="FALSE" />
    <xs:attribute name="Percentage" type="TRUEFALSE" default="FALSE" />
    <xs:attribute name="PIAttribute" type="xs:string" />
    <xs:attribute name="PITarget" type="xs:string" />
    <xs:attribute name="PrependId" type="TRUEFALSE" default="FALSE" />
    <xs:attribute name="Presence" type="TRUEFALSE" default="FALSE" />
    <xs:attribute name="PreviousName" type="xs:string" />
    <xs:attribute name="PrimaryKey" type="TRUEFALSE" />
    <xs:attribute name="PrimaryPIAttribute" type="xs:string" />
    <xs:attribute name="PrimaryPITarget" type="xs:string" />
    <xs:attribute name="ReadOnly" type="TRUEFALSE" default="FALSE" />
    <xs:attribute name="ReadOnlyEnforced" type="TRUEFALSE" default="FALSE" />
    <xs:attribute name="RelationshipDeleteBehavior type="RelationshipDeleteBehaviorAttribute"
default="None" />
    <xs:attribute name="RenderXMLUsingPattern" type="TRUEFALSE" default="FALSE" />
    <xs:attribute name="Required" type="TRUEFALSE" default="FALSE" />
    <xs:attribute name="RestrictedMode" type="TRUEFALSE" default="TRUE" />
    <xs:attribute name="ResultType" type="FieldInternalType" />
    <xs:attribute name="ResyncOnChange" type="TRUEFALSE" default="FALSE"/>
    <xs:attribute name="RichText" type="TRUEFALSE" default="FALSE" />
    <xs:attribute name="RichTextMode" type="FieldRichTextMode" default="Compatible" />
    <xs:attribute name="RowOrdinal" type="xs:int" default="0" />
    <xs:attribute name="RowOrdinal2" type="xs:int" default="0" />
    <xs:attribute name="Sealed" type="TRUEFALSE" default="FALSE" />
    <xs:attribute name="SeparateLine" type="TRUEFALSE" default="FALSE" />
    <xs:attribute name="SetAs" type="xs:string" />
    <xs:attribute name="ShowAddressBookButton" type="TRUEFALSE" default="FALSE" />
    <xs:attribute name="ShowField" type="xs:string" />
    <xs:attribute name="ShowInDisplayForm" type="TRUEFALSE" />
    <xs:attribute name="ShowInEditForm" type="TRUEFALSE" default="TRUE" />
    <xs:attribute name="ShowInFileDialog" type="TRUEFALSE" />
    <xs:attribute name="ShowInFileDlg" type="TRUEFALSE" />
    <xs:attribute name="ShowInListSettings" type="TRUEFALSE" />
    <xs:attribute name="ShowInNewForm" type="TRUEFALSE" default="TRUE" />

    <xs:attribute name="ShowInVersionHistory" type="TRUEFALSE" />
    <xs:attribute name="ShowInViewForms" type="TRUEFALSE" />
    <xs:attribute name="Sortable" type="TRUEFALSE" />
    <xs:attribute name="SourceID" type="xs:string" />
    <xs:attribute name="StaticName" type="xs:string" />
    <xs:attribute name="StorageTZ" type="xs:string" />
    <xs:attribute name="StripWS" type="xs:string" />
    <xs:attribute name="SuppressNameDisplay" type="TRUEFALSE" />
    <xs:attribute name="TextOnly" type="TRUEFALSE" />
    <xs:attribute name="Title" type="xs:string" />
    <xs:attribute name="TitleField" type="TRUEFALSE" default="FALSE" />
    <xs:attribute name="Type" type="xs:string" use="required" />
    <xs:attribute name="UniqueId" type="xs:string" />
    <xs:attribute name="UnlimitedLengthInDocumentLibrary" type="TRUEFALSE" />
    <xs:attribute name="URLEncode" type="TRUEFALSE" />
    <xs:attribute name="URLEncodeAsURL" type="TRUEFALSE" />
    <xs:attribute name="UserSelectionMode" type="xs:string" />
    <xs:attribute name="UserSelectionScope" type="xs:int" />
    <xs:attribute name="Version" type="xs:int" default="0" />
    <xs:attribute name="Viewable" type="FALSE_Case_Insensitive_Else_Anything" />
    <xs:attribute name="WebId" type="UniqueIdentifierWithoutBraces" />
    <xs:attribute name="Width" type="xs:int" />
    <xs:attribute name="WikiLinking" type="TRUEFALSE" default="FALSE" />
    <xs:attribute name="WorkflowStatusURL" type="xs:string" use="optional" />
    <xs:attribute name="XName" type="xs:string" />
    <xs:anyAttribute namespace="##other" processContents="lax" />
  </xs:complexType>
```

2.2.8.3.3.2 Attributes

Aggregation: For fields with the Node or node attribute, a reader MUST use this attribute to control promotion from XML or site template files.

For other fields, a reader MUST ignore this attribute.

aggregation: For fields whose **Type** attribute maps to the **Note** field internal type, the reader can permit a filtering user interface if this attribute contains the value merge (with case-insensitive comparison).<3> Otherwise a reader MUST ignore this attribute.

A writer SHOULD NOT include this attribute.

AllowDeletion: If this attribute is FALSE, then this is used to specify that a server MUST NOT permit the field to be removed from the schema of a **list**.

AllowHyperlink: For fields whose **Type** maps to the **Note** field internal type, and for which **RichText** is TRUE and **RichTextMode** is Compatible, a reader can use this attribute to determine if the editing UI allows insertion of hyperlinks. If this attribute is TRUE, a server or client presenting an editing user interface for this field MUST include the insert hyperlink command. Otherwise, the editing user interface MUST NOT include the insert hyperlink command.

AllowMultiVote: The reader MUST ignore this attribute.

AppendOnly: If TRUE, for fields whose **Type** maps to the **Note** field internal type, a client or server which presents a way of editing the value for this field MUST do so in a way that ensures preservation of old data in the field.

AuthoringInfo: Text describing the field to schema authors. A client or server presenting a schema-editing UI can display the text in this attribute.

BaseRenderingType: Field internal type that the server uses to render a field. If not present, the **Type** attribute does not map to an internal type in a way suitable for rendering.

The server MUST derive the value of this attribute from the **Type** attribute, overriding the client-specified value.

BaseType: Allows a schema author to override the internal storage format for choice fields. A reader MUST ignore this attribute unless the field's internal type is **Choice**, **MultiChoice**, or **GridChoice**. A writer MUST NOT set this attribute's value to **Choice**, **MultiChoice**, or **GridChoice**.

CalType: Associates a calendar type with a field. A reader SHOULD take this value into consideration when rendering the field or performing date calculations on it. If this attribute is not present, a reader MUST use the **CalendarType** attribute. If that is also not present, it SHOULD fall back to the user's preference, and if that is not specified, to the calendar type for the site.

CalendarType: This is equivalent to **CalType**. A reader MUST use the value of **CalType** if present, but if that attribute is not present, it MUST use this attribute for the same purpose. A writer SHOULD omit this attribute, using **CalType** instead where it is needed.

Calculated: The reader MUST ignore this attribute.

CAMLRendering: If TRUE, a field MUST be rendered using its **DisplayPattern** child element in a **list view**.

CanToggleHidden: If TRUE, a server MUST permit the user to change the value of the **Hidden** attribute. Otherwise, a server MUST prevent the Hidden attribute from changing.

CountRelated: A writer MUST NOT include this attribute unless the internal type of the field is **Lookup**. A writer MUST NOT include this attribute if the **Mult** attribute is present. A reader MUST ignore the attribute if the field internal type of the field is not **Lookup**.

ClassInfo: A writer SHOULD use values Menu or Icon, or it SHOULD omit the attribute. When rendering the value of the field in a view, a server SHOULD use this to format the field for a menu or an icon. The server MUST ignore values other than Menu or Icon and MUST fall back to the default behavior for other values.

ColName: A server SHOULD use this attribute to describe the physical storage location for the field. A client MUST ignore this attribute and a server MUST pick its own value for this attribute when accepting **CAML** from a client.

ColName2: This has the same restrictions and semantics as the **ColName** attribute except it describes a secondary physical storage location.

Commas: A writer SHOULD NOT use this attribute; a reader MUST ignore this attribute.

Customization: A writer SHOULD NOT use this attribute; a reader MUST ignore this attribute.

Decimals: Number of digits to render in the fractional part of a number. A writer SHOULD only use this attribute on fields whose field internal type is **Number** or **Calculated**. The reader MUST ignore this attribute on files with other internal types. A writer SHOULD omit this attribute rather than write the default value of "-1".

When rendering a numeric field, a server MUST use this attribute to determine the number of decimal places to render. For the special value of -1, trailing zeros MUST be omitted.

DefaultListField: A server or client can determine whether to request field values for a document or list item after a user action such as uploading a new document when editable fields are present on the list. When this attribute is set to TRUE, the server or client can ignore the presence of this field for the purposes of that determination.

DefaultURLDesc: For a field whose field internal type is **URL**, the value to use for the description part when the user does not provide a description. The default is to use the URL part of the field. If the field's internal type is not URL, a reader MUST ignore this attribute.

Description: Textual description of the field to be displayed in the user interface for schema authoring.

Direction: Specifies that HTML field rendering will be done in a left-to-right, right-to-left, or in the ambient context of the surrounding page.

Dir: A writer SHOULD NOT use this attribute; a reader MUST ignore this attribute.

DisplaceOnUpgrade: When TRUE, indicates that a server depends on the name of a field. The client MUST NOT set this attribute.

DisplayImage: When specified, this attribute supplies the URL, relative to the server's /_layouts/images folder, to render when displaying the field. A reader MUST ignore this attribute except for **CrossProjectLink** fields and **Recurrence** fields.

If not specified, a blank placeholder image is used instead.

DisplayName: Text to display in the user interface when referring to the field.

DisplayNameSrcField: When present, specifies the name of another field on the list to which the **DisplayName** of the current field is synchronized.

DisplaySize: Number of columns to offer when displaying the field.

Div: In fields whose field internal type is **Integer**, **Number**, or **Calculated**, a reader MUST interpret this as a floating point number by which the value is divided at render time.

In fields of other internal types, a reader MUST ignore this attribute and a writer MUST NOT include this attribute.

EnableLookup: Specifies whether the user interface for creating a **Lookup** field lists a **Computed** field as a possible target for the lookup. Unless the target field type is **Computed**, a writer SHOULD NOT include this attribute and a reader MUST ignore it.

EnforceUniqueValues: If this attribute is TRUE, then the server MUST NOT permit any list item to have the same value for the field as any other list item in the list, unless the value of the field is NULL.

ExceptionImage: Specifies an image to display in place of **DisplayImage** for recurrence fields for items that are exceptions to the recurrence. A writer MUST include this attribute for recurrence fields if the **DisplayImage** is also included and MUST NOT include it otherwise.

FieldRef: When present on a **Lookup** field, specifies the **Name** of another field on the list from which to obtain the local value for the lookup. **Lookup** fields for which this attribute is not specified supply their own storage for the local value.

FillInChoice: Whether a form generated to let the user edit a choice field or multichoice field allows values other than those listed in the **CHOICES** child element.

A reader MUST ignore this attribute except for **Choice** fields and **MultiChoice** fields.

Filterable: Whether items can be omitted or included from the results of a query based on the value of the field. True if the field can be filtered; false otherwise.

FilterableNoRecurrence: When TRUE, and the **Filterable** attribute is FALSE, specifies that items can be included or omitted from the results of a query in views that do not expand recurring **events**.

ForcedDisplay: If present, specifies a value for the field to display in place of the field's real value.

Format: Specifies how to render the field. The interpretation of this attribute and the range of legal values depend on the value of the **Type** attribute, as **follows:**

- For a **datetime** field, the value MUST be DateTime, DateOnly, TimeOnly, ISO8601, ISO8601Basic, ISO8601Gregorian, or not present. A reader MUST interpret a missing attribute to mean DateTime.
- For a **Choice** field, the value MUST be DropDown, RadioButtons, Checkbox, or not present. A reader MUST interpret a missing attribute to mean **DropDown**.
- For a **CrossProjectLink** field, the value MUST be EventList or not present.
- For **URL** fields, the value MUST be Image, Hyperlink, or not present. A reader MUST interpret a missing attribute as **Hyperlink**.
- For a **Boolean** field, the value MUST be CheckboxIcons, CheckboxIconsWithHeaderIcon, or not present.
- A reader MUST ignore this attribute for other field types.

ForcePromoteDemote: Specifies that the field SHOULD participate in property promotion and demotion.

FromBaseType: Indicates that the field was inherited from the list's base type. If TRUE, a server prevents the field from being deleted or having its type changed.

GridEndNum: Specifies the largest choice possible in a **GridChoice** field. A writer MUST include this attribute for **GridChoice** fields. A reader MUST ignore this attribute for other field types.

GridNATxt: Textual value to render for the choice that indicates "not applicable" in a **GridChoice** field. A reader MUST ignore this attribute for fields of other types.

GridStartNum: Specifies the smallest choice possible in a **GridChoice** field. A writer MUST include this attribute for **GridChoice** fields. A reader MUST ignore this attribute for fields of other types. A writer SHOULD set the value of this attribute to 1.

GridTxtRng1: Textual value to render for the choice that indicates the value corresponding to that specified by the **GridStartNum** attribute in a **GridChoice** field. A reader MUST ignore this attribute for other field types.

GridTxtRng2: Textual value to render for the choice that indicates the middle value in a **GridChoice** field. A reader MUST ignore this attribute for other field types.

GridTxtRng3: Textual value to render for the choice that indicates the value corresponding to that specified by the **GridEndNum** attribute in a **GridChoice** field. A reader MUST ignore this attribute for fields of other types.

Group: Name of the field that is used for grouping purposes. A reader MUST take a localized version of "Custom Columns" as the default value for this attribute.

HeaderImage: Specifies a URL, relative to the /_layouts/images folder on the server, to render in place of the field's **DisplayName** in views and forms.

Height: Height in pixels to render an image for a URL field whose **Format** attribute is equal to Image. If not present, specifies that the target image does not need to be scaled.

A reader MUST ignore this attribute except for URL fields where the **Format** attribute specifies Image and the **Width** attribute is present.

Hidden: Specifies whether to render a field in views or forms.

ID: GUID for the field.

Id: A writer SHOULD NOT include this attribute. A reader MUST ignore this attribute.

IMEMode: If specified, indicates a value for the ime-mode attribute to be applied to an <input> tag when reading the field.

Indexed: Specifies that a server can optimize for queries that filter on this field.

ImnHeader: This attribute is a marker. A reader MUST only check for the existence of the attribute. A writer MUST either not include the attribute or set its value to TRUE.

IsolateStyles: For a Text field whose **RichText** attribute is TRUE and whose **RichTextMode** attribute is **FullHtml**, this attribute specifies that a server will rewrite the HTML of the field to ensure it will not interfere with the rendering of the surrounding page.

A reader MUST ignore this attribute in other circumstances.

IsRelationship: Specifies whether the **Lookup** field is a relationship lookup field.

A reader MUST ignore this attribute if the field is not a **Lookup** field.

JoinColName: Specifies the physical storage location in the list referred to by the **List** attribute to compare with the local value of a **Lookup** field.

JoinRowOrdinal: A writer SHOULD NOT include this attribute.<4>

JoinType: Specifies how to treat items that have no corresponding matching item in the list indicated by the **List** attribute.

A reader MUST ignore this attribute except for **Lookup** fields.

JumpTo: Specifies the **Name** of the next field in a survey list when the field has a specified value.

A reader MUST ignore this attribute for **Boolean** fields.

JumpToFillinChoice: Specifies the **Name** of the next field in a survey list when a value other than those listed in the CHOICES child element is selected.

A reader MUST ignore this attribute except for **Choice** fields and **MultiChoice** fields for which the **FillInChoice** attribute is TRUE.

JumpToNo: Specifies the **Name** of the next field in a survey list for a **Boolean** field for which is false.

JumpToYes: Specifies the **Name** of the next field in a survey list for a **Boolean** field for which is true.

LCID: LCID used to render **Number** fields, **Currency** fields, and **DateTime** fields. When not present, specifies that the default LCID for the user is to be used. A reader MUST ignore this attribute for other types of fields.

LinkToItem: Specifies whether the field value is rendered as a link to the list item.

LinkToItemAllowed: Specifies the allowed state of the **LinkToItem** attribute. The allowed value are **Prohibited**, **Required** and **Allowed**. Value **Allowed** means the link can be optionally shown, **Required** means the link MUST be shown, **Prohibited** means the link can't be shown.

List: Specifies the foreign list for a **Lookup** field.

A writer MUST include this attribute for **Lookup** fields. A reader MUST ignore this attribute for other kinds of fields.

ListItemMenu: Specifies whether the field value is rendered with a drop-down option menu.

ListItemMenuAllowed: Specifies the allowed state of the **ListItemMenu** attribute. The allowed value are **Prohibited**, **Required** and **Allowed**. Value **Allowed** means the menu can be optionally shown, **Required** means the menu MUST be shown, **Prohibited** means the menu can't be shown.

Max: Specifies the maximum value for a **Number** field. A reader MUST ignore this attribute for other kinds of fields.

MaxLength: Specifies the maximum number of characters allowed in a **Text** field.

maxLength: The reader MUST ignore this attribute.

Min: Specifies the minimum value for a **Number** field. A reader MUST ignore this attribute for other kinds of fields.

Mult: In fields whose field internal type is **Integer**, **Number**, or **Calculated**, a reader MUST interpret this as a floating point number by which the value is multiplied at render time. In this context, a reader MUST interpret the default to be 1.0.

In the context of a **Lookup** field, a reader MUST interpret the presence of this attribute as an indication that the field is a multi-value lookup, and the absence of which as an indication that the field is a single-value lookup. In this context, a writer MUST either omit the attribute or set the value to TRUE.

In other contexts, a reader MUST ignore the attribute and the writer SHOULD NOT include the attribute.

Name: String that identifies the field within its list.

NegativeFormat: A writer SHOULD NOT include this attribute.<5> A reader MUST ignore this attribute.

node: A writer SHOULD use the **Node** attribute rather than this attribute. A reader MUST ignore this attribute when the **Node** attribute is present. Otherwise, a reader MUST use this attribute for the same purpose as it would use the **Node** attribute.

Node: When present, specifies an **XPath** to be used to read or write the value of the field into an **XML document**.

NoEditFormBreak: For fields where **RichText** is TRUE or where the Type is **Choice** or **MultiChoice**, a reader MUST ignore this attribute. For other fields, if TRUE, this attribute specifies that the following field to be rendered is on the same line as this field.

NumLines: Number of lines to render when accepting input for a **Note** field. A reader MUST ignore this attribute for other fields.

OverwriteInChildScopes: When set to TRUE, the old field is deleted and the new field is added. When set to FALSE, creation of the field fails.

Percentage: When specified on a **Number** field or a **Calculated** field that evaluates to a number, specifies that the field is rendered as a percentage (100 times its value and with a trailing "%" character).

A reader MUST ignore this attribute for other fields.

PIAttribute: A reader MUST ignore this attribute if either the **Node** or node attribute is present, or if both the **PrimaryPIAttribute** and **PrimaryPITarget** attributes are present, or if the **PITarget** attribute is not present.

Otherwise, the reader MUST treat this attribute as it would the **PrimaryPIAttribute** attribute.

PITarget: A reader MUST ignore this attribute if either the **Node** or **node** attribute is present, if both the **PrimaryPIAttribute** and **PrimaryPITarget** attributes are present, or if the **PIAttribute** attribute is not present.

Otherwise, the reader MUST treat this attribute as it would the **PrimaryPITarget** attribute.

PrependId: Whether the edit form for a Lookup field for which the **Multi** attribute is TRUE will present choices sorted by ID rather than by the **ShowField**.

A reader MUST ignore this attribute in other circumstances.

Presence: Whether a user field will be decorated with instant messaging presence information.

A reader MUST ignore this attribute for fields that are not user fields.

PreviousName: Indicates the value of the **Name** attribute of the field in an earlier revision of the containing list's schema.

PrimaryKey: A reader MUST ignore this attribute.

PrimaryPIAttribute: When present, specifies an attribute for an XML processing instruction to be used to read or write the value of the field into an XML document.

PrimaryPITarget: When present, specifies an XML processing instruction to be used to read or write the value of the field into an XML document.

ReadOnly: Whether the user is allowed to change the field through the user interface. If TRUE, only programmatic changes are allowed.

ReadOnlyEnforced: Whether the user is allowed to change the field by any means. If TRUE, the field can only be changed by the system.

R**elationshipDeleteBehavior:** Describes the relationship, if any, between an item in the list and an item that is the target of a **Lookup** field.

A reader MUST ignore this attribute when the field is not a **Lookup** field.

RenderXMLUsingPattern: Whether a **Computed** field is rendered using its **DisplayPattern** child element even when rendering for the object model or SOAP interfaces.

A reader MUST ignore this attribute except for **Computed** fields.

Required: Whether forms presented to accept data for the list item permits blank values for the field

RestrictedMode: For **Notes** fields for which the **RichText** attribute is TRUE, specifies whether the field permits cut, copy, paste, and insert image commands.

For other kinds of fields, the reader MUST ignore this attribute.

ResultType: If the **Type** attribute specifies that the internal type of the field is **Calculated**, a reader SHOULD use this attribute as in place of the internal type of the field to render results.

A reader MUST ignore this attribute if the internal type of the field is not **Calculated**.

ResyncOnChange: Whether the field's value changes value on form submission.

RichText: Whether a **Note** field contains formatted text; the exact text formatting is subject to the value of the **RichTextMode** attribute.

A reader MUST ignore this attribute except for **Note** fields.

RichTextMode: Specifies the serialization of formatted text.

RowOrdinal: The comments for the **ColName** attribute apply to this attribute as well.

RowOrdinal2: The comments for the **ColName** attribute apply to this attribute as well.

Sealed: Specifies how the server allows the field to be changed. If TRUE, the server MUST prevent changes to the field except for the **DisplayName**, **Description**, **Hidden**, and **Indexed** attributes and the **ParserRefs** child element.

SeparateLine: Determines whether or not the field is displayed on a different row for views that support this feature.

SetAs: A writer SHOULD NOT include this attribute. A reader MUST ignore this attribute.

ShowAddressBookButton: Determines whether or not a field's edit control includes a facility for entering users from an address book. A reader MUST ignore this attribute except when rendering the field in the context of a form that supports this functionality.

ShowField: Specifies the field in the foreign list that supplies the value of a **Lookup** field. A reader MUST ignore this attribute unless the field is a **Lookup** field.

If the **Type** attribute is user, then the default value for this attribute is **InmName**. If the **Type** is WorkflowStatus, the default value for this attribute is Status1. Otherwise, the default value is **Title**.

ShowInDisplayForm: Determines whether or not the field is shown or hidden in a form designed to show the item in a read-only fashion. If TRUE, then the field will be shown in the form for the item. If FALSE, then the field will not be shown in the form. If not specified, an implementation-specific algorithm is used to determine if the field will be shown.

ShowInEditForm: When FALSE, indicates that the field is not included in the form that is used to modify an item. When TRUE, indicates that the field's inclusion in such a form depends on implementation.

ShowInFileDialog: The reader MUST ignore this attribute.

ShowInFileDlg: Similar to **ShowInEditForm** except that it applies to a form designed to collect information about a document from within the context of an application.

ShowInEditDlg: Similar to **ShowInEditForm** except that it applies to a form designed to collect information about an item that is being modified.

ShowInListSettings: Similar to **ShowInEditForm** except that it applies to the field's inclusion in the user interface that is presented to list schema **editors**.

ShowInNewForm: Similar to **ShowInEditForm** except that it applies to the field's inclusion in a form designed to collect information about an item that is being created.

ShowInVersionHistory: Similar to **ShowInEditForm** except that it applies to the field's inclusion in a form designed to display a read-only rendition of a historical version of an item.

ShowInViewForms: Similar to **ShowInEditForm** except that it applies to the field's inclusion in the user interface that is presented to view authors.

Sortable: When FALSE, specifies that query results are not allowed to be ordered with respect to this field.

SourceID: URI suitable for use in an **XML namespace** in cases where the list schema is transformed to an XSD.

StaticName: Local part of an XML element name that is unique within the namespace given by the **SourceID** attribute.

StorageTZ: For **DateTime** fields, specifies the time zone in which the field is stored. If TRUE, then **UTC** is indicated; otherwise, the site's local time zone is indicated.

The reader MUST ignore this attribute unless the field's type is **DateTime**.

StripWS: The reader MUST ignore this attribute.

SuppressNameDisplay: When specified on a user field, the user's name is not displayed but all other rendering information is.

TextOnly: The reader MUST ignore this attribute.

Title: When present on a **CrossProjectLink** fields whose **DisplayImage** attribute is set and whose **Format** attribute specifies EventList, this attribute specifies a textual alternative to the image. When not present but the other conditions are specified, the reader MUST infer a default value from the **DisplayName** attribute. When these conditions are not met, the reader MUST ignore the attribute.

TitleField: Specifies that the file is suitable to use for a title of the item.

Type: Specifies the rendering properties and internal type of the field.

UniqueId: The reader MUST ignore this attribute.

UnlimitedLengthInDocumentLibrary: On a **Note** field or on a **Lookup** field for which the **Mult** attribute is TRUE, this specifies that the length of the content is not limited.

URLEncode: The reader MUST ignore this attribute.

URLEncodeAsURL: The reader MUST ignore this attribute.

UserSelectionMode: When present on a user field, **PeopleOnly** indicates that only **principals** that are users are selected, and **PeopleAndGroups** indicates that both principals that are users and those that are **groups** can be selected.

UserSelectionScope: When present on a user field, specifies the group to which the selected user or users belong.

Version: Current version of the field. The server MUST increment the value by 1 each time the field definition is changed. The client MUST ignore this attribute.

Viewable: Specifies whether or not the field was added to the **default view** of the list when it was added to the schema of the list. The reader SHOULD ignore this attribute.

WebId: For **Lookup** fields, specifies the identifier of the site that contains the list referenced by the **List** attribute.

Width: This attribute has the same semantics as the **Height** attribute except that it refers to the width of the image rather than the height.

WikiLinking: When TRUE on a **Note** field whose **RichText** attribute is TRUE, additional processing is done to facilitate entering hyperlinks within the content of the field. In other circumstances, the reader MUST ignore the attribute.

WorkflowStatusURL: The reader MUST ignore this attribute.

XName: String that is used to correlate the field in an external schema.

2.2.8.3.3.3 Child Elements

Customization: Provides an arbitrary XML document to provide vendor extensibility.

CHOICES: A set of **CHOICE** string elements that represent a **list** of available choices for a field. The reader MUST ignore **CHOICES** if the **Type** attribute is not Choice or MultiChoice. The writer SHOULD omit **CHOICES** if the **Type** attribute is not Choice or MultiChoice.<6>

Default: Default value of instances of data for this field in a list item. The reader MUST ignore this element when the **DefaultFormula** element is present and not empty.

DefaultFormula: Formula used to calculate the default value for this field in a list item.

DisplayBidiPattern: The implementation-specific XML that specifies the rendering of a **Computed** field to be used for a site with a locale identifier that specifies a bidirectional read order. A reader MUST ignore **DisplayBidiPattern** if the **Type** attribute is not **Computed**. The writer SHOULD omit **DisplayBidiPattern** if the Type attribute is not **Computed**.<7>

DisplayPattern: The implementation-specific XML that specifies the rendering of a **Computed** field. A reader MUST ignore **DisplayPattern** if the **Type** attribute is not **Computed**. The writer SHOULD omit **DisplayPattern** if the **Type** attribute is not **Computed**.<8>

FieldRefs: Specifies fields needed to render **DisplayPattern** or **DisplayBidiPattern**. DisplayPattern and **DisplayBidiPattern** MUST NOT use fields not listed in this element.

Formula: Formula used to calculate a value for a list item based on the value of other fields in the list. Reader MUST ignore **Formula** if the **Type** attribute is not **Calculated**. Writer SHOULD omit **Formula** if the **Type** attribute is not **Calculated**.<9>

FormulaDisplayNames: The **Formula** element formatted according to the regional settings of the site.

MAPPINGS: A set of **MAPPING** string elements that represents a canonical, language-agnostic identifier for a corresponding **CHOICE** with the value specified by the **MAPPING** element. The reader MUST ignore **MAPPINGS** if the **Type** attribute is not **Choice** or **MultiChoice**. The writer SHOULD omit **MAPPINGS** if the Type attribute is not **Choice** or **MultiChoice**.<10>

ParserRefs: Specifies alternate names for the field when accessed by the parsers listed in child elements of this element.

Validation: The implementation-specific XML that specifies the validation criteria to be used for items in this list.

2.2.8.3.4 FieldDefinitionDatabase

The **FieldDefinitionDatabase** type provides an overall container for field definitions in a Content Database.

2.2.8.3.4.1 Schema

```
    <xs:complexType name="FieldDefinitionDatabase">
      <xs:sequence>
        <xs:element name="Index" type="IndexDefinitionTP" minOccurs="0" maxOccurs="unbounded"
  />
        <xs:choice minOccurs="0" maxOccurs="unbounded">
          <xs:element name="FieldRef" type="FieldRefDefinitionTP" />
          <xs:element name="Field" type="FieldDefinitionTP" />
        </xs:choice>
      </xs:sequence>
    </xs:complexType>
```

2.2.8.3.4.2 Attributes

None.

2.2.8.3.4.3 Child Elements

Index: Specifies a composite field index to be used in this **list**.

FieldRef: Specifies a reference to an existing field definition that is used in the current list.

Field: Specifies either a definition of a new field definition to be used in this list, or a reference to an existing field that was deleted.

2.2.8.3.5 FieldDefinitionDatabaseWithVersion

The **FieldDefinitionDatabaseWithVersion** type provides an overall container for field definitions in **tp_Fields**, specified as follows.

2.2.8.3.5.1 Schema

```
    <xs:complexType name="FieldDefinitionDatabaseWithVersion" mixed="true">
      <xs:all>
        <xs:element name="tp_Fields" type="FieldDefinitionDatabase" minOccurs="1"
  maxOccurs="1"/>
```

```
    </xs:all>
  </xs:complexType>
```

2.2.8.3.5.2 Attributes

None.

2.2.8.3.5.3 Child Elements

tp_Fields: Specifies a collection of field definitions that are stored in a content database.

<Content>: The body text of this element MUST contain a string with the following pattern and MUST precede any child elements.

```
([0-9][0-9]*[.]) ([0-9][0-9]*[.]) ([0-9] [0-9]*[.]) ([0-9][0-9]*[.]) ([0-9][0-9]*[.]) ([0-
9][0-9]*)
```

The first two **groups** MUST correspond to the major version and minor version of the product. The second two groups MUST correspond to the revision version and build number of the product. The last two groups MUST correspond to the revision of the field definitions on the **list** and the list's template, respectively.

2.2.8.3.6 FieldDefinitionTP

The **FieldDefinitionTP** type specifies a field and its associated components. **FieldDefinitionTP** has the same structure as **FieldDefinition**. However, if only the **ID** attribute of **FieldDefinitionTP** is specified, the element specifies a field definition that was deleted by a user on a front-end Web Server.

2.2.8.3.6.1 Schema

```
  <xs:complexType name="FieldDefinitionTP">
    <xs:complexContent>
      <xs:extension base="FieldDefinition">
        <xs:attribute name="Type" type="xs:string" use="optional" />
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
```

2.2.8.3.6.2 Attributes

See **FieldDefinition** (section 2.2.8.3.3).

2.2.8.3.6.3 Child Elements

See **FieldDefinition** (section 2.2.8.3.3).

2.2.8.3.7 FieldParserRef

The **FieldParserRef** type allows schema authors to override the **Name** attribute of a **FieldDefinition** (section 2.2.8.3.3) type for a specific parser.

2.2.8.3.7.1 Schema

```
<xs:complexType name="FieldParserRef">
    <xs:attribute name="Name" type="xs:string" />
    <xs:attribute name="ProgId" type="xs:string" />
```

```
</xs:complexType>
```

2.2.8.3.7.2 Attributes

Name: Specifies an alternative **Name** to provide to the parser specified by the **ProgId** attribute when describing the field.

ProgId: Specifies the parser to which to provide the alternative **Name**.

2.2.8.3.7.3 Child Elements

None.

2.2.8.3.8 FieldParserRefs

The **FieldParserRefs** type allows schema authors to override the **Name** attribute of a **FieldDefinition** (section 2.2.8.3.3) type for a collection of parsers.

2.2.8.3.8.1 Schema

```
  <xs:complexType name="FieldParserRefs">
    <xs:sequence>
      <xs:element name="ParserRef" type="FieldParserRef" minOccurs="0" maxOccurs="unbounded"
/>
    </xs:sequence>
  </xs:complexType>
```

2.2.8.3.8.2 Attributes

None.

2.2.8.3.8.3 Child Elements

ParserRef: Specifies the overridden **Name** for a specific parser.

2.2.8.3.9 FieldRefDefinitionField

The **FieldRefDefinitionField** type specifies field definitions that are referenced within another field definition.

2.2.8.3.9.1 Schema

```
  <xs:complexType name="FieldRefDefinitionField" mixed="true" >
    <xs:attribute name="Name" type="xs:string" use="required" />
    <xs:attribute name="ID" type="UniqueIdentifierWithOrWithoutBraces" use="optional" />
    <xs:attribute name="ShowField" type="xs:string" use="optional" />
    <xs:attribute name="RefType" type="FieldRefType" use="optional" />
    <xs:attribute name="CreateURL" type="xs:string" use="optional" />
    <xs:attribute name="Key" type="xs:string" use="optional" />
    <xs:attribute name="DisplayName" type="xs:string" use="optional" />
    <xs:attribute name="Format" type="xs:string" use="optional" />
  </xs:complexType>
```

2.2.8.3.9.2 Attributes

See attributes section of **FieldDefinition** (section 2.2.8.3.3).

Name: Specifies the **Name** attribute of the referenced field.

ID: Specifies the **ID** attribute of the referenced fields. When both **ID** and **Name** are specified, the reader MUST use **ID** first and fall back to **Name** if the **ID** does not match.

ShowField: Specifies an alternate value for the **ShowField** attribute on a lookup field when rendering in the context of a computed field.

RefType: It describes the type of reference of the field in an events **list**. This MUST be a **FieldRefType** as specified in section 2.2.8.2.4. In other cases, this attribute MUST NOT be present.

CreateURL: The URL to create a **Meeting Workspace site**. If the **RefType** is LinkURL, this attribute MUST be present. Otherwise it MUST NOT be present.

Key: If this value of this attribute is set to "Primary", the server MUST give this field priority in the ordering of the items.

DisplayName: The reader MUST ignore this attribute.

Format: This attribute is only used if the referenced field is DateTime type. If the value of this attribute is set to "Original", the referenced field value will not get rendered.

2.2.8.3.9.3 Child Elements

<content>: Describes the Meeting Workspace site created by the URL in **CreateURL**. If the **RefType** attribute is present and has the value LinkURL, then the element MUST have content. In other cases, the reader MUST ignore any content.

2.2.8.3.10 FieldRefDefinitionIndex

The **FieldRefDefinitionIndex** type specifies a field to use in a composite field index.

2.2.8.3.10.1 Schema

```
<xs:complexType name="FieldRefDefinitionIndex">
  <xs:attribute name="ID" type="UniqueIdentifierWithBracesUppercase" use="required" />
</xs:complexType>
```

2.2.8.3.10.2 Attributes

ID: Identifies a field for which the server can optimize queries.

2.2.8.3.10.3 Child Elements

None.

2.2.8.3.11 FieldRefDefinitionTP

The **FieldRefDefinitionTP** type specifies a reference to a field definition. The attributes specified here override existing values in the field definition.

2.2.8.3.11.1 Schema

```
<xs:complexType name="FieldRefDefinitionTP">
    <xs:attribute name="Name" type="xs:string" use="required" />
    <xs:attribute name="ColName" type="xs:string" use="optional" />
    <xs:attribute name="ColName2" type="xs:string" use="optional" />
    <xs:attribute name="RowOrdinal" type="xs:int" default="0" use="optional" />
    <xs:attribute name="RowOrdinal2" type="xs:int" default="0" use="optional" />
    <xs:attribute name="ID" type="UniqueIdentifierWithOrWithoutBraces" use="optional" />
    <xs:attribute name="SourceID" type="xs:string" use="optional" />
```

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

2.2.8.3.11.2 Attributes

See **FieldDefinition** (section 2.2.8.3.3).

2.2.8.3.11.3 Child Elements

None.

2.2.8.3.12 IndexDefinitionTP

The **IndexDefinitionTP** type specifies a composite field index. A server can optimize for queries that filter on the fields specified within the composite field index.

2.2.8.3.12.1 Schema

```
<xs:complexType name="IndexDefinitionTP">
  <xs:sequence>
    <xs:element name="FieldRef" type="FieldRefDefinitionIndex" minOccurs="2" maxOccurs="2" />
    </xs:sequence>
  <xs:attribute name="ID" type="UniqueIdentifierWithBracesUppercase" use="required" />
</xs:complexType>
```

2.2.8.3.12.2 Attributes

ID: Identifier for the composite field index.

2.2.8.3.12.3 Child Elements

FieldRef: Specifies a reference to an existing field definition that is used in the current **list**.

2.2.8.3.13 MAPPINGDEFINITION

The **MAPPINGDEFINITION** type defines a canonical value for localizable **CHOICE** entries. Each **MAPPINGDEFINITION** MUST define in its contents a corresponding value from a choice.

2.2.8.3.13.1 Schema

```
<xs:complexType name="MAPPINGDEFINITION">
    <xs:simpleContent>

      <xs:extension base="xs:string">
        <xs:attribute name="Value" type="xs:string" />
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
```

2.2.8.3.13.2 Attributes

Value: String which contains a canonical, non-localizable value for a **CHOICE**.

2.2.8.3.13.3 Child Elements

<content>: Contains a string that MUST specify exactly the string of a corresponding **CHOICE**.

2.2.8.3.14 MAPPINGDEFINITIONS

The **MAPPINGDEFINITIONS** type is a container for one or more **MAPPINGS**. **MAPPINGS** MUST NOT be defined for fields other than **Choice** or **MultiChoice**. There MUST be either no **MAPPINGDEFINITION** elements defined for a **Choice** field, or exactly one **MAPPING** element for each corresponding **CHOICE** element.

2.2.8.3.14.1 Schema

```
<xs:complexType name="MAPPINGDEFINITIONS">
    <xs:sequence>
      <xs:element name="MAPPING" type="MAPPINGDEFINITION" minOccurs="0" maxOccurs="unbounded"
/>
    </xs:sequence>
</xs:complexType>
```

2.2.8.3.14.2 Attributes

None.

2.2.8.3.14.3 Child Elements

MAPPING: A canonical value mapping for a **CHOICE**.

2.2.8.4 Elements

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

2.2.8.5 Attributes

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

2.2.8.6 Groups

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

2.2.8.7 Attribute Groups

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

3 Protocol Details

3.1 Back-End Database Server Details

3.1.1 Abstract Data Model

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

Configuration Objects: A set of information about farm configuration. The Configuration Objects are stored in a Configuration Database.

Site Map: A set of information mapping URLs for site collections to site collection identifiers (section 2.2.1.9), and the content databases that contain each site collection's data. The Site Map is stored in the Configuration Database. Site Map entries are identified by site collection identifiers and also are represented by either absolute URLs or store relative form URLs.

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

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 URLs.

Sites: A set of information about all **sites** in a content database. Site entries are identified by site identifiers (section 2.2.1.11) and are also represented by store relative form URLs.

Lists: A set of information about all **lists** in a content database. List entries are identified by list identifiers (section 2.2.1.5) and are also represented by store relative form URLs.

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

Documents: A set of information about all documents in a content database. Document entries are identified by **document identifiers** (section 2.2.1.2) and are also are represented by Store Relative Form URLs.

Users: A set of information about all users in a content database. User entries are identified by user identifiers (section 2.2.1.13).

Site Groups: A set of information about all site **groups** in a content database. Site group entries are identified by site group identifiers (section 2.2.1.10).

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

3.1.2 Timers

An execution timeout timer is set up on the back-end database server to govern the execution time for any requests. The amount of time is governed by a time-out value configured on the back-end database server for all connections.

3.1.3 Initialization

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

The data structures, stored procedures, and actual data are persisted by the back-end database server within databases, so any operations to initialize the state of the database MUST occur before the back-end database server can use this protocol. This protocol requires that the data for site collections, **sites**, **lists**, and document libraries already exist within the back-end database server in a valid state.

3.1.4 Higher-Layer Triggered Events

None.

3.1.5 Message Processing Events and Sequencing Rules

The T-SQL syntax for each stored procedure and result set, and the variables they are composed of, is defined using the T-SQL language specified in [TSQL-Ref]. In the T-SQL syntax, the variable name is followed by the type of the variable, which can optionally have a length value in brackets and can optionally have a default value indicated by an equal sign followed by the default value. Unless otherwise specified, all stored procedures defined in this section are located in the Content Database.

For definitional clarity, a name has been assigned to any columns in the result sets that do not have a defined name in their current implementation. This does not affect the operation of the result set, because the front-end web server can access any column with no defined name by ordinal position. Such names are designated in the text using braces in the form {name}. For interoperability, named columns in result sets are specified with what they SHOULD be named, and columns marked with braces SHOULD have no defined name. front-end web server implementations MUST NOT rely on any column name in a result set.

The logical sequence of returned values and result sets are indicated in each of the individual stored procedures defined in this section. The TDS protocol controls the actual order and structure of how the T-SQL language formatted information is transported over the wire.

All functions, result sets, and stored procedures are defined using T-SQL.

3.1.5.1 fn_GetFullUrl

The **fn_GetFullUrl** function is invoked to construct a **full URL** from two component parts.

```
FUNCTION fn_GetFullUrl(
      @DirName                      nvarchar(256),
      @LeafName                     nvarchar(128)
)
RETURNS                             nvarchar(260)
```

@DirName: The directory name component of the full URL.

@LeafName: The leaf name component of the full URL.

Return Values: The full URL, which is formed from *@DirName* and *@LeafName* as **follows:**

If *@DirName* is an empty string, then *@LeafName* MUST be returned. If *@LeafName* is an empty string, then *@DirName* MUST be returned. If either *@DirName* or *@LeafName* is NULL, then NULL MUST be returned. Otherwise **fn_GetFullUrl** MUST return *@DirName* + '/' + *@LeafName*.

3.1.5.2 proc_AddBuildDependency

The **proc_AddBuildDependency** stored procedure is invoked to associate a build dependency with a specified document.

```
PROCEDURE proc_AddBuildDependency(
      @DocSiteId                    uniqueidentifier,
      @DocDirName                   nvarchar(256),
      @DocLeafName                  nvarchar(128),
      @TargetDirName                nvarchar(256),
      @TargetLeafName               nvarchar(128),
      @DirectDependency             bit,
      @RequestGuid                  uniqueidentifier  = NULL OUTPUT
);
```

@DocSiteId: The site collection identifier (section 2.2.1.9) for the site collection that contains the document. This parameter MUST NOT be NULL.

@DocDirName: The directory name of the document in store-relative form. This parameter MUST NOT be NULL.

@DocLeafName: The leaf name of the document. This parameter MUST NOT be NULL.

@TargetDirName: The directory name of the item to declare as a dependency of the document. This parameter MUST NOT be NULL.

@TargetLeafName: The leaf name of the item to declare as a dependency of the document. This parameter MUST NOT be NULL.

@DirectDependency: Set to 1 if the target is a direct dependency. Set to 0 if indirect, such as a dependency of a dependency. This parameter MUST NOT be NULL.

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

Return Values: The **proc_AddBuildDependency** stored procedure MUST return an integer return code of 0. The **proc_AddBuildDependency** stored procedure MUST NOT return a result set.

3.1.5.3 proc_AddDocument

The **proc_AddDocument** stored procedure is invoked to add a document to the back-end database server with the specified parameters.

```
PROCEDURE proc_AddDocument(
      @DocSiteId                    uniqueidentifier,
      @DocWebId                     uniqueidentifier,
      @UserId                       int,
      @AuthorId                     int,
      @DocDirName                   nvarchar(256),
      @DocLeafName                  nvarchar(128)     OUTPUT,
      @Level                        tinyint,
      @UIVersion                    int               = 512,
      @NewDocId                     uniqueidentifier,
      @DoclibId                     uniqueidentifier,
      @NewDoclibRowId               int,
      @SendingContent               bit,
      @DocMetaInfo                  varbinary(max),
      @DocSize                      int,
      @DocMetaInfoSize              int,
      @DocFileFormatMetaInfo        varbinary(max),
      @DocFileFormatMetaInfoSize    int,
      @EnableMinorVersions          bit,
      @IsModerated                  bit,
      @DocDirty                       bit,
      @DocFlags                       int,
      @DocIncomingCreatedDTM          datetime,
      @DocIncomingDTM                 datetime,
      @GetWebListForNormalization     bit,
      @PutFlags                       int,
      @CreateParentDir                bit,
      @UrlIsSuggestion                bit,
      @ThicketMainFile                bit,
      @CharSet                        int,
      @ProgId                         nvarchar(255),
      @AttachmentOp                   int,
      @VirusVendorID                  int,
      @VirusStatus                    int,
      @VirusInfo                      nvarchar(255),
      @LockTimeout                    int,
      @SharedLock                     bit,
      @Comment                        nvarchar(1023),
      @@DocDTM                        datetime           OUTPUT,
      @fNoQuotaOrLockCheck            bit,
      @DocContentVerBump              int,
      @DocClientId                    varbinary(16)      = NULL,
      @RequestGuid                    uniqueidentifier   = NULL OUTPUT
);
```

@DocSiteId: The site collection identifier (section 2.2.1.9) for the site collection that will contain the document to be stored. This MUST NOT be NULL.

@DocWebId: The site identifier (section 2.2.1.11) for the site that will contain the document to be stored. This MUST NOT be NULL.

@UserId: The user identifier (section 2.2.1.13) of the **current user** making the request to the front-end web server. This value MUST refer to an existing user identifier for the specified site collection.

@AuthorId: The user identifier to use instead of *@UserId*'s value in the owner fields of the document if the publishing level is set to draft or checked out, or the document has a **short-term lock** applied. If NULL is specified, then *@UserId* is used in the owner fields.

@DocDirName: The directory name of the document to be stored. This MUST NOT be NULL.

@DocLeafName: The leaf name of the document to be stored. If *@UrlIsSuggestion* is set to "1", then this name MUST be replaced with a unique name if it is not unique and returned in this output parameter as the actual document leaf name. This parameter MUST NOT be NULL.

@Level: The **Publishing Level** type (section 2.2.2.6) value for the document to be stored. This MUST be a valid value.

@UIVersion: The UI version number to associate with this document. This MUST NOT be NULL.

@NewDocId: The **document identifier** (section 2.2.1.2) of the document to be stored. This MUST NOT be NULL and MUST be unique for a new document, and it MUST be the same for an existing document adding a new publishing level.

@DoclibId: The list identifier (section 2.2.1.5) of the **list** or document library into which the document is to be stored. This MUST only be NULL when a document is not being added to a list or document library.

@NewDoclibRowId: The document library row identifier for the document to be stored. This MUST be NULL if the document is not in a list or document library.

@SendingContent: Specifies whether to store the document stream in the back-end database server. MUST be 1 if the document stream of the document is intended to be stored in the back-end database server; otherwise it MUST be 0.

@DocMetaInfo: The metadata information for the document to be stored. If there is no metadata information for this document, this parameter MUST be NULL.

@DocSize: Final size in bytes of the document stream of the document. This MUST be 0 if *@SendingContent* is 0.

@DocMetaInfoSize: Size in bytes of the document's metadata info. This MUST be 0 if *@DocMetaInfo* is NULL.

@DocFileFormatMetaInfo: The **DocFileFormatMetaInfo** data for the document to be stored. If there is no **DocFileFormatMetaInfo** data for this document, this parameter MUST be NULL.

@DocFileFormatMetaInfoSize: Size in bytes of the **DocFileFormatMetaInfo** data. This MUST be 0 if *@DocFileFormatMetaInfo* is NULL.

@EnableMinorVersions: Specifies whether minor versions are enabled on the document. If this parameter is set to "1", minor versions MUST be enabled; otherwise, minor versions MUST NOT be enabled.

@IsModerated: Specifies whether the list or document library into which the document will be stored has content approval enabled. MUST be 1 if the list or document library into which the document will be stored has content approval enabled; otherwise, it MUST be 0. If the document is not to be stored in a list or document library, then it MUST be 0;

@DocDirty: Specifies whether the document has dependencies such as links to other items. If this parameter is set to "1", the document has dependencies that MUST subsequently be updated.

@DocFlags: A **Doc Flags** (section 2.2.2.3) value that contains options for adding the document.

@DocIncomingCreatedDTM: A time stamp in **UTC** format that specifies when the document was created.

@DocIncomingDTM: A time stamp in UTC format that specifies the document's last modification date.

@GetWebListForNormalization: Specifies whether to return the Site List for Normalization Result Set (section 3.1.5.3.1). If this parameter is set to "1", **proc_AddDocument** MUST return a list of the immediate child subsites of the document's containing site in the Site List for Normalization Result Set. If this parameter is set to "0", the Site List for Normalization Result Set MUST NOT be returned. This parameter MUST NOT be NULL.

@PutFlags: A **Put Flags** type (section 2.2.2.7) value that specifies the options for adding the document.

@CreateParentDir: Specifies whether to create the parent directory for the document to be added if it does not already exist. If this parameter is set to "1", the parent directory specified by *@DocDirName* MUST be created if it does not exist. If this parameter is set to "0", **proc_AddDocument** MUST fail if the parent directory does not exist. This parameter MUST NOT be NULL.

@UrlIsSuggestion: Specifies whether the *@DocLeafName* provided MUST be changed to a unique value if it is not unique. If this parameter is set to "1", *@DocLeafName* MUST be updated to get a guaranteed unique URL. If this parameter is set to "0" and the URL is not unique, this procedure MUST fail. This parameter MUST NOT be NULL.

@ThicketMainFile: Specifies whether the document is a thicket main file. If this parameter is set to "1", then the document is a thicket main file. If this parameter is set to "0", then the document is not a thicket main file or a thicket supporting file. If this parameter is NULL, then this document is a thicket supporting file.

@CharSet: An optional parameter that specifies a **Windows code page** identifier for the **character set** to be associated with the document.

@ProgId: An optional parameter that specifies a preferred application to open the document. The *@ProgId* value is used to distinguish between different applications that save files with a given file extension (for example, different editing applications for HTML or XML files).

@AttachmentOp: An **Attachments Flag** (section 2.2.3.1) value that specifies the security checks to be performed by **proc_AddDocument** on this document's URL, based on whether it appears to be an attachment.

@VirusVendorID: An optional parameter that specifies the identifier of the **virus scanner** that processed this document.

@VirusStatus: A **Virus Status** (section 2.2.3.18) type that specifies the current virus check status of this document.

@VirusInfo: A string containing a provider-specific message returned by the virus scanner when it last processed the document. This value MUST be NULL if the document does not exist or if the document has not been processed by a virus scanner.

@LockTimeout: An integer value that specifies the number of minutes remaining to set on a short-term lock on the document. This value MUST be a positive value if a short-term lock is requested, and the **proc_AddDocument** stored procedure MUST set the short-term lock for this period. Otherwise, the **proc_AddDocument** stored procedure MUST NOT set a short-term lock on the document.

@SharedLock: Specifies whether to establish a shared lock on the document after it is added. This parameter MUST be set to 1 to establish a shared lock; otherwise it MUST be set to 0.

@Comment: An optional text check in comment to associate with the document. This value MUST be ignored when *@Level* is set to "255".

@@DocDTM: An output parameter for the time stamp of the last modification date of the document. This parameter MUST be set to the value of *@DocIncomingDTM* or to the current UTC datetime if *@DocIncomingDTM* is NULL.

@fNoQuotaOrLockCheck: Specifies whether to bypass the disk quota and disk lock check. If this parameter is set to "1", the checks are bypassed. If this parameter is set to "0", an explicit check will be made to see if the site is locked or if quota is reached.

@DocContentVerBump: An integer value used to increase the version of the content stored in the back-end database server. This value is added to the current content version number.

@DocClientId: An optional parameter that specifies the client ID of the document to be created.

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

Return Values: The **proc_AddDocument** stored procedure returns an integer return code, which MUST be one of the values listed in the following table.

Value	Description
0	Successful execution returned when all inputs are valid and all constraints are satisfied.
3	The path specified for the document (*@DocDirName*) was not found.
5	Access denied error while attempting to create a directory (*@DocDirName*).

Value	Description
80	Attempted to create a directory when the specified directory exists. The document (*@DocDirName*) already exists. The attempt to add the document failed. **SendingContent** was specified, but the document is not checked out. The attempt to add the document stream failed.
212	The disk storage was locked.
1816	Disk quota exceeded.

The **proc_AddDocument** stored procedure returns zero to two result sets in the order listed.

3.1.5.3.1 Site List for Normalization Result Set

The Site List for Normalization Result Set returns a list of URLs for the immediate child subsites of the site containing the newly added document. The Site List for Normalization Result Set MUST be produced when the input parameter *@GetWebListForNormalization* is set to "1" and execution has been successful up to the point of inserting the document. The Site List for Normalization Result Set MUST contain one row for each subsite found. The Site List for Normalization Result Set is defined in URL Result Set (section 2.2.5.25).

3.1.5.3.2 Checkout Information Result Set

The Checkout Information Result Set returns check-out information about the document. The Checkout Information Result Set MUST be returned on successful completion if either the input parameter *@Level* is set to "255", indicating that the document is checked out, or if the input parameter *@LockTimeout* is not NULL, indicating that a **short-term lock** was applied. The Checkout Information Result Set MUST contain one row.

```
tp_Login                        nvarchar(255),
CheckoutDate                    datetime,
{CheckoutExpires}               datetime;
```

tp_Login: The login name of the **principal** to whom the document is checked out. If the document is currently checked in, this MUST be NULL.

CheckoutDate: A time stamp, in **UTC** format, indicating when this document was checked out. If the document is currently checked in, this MUST be NULL.

{CheckoutExpires}: A time stamp, in UTC format, indicating when the short-term lock for this document will expire. If the document is currently checked in or has a long-term checkout, this MUST be NULL.

3.1.5.4 proc_AddListItem

The **proc_AddListItem** stored procedure is invoked to add a list item to a **list**.

```
PROCEDURE proc_AddListItem (
      @SiteId                        uniqueidentifier,
      @WebId                         uniqueidentifier,
      @ListID                        uniqueidentifier,
      @RowOrdinal                    tinyint,
      @ItemId                        int                            OUTPUT,
      @UserId                        int,
      @Size                          int,
      @TimeNow                      datetime,
      @CopySecurityFromMasterID     int                = NULL,
      @ExtraItemSize                int                = NULL,
      @CheckDiskQuota               bit                = 1,
      @ItemDocType                  tinyint            = 0,
      @SortTypeReversed             bit                = 0
      @BaseRowItemId                int                = NULL,
      @DocIdAdded                   uniqueidentifier   = NULL,
      @RetainId                     uniqueidentifier   = NULL,
      @RetainObjectIdentity         bit                = 0,
      @Level                        tinyint            = 1,
      @UIVersion                    int                = 512,
      @ItemCountDelta               int                = 1,
      @ItemName                     nvarchar(255)      = NULL,
      @UseNvarchar1ItemName         bit                = 1,
      @ItemDirName                  nvarchar(256)      = NULL      OUTPUT,
      @ItemLeafName                 nvarchar(128)      = NULL      OUTPUT,
      @ServerTemplate               int                = NULL,
      @IsNotUserDisplayed           bit                = NULL,
      @BaseType                     int                = NULL,
      @CheckSchemaVersion           int                = NULL,
      @OnRestore                    bit                = 0,
      @AddNamespace                 bit                = 0,
      @tp_Ordering                  varchar(512)       = NULL,
      @tp_ThreadIndex               varbinary(512)     = NULL,
      @tp_HasAttachment             bit                = NULL,
      @tp_ModerationStatus          int                = 0,
      @tp_IsCurrent                 bit                = 1,
      @tp_ItemOrder                 float              = NULL,
      @tp_InstanceID                int                = NULL,
      @tp_GUID                      uniqueidentifier   = NULL,
      @tp_Id                        int                = NULL,
      @tp_Author                    int                = NULL,
      @tp_Editor                    int                = NULL,
      @tp_Modified                  datetime           = NULL,
      @tp_Created                   datetime           = NULL,
      @tp_Version                   int                = 1,

      @tp_ContentTypeId             varbinary(512)     = NULL,
      @tp_CopySource                nvarchar(260)      = NULL,
      @tp_HasCopyDestinations       bit                = NULL,
      @tp_WorkflowVersion           int                = 1,
      @tp_WorkflowInstanceID        uniqueidentifier   = NULL,
      @nvarchar1                    nvarchar(255)      = NULL,
      @nvarchar2                    nvarchar(255)      = NULL,
      @nvarchar3                    nvarchar(255)      = NULL,
      @nvarchar4                    nvarchar(255)      = NULL,
      @nvarchar5                    nvarchar(255)      = NULL,
      @nvarchar6                    nvarchar(255)      = NULL,
      @nvarchar7                    nvarchar(255)      = NULL,
      @nvarchar8                    nvarchar(255)      = NULL,
      @nvarchar9                    nvarchar(255)      = NULL,
      @nvarchar10                   nvarchar(255)      = NULL,
      @nvarchar11                   nvarchar(255)      = NULL,
      @nvarchar12                   nvarchar(255)      = NULL,
      @nvarchar13                   nvarchar(255)      = NULL,
      @nvarchar14                   nvarchar(255)      = NULL,
      @nvarchar15                   nvarchar(255)      = NULL,
      @nvarchar16                   nvarchar(255)      = NULL,
      @nvarchar17                   nvarchar(255)      = NULL,
      @nvarchar18                   nvarchar(255)      = NULL,
      @nvarchar19                   nvarchar(255)      = NULL,
      @nvarchar20                   nvarchar(255)      = NULL,
      @nvarchar21                   nvarchar(255)      = NULL,
      @nvarchar22                   nvarchar(255)      = NULL,
      @nvarchar23                   nvarchar(255)      = NULL,
      @nvarchar24                   nvarchar(255)      = NULL,
      @nvarchar25                   nvarchar(255)      = NULL,
      @nvarchar26                   nvarchar(255)      = NULL,
      @nvarchar27                    nvarchar(255)      = NULL,
      @nvarchar28                    nvarchar(255)      = NULL,
      @nvarchar29                    nvarchar(255)      = NULL,
      @nvarchar30                    nvarchar(255)      = NULL,
      @nvarchar31                    nvarchar(255)      = NULL,
      @nvarchar32                    nvarchar(255)      = NULL,
      @nvarchar33                    nvarchar(255)      = NULL,
      @nvarchar34                    nvarchar(255)      = NULL,
      @nvarchar35                    nvarchar(255)      = NULL,
      @nvarchar36                    nvarchar(255)      = NULL,
      @nvarchar37                    nvarchar(255)      = NULL,
      @nvarchar38                    nvarchar(255)      = NULL,
      @nvarchar39                    nvarchar(255)      = NULL,
      @nvarchar40                    nvarchar(255)      = NULL,
      @nvarchar41                    nvarchar(255)      = NULL,
      @nvarchar42                    nvarchar(255)      = NULL,
      @nvarchar43                    nvarchar(255)      = NULL,
      @nvarchar44                    nvarchar(255)      = NULL,
      @nvarchar45                    nvarchar(255)      = NULL,
      @nvarchar46                    nvarchar(255)      = NULL,
      @nvarchar47                    nvarchar(255)      = NULL,
      @nvarchar48                    nvarchar(255)      = NULL,
      @nvarchar49                    nvarchar(255)      = NULL,
      @nvarchar50                    nvarchar(255)      = NULL,
      @nvarchar51                    nvarchar(255)      = NULL,
      @nvarchar52                    nvarchar(255)      = NULL,
      @nvarchar53                    nvarchar(255)      = NULL,
      @nvarchar54                    nvarchar(255)      = NULL,
      @nvarchar55                    nvarchar(255)      = NULL,
      @nvarchar56                    nvarchar(255)      = NULL,
      @nvarchar57                    nvarchar(255)      = NULL,
      @nvarchar58                    nvarchar(255)      = NULL,
      @nvarchar59                    nvarchar(255)      = NULL,
      @nvarchar60                    nvarchar(255)      = NULL,
      @nvarchar61                    nvarchar(255)      = NULL,
      @nvarchar62                    nvarchar(255)      = NULL,
      @nvarchar63                    nvarchar(255)      = NULL,
      @nvarchar64                    nvarchar(255)      = NULL,
      @int1                          int                = NULL,
      @int2                          int                = NULL,
      @int3                          int                = NULL,
      @int4                          int                = NULL,
      @int5                          int                = NULL,
      @int6                          int                = NULL,
      @int7                          int                = NULL,
      @int8                          int                = NULL,
      @int9                          int                = NULL,
      @int10                         int                = NULL,
      @int11                         int                = NULL,
      @int12                         int                = NULL,
      @int13                         int                = NULL,
      @int14                         int                = NULL,
      @int15                         int                = NULL,
      @int16                         int                = NULL,
      @float1                        float              = NULL,
      @float2                        float              = NULL,
      @float3                        float              = NULL,
      @float4                        float              = NULL,
      @float5                        float              = NULL,
      @float6                        float              = NULL,
      @float7                        float              = NULL,
      @float8                        float              = NULL,
      @float9                        float              = NULL,
      @float10                       float              = NULL,
      @float11                       float              = NULL,
      @float12                       float              = NULL,
      @datetime1                     datetime           = NULL,
      @datetime2                     datetime           = NULL,
      @datetime3                     datetime           = NULL,
      @datetime4                      datetime            = NULL,
      @datetime5                      datetime            = NULL,
      @datetime6                      datetime            = NULL,
      @datetime7                      datetime            = NULL,
      @datetime8                      datetime            = NULL,
      @bit1                           bit                 = NULL,
      @bit2                           bit                 = NULL,
      @bit3                           bit                 = NULL,
      @bit4                           bit                 = NULL,
      @bit5                           bit                 = NULL,
      @bit6                           bit                 = NULL,
      @bit7                           bit                 = NULL,
      @bit8                           bit                 = NULL,
      @bit9                           bit                 = NULL,
      @bit10                          bit                 = NULL,
      @bit11                          bit                 = NULL,
      @bit12                          bit                 = NULL,
      @bit13                          bit                 = NULL,
      @bit14                          bit                 = NULL,
      @bit15                          bit                 = NULL,
      @bit16                          bit                 = NULL,
      @uniqueidentifier1              uniqueidentifier    = NULL,
      @ntext1                         nvarchar(max)       = NULL,
      @ntext2                         nvarchar(max)       = NULL,
      @ntext3                         nvarchar(max)       = NULL,
      @ntext4                         nvarchar(max)       = NULL,
      @ntext5                         nvarchar(max)       = NULL,
      @ntext6                         nvarchar(max)       = NULL,
      @ntext7                         nvarchar(max)       = NULL,
      @ntext8                         nvarchar(max)       = NULL,
      @ntext9                         nvarchar(max)       = NULL,
      @ntext10                        nvarchar(max)       = NULL,
      @ntext11                        nvarchar(max)       = NULL,
      @ntext12                        nvarchar(max)       = NULL,
      @ntext13                        nvarchar(max)       = NULL,
      @ntext14                        nvarchar(max)       = NULL,
      @ntext15                        nvarchar(max)       = NULL,
      @ntext16                        nvarchar(max)       = NULL,
      @ntext17                        nvarchar(max)       = NULL,
      @ntext18                        nvarchar(max)       = NULL,
      @ntext19                        nvarchar(max)       = NULL,
      @ntext20                        nvarchar(max)       = NULL,
      @ntext21                        nvarchar(max)       = NULL,
      @ntext22                        nvarchar(max)       = NULL,
      @ntext23                        nvarchar(max)       = NULL,
      @ntext24                        nvarchar(max)       = NULL,
      @ntext25                        nvarchar(max)       = NULL,
      @ntext26                        nvarchar(max)       = NULL,
      @ntext27                        nvarchar(max)       = NULL,
      @ntext28                        nvarchar(max)       = NULL,
      @ntext29                        nvarchar(max)       = NULL,
      @ntext30                        nvarchar(max)       = NULL,
      @ntext31                        nvarchar(max)       = NULL,
      @ntext32                        nvarchar(max)       = NULL,
      @sql_variant1                   sql_variant         = NULL,
      @error_sql_variant1             int                 = 0,
      @sql_variant2                   sql_variant         = NULL,
      @error_sql_variant2             int                 = 0,
      @sql_variant3                   sql_variant         = NULL,
      @error_sql_variant3             int                 = 0,
      @sql_variant4                   sql_variant         = NULL,
      @error_sql_variant4             int                 = 0,
      @sql_variant5                   sql_variant         = NULL,
      @error_sql_variant5             int                 = 0,
      @sql_variant6                   sql_variant         = NULL,
      @error_sql_variant6             int                 = 0,
      @sql_variant7                   sql_variant         = NULL,
      @error_sql_variant7             int                 = 0,
      @sql_variant8                   sql_variant         = NULL,
      @error_sql_variant8            int               = 0,
      @eventData                     varbinary(max)    = NULL,
      @acl                           varbinary(max)    = NULL,
      @DocClientId                   varbinary(16)     = NULL,
      @RequestGuid uniqueidentifier                    = NULL     OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) for the site collection containing the list that the list item is being added to.

@WebId: The site identifier (section 2.2.1.11) for the site containing the list that the list item is being added to.

@ListID: The list identifier (section 2.2.1.5) of the list that the list item is being added to.

@RowOrdinal: The 0-based ordinal index of the current row to add for this list item in the set of rows representing the list item in the **AllUserData** table (section 2.2.7.3). If a list item requires multiple rows to represent it in the **AllUserData** table because it contains more defined data columns than will fit in a single row, the front-end web server MUST call **proc_AddListItem** again, with the additional data using the next row value in the *@RowOrdinal* parameter. This parameter MUST NOT be NULL.

@ItemId: An output parameter that returns the identifier of the list item that has been **added:**

- If *@tp_Id* is not NULL and *@BaseRowItemId* is NULL, **proc_AddListItem** MUST use the value specified by *@tp_Id*.
- If *@BaseRowItemId* is not NULL, **proc_AddListItem** MUST use the value specified by *@BaseRowItemId*.
- If *@tp_Id* is NULL and *@BaseRowItemId* is NULL, **proc_AddListItem** MUST generate a new value for the list item identifier (section 2.2.1.6) that is unique within the list.

@UserId: The user identifier (section 2.2.1.13) for the current user. The **proc_AddListItem** stored procedure uses this for purposes of permission-checking. This value MUST refer to an existing user identifier for the specified site collection.

@Size: The size in bytes of the list item row to be added. This parameter MUST NOT be NULL.

@TimeNow: The current time, in **UTC** format, on the back-end database server.

@CopySecurityFromMasterID: Specifies the optional identifier of the list item to copy the security scope settings from for this list item. A list item that represents an exception to a recurrence item in a Meetings List (a list with a **List Server Template** type (section 2.2.3.12) of 200) MUST have the same security scope settings as the master recurrence item. If this parameter is set to a master recurrence item's list item identifier, then if the list item to be added does not share the security scope of the master recurrence item, the new list item MUST be set to have a unique security scope with a copy of the security settings from the security scope of the master recurrence item. This parameter MUST only be set for list items that are exceptions to recurrence items in a Meetings List.

@ExtraItemSize: The size of the predefined SQL parameter fields in the list item row being added.

@CheckDiskQuota: A bit flag specifying whether or not disk quota SHOULD be checked for the current user before adding the list item. The bit MUST be set to 1 for disk quota to be checked for the current user before adding the list item; otherwise, the bit MUST be set to 0.

@ItemDocType: The **Document Store** type (section 2.2.2.4) of the list item being added to the list.

@SortTypeReversed: Specifies whether or not the list item SHOULD sort using the behavior of folder or file types. The bit MUST be set to "1" for folders to sort like files; otherwise it MUST be set to "0". It MUST be set to "0" for all file type list items.

@BaseRowItemId: An optional value that specifies the list item identifier to be added if a value is not supplied by the *@tp_Id* parameter. If the *@tp_Id* parameter is NULL and this parameter is not NULL, then **proc_AddListItem** MUST use this value for the list item identifier to be added.

@DocIdAdded: The optional **document identifier** (section 2.2.1.2) of the document to be inserted as a list item in the list. If *@DocIdAdded* is not NULL and the document has a document identifier, this parameter MUST be the existing document identifier. If this parameter is NULL, then a new document identifier MUST be generated for the list item. If this list item is being restored to the list as part of an implementation-specific backup restore operation, as specified by the value of *@RetainObjectIdentity*, then this parameter MUST be ignored, and the value of the *@RetainId* parameter MUST be used as the document identifier.

@RetainId: The document identifier of the document to be inserted as a list item in the list, if this list item is being restored to the list as part of a back-up restore operation.

@RetainObjectIdentity: A bit flag specifying whether this list item is being restored to the list as part of an implementation-specific backup restore operation. If *@RetainObjectIdentity* is set to 1, this list item is being restored to the list as part of an implementation-specific backup restore operation and MUST have the *@DocIdAdded* value specified in *@RetainId*.

@Level: The **Publishing Level** type (section 2.2.2.6) value specifying the publishing level of this list item.

@UIVersion: The UI version number for the list item.

@ItemCountDelta: The number to be added to the list item count of the containing list:

- For a list item added with a single call to **proc_AddListItem**, or for one call for a list item with multiple rows to be added, this value MUST be 1.
- For the other calls to **proc_AddListItem** for the additional rows to be added for the list item, this value MUST be 0.

@ItemName: The display name of the list item.

@UseNvarchar1ItemName: If *@ItemName* is NULL, this bit flag specifies whether to use the content of *@nvarchar1* for the display name of the list item.

@ItemDirName: An output parameter containing the directory name of the list item.

@ItemLeafName: An output parameter containing the leaf name of the list item.

@ServerTemplate: The identifier for the **List Server Template** defining the base structure of the list containing this list item.

@IsNotUserDisplayed: A bit flag specifying whether the **user name** is not displayed with list items.

@BaseType: The **List Base** type (section 2.2.3.11) of the list containing the list item.

@CheckSchemaVersion: This specifies an optional schema version number to compare with the list schema version number. If this parameter is not NULL, the version numbers MUST match for successful completion.

@OnRestore: A bit flag that specifies whether this list item is being inserted by an implementation-specific back-up restore operation.

@AddNamespace: A Boolean value specifying whether metadata is being added to the list item. This parameter MUST NOT be NULL.

@tp_Ordering: This parameter specifies the threading structure for this list item in a deprecated discussion board list (a list with a **List Base** type of 3) as a concatenation of UTC date/time stamp values in yyyyMMddHHmmss format. For all list items in lists with other **List Base** types, this parameter MUST be NULL.

@tp_ThreadIndex: This specifies the list item's position within a threaded discussion board list (a list with a **List Base** type of 3) as a binary structure. For all list items in lists with other **List Base** types, this parameter MUST be NULL.

@tp_HasAttachment: A bit flag that specifies whether the list item has an associated attachment.

@tp_ModerationStatus: A **Moderation Status** (section 2.2.3.13) value specifying the current **moderation status** of this list item.

@tp_IsCurrent: A bit flag that specifies whether or not this is the current version of this publishing level of the list item.

@tp_ItemOrder: This specifies the implementation-specific order in which the list item SHOULD be displayed with other list items from the same list. This value can be the same as other list items in the list.

@tp_InstanceID: If this list item is associated with a particular **meeting instance** of a recurring meeting, this specifies the ID of that instance. For all other list items, this parameter MUST be NULL.

@tp_GUID: A GUID that uniquely identifies this list item within the **AllUserData** table.

@tp_Id: The optional list item identifier specified for this list item. If this parameter is not NULL, **proc_AddListItem** MUST use this value for the identifier of the list item to be added.

@tp_Author: The user identifier (section 2.2.1.13) for the user who created the list item.

@tp_Editor: The user identifier for the user who last edited the list item.

@tp_Modified: A time stamp, in UTC format, that specifies when this list item was last modified.

@tp_Created: A time stamp, in UTC format, that specifies when this list item was created.

@tp_Version: Specifies the internal version number used for internal conflict detection.

@tp_ContentTypeId: Specifies the content type identifier for this list item.

@tp_CopySource: Specifies the URL used as a source for this list item. If this list item was not copied from a source list item, then this value MUST be NULL.

@tp_HasCopyDestinations: A bit flag specifying whether destination locations have been set for this list item to be copied to. If this list item does not have a destination location set, then this value MUST be 0.

@tp_WorkflowVersion: If this list item is part of a **workflow**, this parameter specifies the value to set denoting the state of this list item within that workflow. If this list item is not part of a workflow, then this MUST be NULL.

@tp_WorkflowInstanceID: A workflow identifier (section 2.2.1.16) that specifies the currently active **workflow instance** on this list item. If this list item is not part of a workflow, then this MUST be NULL.

The next nine columns are duplicated a variable number of times, depending on the list item's content type within the view definition, with each column referring to a separate **List Server Template**-defined field or user-defined field within the containing list. Each instance of these individual column names is differentiated by a suffix with a value indicated in the column description, which replaces the placeholder "#" symbol shown as follows.

@nvarchar#: User-defined columns in the list containing values of type **nvarchar**. There are 64 columns numbered from 1 to 64. If the column contains no data, then the value MUST be NULL.

@int#: User-defined columns in the list containing values of type **int**. There are 16 columns numbered from 1 to 16. If the column contains no data, then the value MUST be NULL.

@float#: User-defined columns in the list containing values of type **float**. There are 12 columns numbered from 1 to 12. If the column contains no data, then the value MUST be NULL.

@datetime#: User-defined columns in the list containing values of type **datetime**. There are eight columns numbered from 1 to 8. If the column contains no data, then the value MUST be NULL.

@bit#: User-defined columns in the list containing values of type **bit**. There are 16 columns numbered from 1 to 16. If the column contains no data, then the value MUST be NULL.

@uniqueidentifier1: A user-defined column in the list containing values of type **uniqueidentifier**. If the column contains no data, then the value MUST be NULL.

@ntext#: User-defined columns in the list containing values of type **nvarchar(max)**. There are 32 columns numbered from 1 to 32. If the column contains no data, then the value MUST be NULL.

@sql_variant#: User-defined columns in the list containing values of type **sql_variant**. There are eight columns numbered from 1 to 8. If the column contains no data, then the value MUST be NULL.

@error_sql_variant#: An **integer** specifying the type to be applied to the corresponding values specified as arguments for the parameter *@sql_variant#*. The possible types are listed in the following table.

Value	Description
1	Convert the argument value to a **varbinary(2)**.
2	Convert the argument value to a **bit**.
3	Convert the argument value to a **float**.
4	Convert the argument value to a **datetime**.

@eventData: Contains implementation-specific **event (2)** data significant to the front-end web server, but otherwise opaque to the back-end database server, to be stored by the BEDS for eventual writing to a log file.

@acl: The binary serialization of the **WSS ACL Format** (section 2.2.4.6) ACL for the data supplied in *@eventData*, to be stored with the data.

@DocClientId: An optional parameter that specifies the client ID of the list item to be saved. This can be NULL.

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

Return Values: The **proc_AddListItem** stored procedure returns an integer return code, which MUST be listed in the following table.

Value	Description
0	Successful execution.
2	Postprocessing of the list item failed because a prerequisite list item was not found.
3	The directory specified for the list item does not exist.

Value	Description
5	The attempt to create a directory or document failed because the user does not have sufficient permissions.
13	The list item to be added is not valid.
16	Adding the list item caused updating of an existing list item to fail.
33	Cannot move directories that contain checked-out files.
50	A list item could not be deleted.
80	The document being added to the list already exists.
87	Unable to add the list item because the input parameters do not match existing list items, or an error occurred during a table update operation.
138	The list could not be moved to the specified location.
160	Could not create a unique filename.
161	A directory that spans sites cannot be moved.
183	The list item being added already exists in the list.
206	The file or directory name is too long.
212	The database for the site collection is locked.
1150	Failed to update the list.
1359	An internal error occurred while moving a list item.
1638	The current schema version of the list does not match the value in *@CheckSchemaVersion*.
1816	The site collection is over its allocated size quota.
8398	A directory could not be deleted.

The **proc_AddListItem** stored procedure MUST return no result sets.

3.1.5.5 proc_ChangeLevelForDoc

The **proc_ChangeLevelForDoc** stored procedure is invoked to update the publishing level of a document. Depending on the settings in effect on the containing **list** and the permissions of the specified user making the request, **proc_ChangeLevelForDoc** also can update the document's **Moderation Status** (section 2.2.3.13), UI version number, and properties to reflect the change.

```
PROCEDURE proc_ChangeLevelForDoc(
      @SiteId                        uniqueidentifier,
      @DirName                       nvarchar(256),
      @LeafName                      nvarchar(128),
      @Level                         tinyint                    OUTPUT,
      @NewLevel                      tinyint,
      @ModerationStatus              int,
      @EnableMinorVersions           bit,
      @Moderated                     bit,
      @CreateVersion                 bit,
      @UserId                        int,
      @Comment                       nvarchar(1023),
      @bUpdateModified               bit,
      @@DoclibRowId                  int                        OUTPUT,
      @@DocUIVersion                 int                        OUTPUT,
      @@DocFlagsOut                   int                         OUTPUT,
      @@DocVersionOut                 int                         OUTPUT,
      @RequestGuid                    uniqueidentifier    = NULL  OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the document.

@DirName: The directory name of the document.

@LeafName: The leaf name of the document.

@Level: A **Publishing Level** type (section 2.2.2.6) value indicating the document's current publishing **level**. This output parameter also reflects the new publishing level value for the document.

@NewLevel: A **Publishing Level** type value indicating the requested publishing level for the document.

If *@Level* is set to 255 (Checked Out) or 2 (Draft), and *@NewLevel* is set to 1 (Published), then the document's level value MUST be changed to 1 (Published). If *@Moderated* is 0, *@EnableMinorVersions* is 0, *@Level* is not set to 255 (Checked Out) and *@NewLevel* is not set to 1 (Published), then the document's level value MUST NOT change, and output parameters MUST NOT be changed. If *@Level* is 1 (Published) and *@NewLevel* is 2 (Draft), then the document's level value MUST be changed to 2 (Draft). If *@Level* is 255 (Checked Out) and the *@NewLevel* is 1 (Published), the document's **CheckOutUserId**, **CheckoutDate**, and **CheckoutExpires** values MUST be set to NULL, and the document's container (document library or list) value for **tp_CheckOutUserId** MUST be set to NULL.

@ModerationStatus: An integer value indicating the specified document's **Moderation Status** (section 2.2.3.13).

@EnableMinorVersions: A bit indicating whether minor versions are enabled within the containing list. If the list containing the specified document has minor versions enabled, then this MUST be set to 1; otherwise, this MUST be set to 0.

@Moderated: A bit indicating whether moderation is enabled on the containing list. If the list containing the specified document has moderation enabled, then this MUST be set to 1; otherwise, this MUST be set to 0.

@CreateVersion: A bit indicating whether versioning is enabled on the containing list. If the list containing the specified document has versioning enabled, then this MUST be set to 1; otherwise, this MUST be set to 0.

@UserId: A user identifier (section 2.2.1.13) for the user requesting the document change. This value MUST refer to an existing user identifier for the specified site collection.

@Comment: Descriptive text associated with the document on check in or publishing. When the document publishing level is updated from "Check Out" (255) to any other value, or from any value to "Publish" (1), *@Comment* can be set to a descriptive string or it can be NULL. In all other **Publishing Level** type value transitions, this parameter MUST be NULL.

@bUpdateModified: A bit indicating if change to the last modified date and time properties of the document is requested. If *@bUpdateModified* is set to 1, then **proc_ChangeLevelForDoc** MUST update document last modified date and time properties in the store to note that the document was updated at the current time. Otherwise, the last modified date and time properties MUST be left unchanged.

@@DoclibRowId: An output parameter containing the value of the **DoclibRowId** column in the **Docs View** (section 2.2.7.4) for the document after the change.

@@DocUIVersion: An output parameter containing the UI version of the document after the change.

@@DocFlagsOut: An output parameter containing the **Doc Flags** (section 2.2.2.3) of the document after the change.

@@DocVersionOut: An output parameter containing the internal version number of the document after the change.

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

Return Values: The **proc_ChangeLevelForDoc** stored procedure returns an integer return code, which MUST be included in the following table.

Value	Description
0	Successful execution.
87	The specified document could not be found, or the specified publishing level is not the current level of the specified document.

The **proc_ChangeLevelForDoc** stored procedure MUST NOT return a result set.

3.1.5.6 proc_CheckRbsInstalled

The **proc_CheckRbsIntstalled** stored procedure is invoked to test whether the back-end database server can support remote blob storage.

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

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

Return Values: The **proc_CheckRbsIntstalled** stored procedure returns an integer return code, which MUST be in the following table.

Value	Description
0	The back-end database server does not support remote blob storage.
1	The back-end database server supports remote blob storage.

The **proc_CheckRbsIntstalled** stored procedure MUST NOT return any result sets.

3.1.5.7 proc_CheckoutDocument

The **proc_CheckoutDocument** stored procedure is invoked to place a **short-term lock** on a document, refresh, convert or release an existing short-term lock, or to check out a document. Short term locks are either of type shared or exclusive.

```
PROCEDURE proc_CheckoutDocument(
      @SiteId                        uniqueidentifier,
      @WebId                         uniqueidentifier,
      @DirName                       nvarchar(256),
      @LeafName                      nvarchar(128),
      @Level                         tinyint,
      @EnableMinorVersions           bit,
      @IsModerated                   bit,
      @UserId                        int,
      @CheckoutTimeout               int,
      @RefreshLock                   bit,
      @CheckoutToLocal              bit,
      @IsForceCheckout              bit,
      @IsSharedLock                 bit,
      @IsConvertLock                bit,
      @DocMetaInfo                  varbinary(max),
      @DocMetaInfoSize              int,
      @DocMetaInfoVersion           int,
      @GetLinkInfo                  bit,
      @RequestGuid                  uniqueidentifier   = NULL   OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) for the site collection containing the document to be checked out, locked, or unlocked.

@WebId: The site identifier (section 2.2.1.11) for the site containing the document.

@DirName: The directory name of the document.

@LeafName: The leaf name of the document.

@Level: A **Publishing Level** type (section 2.2.2.6) value specifying the publishing status of the document.

@EnableMinorVersions: A bit flag specifying whether the document library or list containing the document has minor version numbering enabled. If this parameter is set to "1", then minor version numbering is enabled for the document library or list; otherwise, this MUST be set to 0.

@IsModerated: A bit flag specifying whether moderation is in effect on the document. If this parameter is set to "1", then moderation is in effect on this document, which is used to implement an approval process to set the publishing level to published after the document is created or modified. This parameter MUST NOT be NULL.

@UserId: The user identifier (section 2.2.1.13) for the current user who is requesting a short-term lock or checking out the document. This value MUST refer to an existing user identifier for the specified site collection.

@CheckoutTimeout: Specifies the remaining time in minutes that short-term lock will be in effect for the document. A value of 0 means that the existing short-term lock MUST be released. The *@CheckoutTimeout* parameter MUST be NULL if the document is being checked out instead of having a short-term lock applied.

@RefreshLock: A bit flag specifying whether the short-term lock on the document is being refreshed. If this parameter is set to "1", then the existing short-term lock on the document MUST be refreshed for the number of minutes specified by the *@CheckoutTimeout* parameter. This parameter MUST be set to "0" to check out the document or to request a new short-term lock. This parameter MUST NOT be NULL.

@CheckoutToLocal: A bit flag specifying whether the document is to be copied to local storage on the user's client for editing. If this parameter is set to "1", then the user's client MUST make a local copy of the document stream for editing, and **proc_CheckoutDocument** MUST NOT make a checked-out version of the document in the store. This parameter MUST NOT be NULL.

@IsForceCheckout: A bit flag specifying whether the containing document library requires documents to be checked out before any changes can be made. If this parameter is set to "1", then the containing document library has "Require Check Out" turned on. This parameter MUST NOT be NULL.

@IsSharedLock: A bit flag specifying whether the desired short term lock on the document is a shared lock or an exclusive lock. This parameter MUST be set to "1" if the desired short-term lock is type shared; otherwise it MUST be set to "0".

@IsConvertLock: A bit flag specifying whether to convert an existing short-term lock from one short-term lock type to a different short-term lock type. This parameter MUST be set to "1" to convert the type of an existing short-term lock; otherwise it MUST be set to "0".

@DocMetaInfo: The metadata information for the document to be checked out. If there is no metadata information for this document, then this parameter MUST be NULL.

@DocMetaInfoSize: Size in bytes of the document's metadata info. This MUST be NULL if *@DocMetaInfo* is NULL.

@DocMetaInfoVersion: The version of the metadata information for the document to be checked out. This MUST be NULL if *@DocMetaInfo* is NULL.

@GetLinkInfo: A bit flag specifying whether the desired Link Info Single Doc Result Set (specified in section 3.1.5.7.1) is returned. This parameter MUST be set to "1" to return the Link Info Single Doc Result Set; otherwise it MUST be set to "0".

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

Return Values: The **proc_CheckoutDocument** stored procedure returns an integer return code, which MUST be in the following table.

Value	Description
0	Successful execution.
3	File not found. A document corresponding to the specified *@SiteId*, *@WebId*, *@DirName*, *@LeafName*, and *@Level* parameters was not found.
33	Short-term lock error. The document cannot have a short-term lock applied because another user has the document checked out.
154	Invalid minor version value. The minor version value for the document would exceed the maximum allowed value (511) if checked out.
158	Checkout required. The document is in a document library with the "Require Check Out" option set, but the document is not being checked out.
173	Checkout error. The document cannot be checked out, because it is already checked out to or locked by another user.
212	Site collection locked. The site collection is in disk write lock.
1630	Unsupported document type. The document specified is not valid for check out; folders and sites cannot be checked out.
1816	Disk quota error. The site collection disk quota has been reached.
6002	Short-term lock error. The document cannot have a short-term lock applied because another user has a shared short-term lock on the file.
6009	Short-term lock error. The document cannot have a short-term lock applied because another user has an exclusive short-term lock on the file.

The **proc_CheckoutDocument** stored procedure MUST return multiple result sets. Some of the result sets are returned 0 or more times depending upon conditions specified in the following sections, and all result sets that are returned will be sent in the order specified in sections 3.1.5.7.1 through 3.1.5.7.4.

3.1.5.7.1 Link Info Single Doc Result Set

The Link Info Single Doc Result Set returns information about all forward links and backward links associated with the document. The Link Info Single Doc Result Set MUST be returned once and MUST hold one row for each forward and backward link associated with the specified document at the specified publishing level.

The Link Info Single Doc Result Set is defined in Link Information Result Set, section 2.2.5.11.

3.1.5.7.2 Document Metadata Result Set

The Document Metadata Result Set returns the metadata for the document. The Document Metadata Result Set MUST be returned and MUST contain a single row corresponding to the checked-out or locked document.

The Document Metadata Result Set is defined in Document Metadata Result Set, section 2.2.5.6.

3.1.5.7.3 Event Receivers Result Set

The Event Receivers Result Set contains information about the list item **event receivers** defined for this document.

The Event Receivers Result Set MUST only be returned if the requested document matching the *@Level* parameter exists within the site specified by the *@WebId* parameter for the current user. The Event Receivers Result Set MUST contain one row for each event receiver registered with an **Event Host** type (section 2.2.3.5) of 3 (list item) for this document.

The Event Receivers Result Set is defined in Event Receivers Result Set, section 2.2.5.9.

3.1.5.7.4 Audit Mask Result Set

The Audit Mask Result Set returns audit configuration information for the document. The Audit Mask Result Set MUST be returned with one row of audit configuration information on successful execution.

The Audit Mask Result Set is defined in Site Audit Mask Result Set, section 2.2.5.20.

3.1.5.8 proc_ClearLinks

The **proc_ClearLinks** stored procedure deletes all link information associated with a particular field in a list item as part of a list item or document update. All link information that matches all of the parameters and does not have associated Web parts MUST be deleted by **proc_ClearLinks**. Link information associated with Web parts MUST NOT be deleted by **proc_ClearLinks**.

```
PROCEDURE proc_ClearLinks(
      @SiteId                            uniqueidentifier,
      @DirName                           nvarchar(256),
      @LeafName                          nvarchar(128),
      @Level                             tinyint,
      @FieldId                           uniqueidentifier,
      @RequestGuid                       uniqueidentifier   = NULL   OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection that contains the list item with the link information to be deleted.

@DirName: The directory name of the list item with the link information to be deleted.

@LeafName: The leaf name of the list item with the link information to be deleted.

@Level: The **Publishing Level** type (section 2.2.2.6) value associated with the list item with the link information to be deleted.

@FieldId: The field identifier of the field in the list item with the link information to be deleted.

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

Return Values: The **proc_ClearLinks** stored procedure returns an integer return code, which MUST be 0. The **proc_ClearLinks** stored procedure MUST NOT return a result set.

3.1.5.9 proc_CreateDir

The **proc_CreateDir** stored procedure is invoked to create a directory or folder with a specified name at a specified location.

```
PROCEDURE proc_CreateDir(
      @DirSiteId                uniqueidentifier,
      @DirWebId                 uniqueidentifier,
      @DirDirName               nvarchar(256)                  OUTPUT,
      @DirLeafName              nvarchar(128)                  OUTPUT,
      @DirLevel                 tinyint,
      @AddMinorVersion          bit,
      @DocFlags                 int,
      @CreateDirFlags           int,
      @UserId                   int                =NULL,
      @ProgId                   nvarchar(255)      =NULL,
      @DirMetaInfo              varbinary(max)     =NULL,
      @DirMetaInfoSize          int                =0,
      @DirClientId              varbinary(16)      =NULL,
      @DirId                    uniqueidentifier   =NULL       OUTPUT,
      @ScopeId                  uniqueidentifier   =NULL       OUTPUT,
      @DoclibRowIdRequired      int                =NULL,
      @ReverseSortOrder         bit                =0,
      @ScopeIdOverride          uniqueidentifier   =NULL,
      @bAlreadyExists           bit                =NULL       OUTPUT,
      @RequestGuid              uniqueidentifier   =NULL       OUTPUT
);
```

@DirSiteId: The site collection identifier (section 2.2.1.9) for the site collection containing the directory to be created.

@DirWebId: The site identifier (section 2.2.1.11) for the site containing the directory to be created.

@DirDirName: This is both an input and an output parameter. The input parameter MUST specify the store-relative form URL of the parent location in which the specified directory is to be created. The directory name of the created directory MUST be returned as the output parameter value.

@DirLeafName: This is both an input and an output parameter. The input parameter MUST specify the name of the directory to be created in the parent location specified by the *@DirDirName* parameter. The leaf name of the created directory MUST be returned as the output parameter value.

@DirLevel: A **Publishing Level** type (section 2.2.2.6) value specifying the publishing status of the directory to be created. This value MUST be 1 (Published) if the directory is not being created in a list or document library. If this parameter is 2, specifying that the list item is to be created as a draft, then the *@UserId* value MUST be set as the draft owner for the created directory.

@AddMinorVersion: If this parameter has a value of 1, **proc_CreateDir** MUST set the major version number of the created directory to 0 and the minor version number to 1. If this parameter has a value of 0, then **proc_CreateDir** MUST set the major version number of the created directory to 1 and the minor version number to 0. This parameter MUST NOT be NULL.

@DocFlags: A **Doc Flags** (section 2.2.2.3) value specifying metadata about the directory to be created. If the *@DocFlags* value has the 0x00002000 bit set (specifying a custom ordering of content types), then this bit flag MUST be ignored and MUST NOT be included in the metadata associated with the created directory.

@CreateDirFlags: A 4-byte unsigned integer bit mask containing options used while creating the directory. The value MUST be 0, with none of the bits set, or bits MUST be set as follows.

Value	Description
0x00000007	These three bits contain an **Attachments Flag** (section 2.2.3.1) value.
0x00000008	Return an access denied error if the specified directory already exists.
0x00000010	Do not promote the directory to a document library.
0x00000020	The directory contains a **thicket**.
0x00000380	These three bits contain a **Moderation Status** (section 2.2.3.13) value.
0xFFFFFC40	These values are currently unused and MUST be ignored.

@UserId: The user identifier (section 2.2.1.13) for the current user who is directly or indirectly requesting the directory creation. If this parameter is not NULL, then it MUST be used by **proc_CreateDir** for permission checking and setting ownership of the created directory. If this parameter is NULL, then the ownership of the created directory MUST be set from the parent.

@ProgId: An optional parameter that specifies a preferred application to open the directory. The *@ProgId* value is used to distinguish between different applications that save files with a given file extension. This can be NULL.

@DirMetaInfo: The metadata information for the directory to be created. This can be NULL.

@DirMetaInfoSize: Size, in bytes, of the directory's metadata info. This MUST be 0 if *@DirMetaInfo* is NULL.

@DirClientId: An optional parameter that specifies the client identifier of the directory to be created. This can be NULL.

@DirId: The identifier for the created directory. This is both an input and output parameter. If this parameter is passed in to **proc_CreateDir** with a non-NULL value, then this value MUST be unique in the back-end database server and MUST be used as the **document identifier** (section 2.2.1.2) for the directory to be created. If this parameter is passed in with a NULL value, then a new **uniqueidentifier** value MUST be generated for the directory. If **proc_CreateDir** creates a new directory, then it MUST return the document identifier for the created directory in the output parameter upon successful completion. If **proc_CreateDir** does not create a new directory (for example, if the directory already exists), then this output parameter MUST be ignored.

@ScopeId: An output parameter that MUST contain the non-NULL value specified for *@ScopeIdOverride* or, if a NULL value is specified for *@ScopeIdOverride*, the scope identifier (section 2.2.1.8) of the parent directory. This identifies the specific ACL to use for calculating the permission settings on the created directory.

@DoclibRowIdRequired: Specifies the list item identifier (section 2.2.1.6) to be used for this directory if created within a document library. If this parameter is not NULL and the directory to be created is within a document library, then **proc_CreateDir** MUST set this as the list item identifier for the created directory. This parameter MUST be NULL for directories that are not created within a document library.

@ReverseSortOrder: Specifies whether the directory SHOULD sort using the behavior of the file or directory type. The bit MUST be set to "1" for directories to sort like files; otherwise, it MUST be set to "0".

@ScopeIdOverride: Specifies the scope to use for the directory to be created. If this parameter is not NULL, then it MUST be set as the scope identifier for the directory. Otherwise, the scope identifier of the parent directory MUST be used. The scope set for the created directory MUST be returned in the *@ScopeId* output parameter.

@bAlreadyExists: An output parameter containing a bit flag set to 0 if the directory did not already exist and set to 1 if the directory already existed.

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

Return Values: The **proc_CreateDir** stored procedure MUST return an integer return code that MUST be in the following table.

Value	Description
0	Successful execution.
3	The parent directory was not found.
5	The current user does not have sufficient permissions to create the directory at the specified location.
13	The list item to be added is not valid.
16	Adding the list item caused updating of an existing list item to fail.
80	The specified directory already exists.
87	Unable to add the list item because the input parameters do not match existing list items, or an error occurred during a table update operation.
206	The file or directory name is too long.
212	The site collection is locked.
1150	Failed to update the list.
1816	There is not enough database quota for the current user to complete the operation.

The **proc_CreateDir** stored procedure MUST return no result sets.

3.1.5.10 proc_DeleteAllDocumentVersions

The **proc_DeleteAllDocumentVersions** stored procedure is invoked to delete either the draft versions or older **published versions** of a document and optionally place them in the recycle bin.

```
PROCEDURE proc_DeleteAllDocumentVersions(
      @DocSiteId                    uniqueidentifier,
      @DocDirName                   nvarchar(256),
      @DocLeafName                  nvarchar(128),
      @UserId                       int,
      @DeleteOp                     int,
      @Level                        tinyint,
      @RequestGuid                  uniqueidentifier   = NULL      OUTPUT
);
```

@DocSiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the document whose specified versions are being deleted.

@DocDirName: The directory name of the document whose specified versions will be deleted. If there is no forward slash in the directory name, then the value MUST be an empty string.

@DocLeafName: The leaf name of the document whose specified versions will be deleted.

@UserId: The user identifier (section 2.2.1.13) of the current user requesting the deletion. This value MUST refer to an existing user identifier for the site collection specified by *@DocSiteId*.

@DeleteOp: A parameter specifying the delete options. The value MUST be listed in the following table.

Value	Description
3	The deleted document versions MUST NOT be placed in the recycle bin (nonrecoverable delete).
4	The deleted document versions MUST be placed in the recycle bin (recoverable delete).

@Level: A **Publishing Level** type (section 2.2.2.6) value specifying which versions of the document are to be deleted. The value MUST be listed in the following table.

Value	Description
1	All versions of the document MUST be deleted, excluding the current version and the current published version. If the current version and the current published version are the same, then only one version will remain.
2	All draft versions of the document MUST be deleted, excluding the current version.

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

Return Values: The **proc_DeleteAllDocumentVersions** stored procedure returns an integer return code that MUST be listed in the following table.

Value	Description
0	Successful execution.
2	The document specified cannot be found in the site collection.
1359	Invalid Parameter: *@DeleteOp* does not contain a valid value.

The **proc_DeleteAllDocumentVersions** stored procedure MUST NOT return a result set.

3.1.5.11 proc_DeleteDocBuildDependencySet

The **proc_DeleteDocBuildDependencySet** stored procedure is invoked to delete a **build dependency set** for a specified document.

```
PROCEDURE proc_DeleteDocBuildDependencySet(
      @DocSiteId                    uniqueidentifier,
      @DocDirName                   nvarchar(256),
      @DocLeafName                  nvarchar(128),
      @Level                        tinyint
      @RequestGuid                  uniqueidentifier   = NULL      OUTPUT
);
```

@DocSiteId: The site collection identifier (section 2.2.1.9) for a site collection containing the document.

@DocDirName: The directory name containing the document. If there is no forward slash in the directory name, the value MUST be an empty string.

@DocLeafName: The leaf name containing the document.

@Level: The **Publishing Level** type (section 2.2.2.6) value of the document.

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

Return Values: The **proc_DeleteDocBuildDependencySet** stored procedure MUST return an integer return code of 0.

The **proc_DeleteDocBuildDependencySet** stored procedure MUST NOT return a result set.

3.1.5.12 proc_DeleteDocumentVersion

The **proc_DeleteDocumentVersion** stored procedure is invoked to delete a document version and optionally place it in the recycle bin. A current version (whether published or draft) cannot be deleted. A current version is either the current published version or the current draft version of the document.

```
PROCEDURE proc_DeleteDocumentVersion(
      @DocSiteId                    uniqueidentifier,
      @DocDirName                   nvarchar(256),
      @DocLeafName                  nvarchar(128),
      @DocVersion                   int,
      @UserId                       int,
      @DeleteOp                     int
      @RequestGuid                  uniqueidentifier   = NULL      OUTPUT
);
```

@DocSiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the document whose specified version is being deleted.

@DocDirName: The directory name of the document to delete. If there is no forward slash in the directory name, then the value MUST be an empty string.

@DocLeafName: The leaf name of the document to delete.

@DocVersion: The version of the document to delete. This value MUST NOT be a current version.

@UserId: The user identifier (section 2.2.1.13) of the current user requesting the deletion. This value MUST refer to an existing user identifier for the specified site collection.

@DeleteOp: A value determining delete options. The value MUST be listed in the following table.

Value	Description
3	The deleted document version MUST NOT be placed in the recycle bin (non-recoverable delete).
4	The deleted document version MUST be placed in the recycle bin (recoverable delete).

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

Return Values: The **proc_DeleteDocumentVersion** stored procedure returns an integer return code that MUST be in the following table.

Value	Description
0	Successful execution.
2	The document version specified by *@DocSiteId*, *@DocDirName*, *@DocLeafName* and *@DocVersion* cannot be found.

Value	Description
186	Invalid **Parameter:** An invalid version was specified. The value specified in *@DocVersion* is a current version.

The **proc_DeleteDocumentVersion** stored procedure MUST NOT return a result set.

3.1.5.13 proc_DeleteUrl

The **proc_DeleteUrl** stored procedure is invoked to delete a document. **proc_DeleteUrl** accepts documents of all types except sites or attachment folders.

```
    PROCEDURE proc_DeleteUrl(
          @WebSiteId                      uniqueidentifier,
          @WebId                          uniqueidentifier,
          @Url                            nvarchar(260),
          @UserId                         int,
          @LogChange                      bit                 =1,
          @ListDeletedUrls                bit                 =1,
          @ListDeletedAliases             bit                 =0,
          @AttachmentsFlag                tinyint             =0,
          @AttachmentOp                   int                 =3,
          @IgnoreCheckedOutFiles          bit                 =0,
          @IgnoreLookupRelationshipsCheck bit                 =0,
          @DeleteOp                       int                 =3,
          @ThresholdRowCount              int                 =0,
          @QueryAuditFlags                bit                 =0,
          @FailedUrl                      nvarchar(260)       =NULL       OUTPUT,
          @DeleteTransactionId            uniqueidentifier    =
              '00000000-0000-0000-0000-000000000000'         OUTPUT,
          @eventData                      varbinary(max)      =NULL,
          @acl                            varbinary(max)      =NULL,
          @IsCascadeDeleteOperation       bit                 =0,
          @IsCascadeParent                bit                 =0,
          @ChildDeleteTransactionId       varbinary(16)       =NULL       OUTPUT,
          @RequestGuid                    uniqueidentifier    =NULL       OUTPUT
    );
```

@WebSiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the document.

@WebId: The site identifier (section 2.2.1.11) of the site containing the document.

@Url: The store-relative form URL of the document.

@UserId: The user identifier (section 2.2.1.13) for the current user requesting the operation. This parameter is used by **proc_DeleteUrl** for purposes of permission-checking. This value MUST refer to an existing user identifier for the specified site collection.

@LogChange: This parameter is reserved for internal use and MUST be the default value.

@ListDeletedUrls: If this bit is set to 1, then it specifies that result set information about the deleted documents is requested. See Deleted Documents Result Set (section 3.1.5.13.1).

@ListDeletedAliases: If this bit is set to 1, then it specifies that result set information about deleted lists that had configured email addresses is requested. See Deleted Aliased Lists Result Set (section 3.1.5.13.2).

@AttachmentsFlag: An **Attachments Flag** (section 2.2.3.1) describing the document specified by *@Url*.

@AttachmentOp: This parameter specifies whether or not the attachment folder record in the store MUST be updated to reflect whether or not it has any attachments remaining, and which metadata in the store for the folder MUST be refreshed, as follows.

Value	Description
0	No updates.
1	The attachment folder record in the store MUST be updated to reflect whether it has any attachments remaining.
2	The attachment folder record in the store MUST be updated, and the internal version number of the folder MUST be incremented.
3	The attachment folder record in the store MUST be updated, and the last modified date and time of the folder MUST be refreshed.

@IgnoreCheckedOutFiles: This parameter is reserved for internal use and MUST be the default value.

***@IgnoreLookupRelationshipsCheck*:** This parameter is reserved for internal use and MUST be the default value.

@DeleteOp: This value defines the type of delete operation to attempt. If no value is specified, then a default value of 3 is used. Otherwise, *@DeleteOp* MUST be one of the values listed in the following table.

Value	Description
3	Delete the item without placing the deleted item into the recycle bin.
4	Delete the item and place the deleted item into the recycle bin.

@ThresholdRowCount: If this value is not 0, then the stored procedure MUST NOT proceed with the delete operation if the number of document rows deleted by the operation exceeds the value of this parameter.

@QueryAuditFlags: This parameter is set to 1 to specify that an **audit entry** MUST be created for the delete operation.

@FailedUrl: An output parameter indicating the store-relative form URL at which the delete operation failed. This parameter MUST be set to NULL if the deletion was successful.

@DeleteTransactionId: This parameter is a value which is used to identify all files and items that are deleted as part of a single SQL transaction. The first call to **proc_DeleteUrl** within the SQL transaction MUST pass NULL. The implementation assigns a new delete transaction ID and returns it as output. On output, this parameter MUST return either an all-zero GUID, indicating that the item was not placed into the recycle bin, or a valid transaction ID generated by the implementation. All subsequent calls to **proc_DeleteUrl** within the SQL transaction MUST pass the value returned by the first call.

@eventData: This parameter is reserved for internal use and MUST be the default value.

@acl: This parameter is reserved for internal use and MUST be the default value.

@IsCascadeDeleteOperation: This parameter is reserved for internal use and MUST be the default value.

@IsCascadeParent: This parameter is reserved for internal use and MUST be the default value.

@ChildDeleteTransactionId: This parameter is reserved for internal use and MUST be the default value.

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

Return Values: The **proc_DeleteUrl** stored procedure returns an integer return code, which MUST be listed in the following table.

Value	Description
0	Successful execution.
3	The specified URL was not found.
5	The user is not authorized to make this change.
33	Cannot delete a folder containing checked-out or locked files.
36	The number of rows deleted by the operation exceeds the threshold.
50	Cannot delete a site or an attachments folder.
51	Cannot delete a forms folder.
138	Cannot delete a URL containing lists.
161	Cannot delete a URL that contains sites.
206	The URL is too long.
1150	A concurrency violation or unknown error occurred.
1359	Internal error, or bad parameter specified (**Attachments Flag** specifies an attachment file, but the URL to be deleted is a file in a document library).
4335	Cannot delete a URL due to a lookup relationship referential integrity constraint.
8398	There was an error deleting a list. At least one list could not be deleted.

The **proc_DeleteUrl** stored procedure MUST return zero, one, or two result sets in the order shown.

3.1.5.13.1 Deleted Documents Result Set

The Deleted Documents Result Set contains information about the deleted documents. It MUST be returned on successful completion when *@ListDeletedUrls* is set to 1.

If the Deleted Documents Result Set is returned, it MUST contain one row for each document deleted.

If the document specified by *@Url* has a **Document Store** type (section 2.2.2.4) of 0 (File), then the columns in the Deleted Documents Result Set MUST NOT be named; else they are named as **follows:**

```
{Url}                              nvarchar(260),
Type                             tinyint;
```

Url: The store-relative form URL of the deleted document.

Type: The **Document Store** type of the deleted document.

3.1.5.13.2 Deleted Aliased Lists Result Set

The Deleted Aliased Lists Result Set contains information about all deleted lists that were configured with an email address. Such lists provide implementation-specific email integration features. The Deleted Aliased Lists Result Set MUST be returned only when *@ListDeletedAliases* is set to 1 and the document specified by *@Url* does not have a **Document Store** type (section 2.2.2.4) of 0 (File).

If the Deleted Aliased Lists Result Set is returned, there MUST be one row returned for each deleted alias list.

```
          {WebSiteId}                     uniqueidentifier,
          tp_WebId                        uniqueidentifier,
          tp_Id                           uniqueidentifier;
```

{WebSiteId}: The site collection identifier (section 2.2.1.9) of the site collection containing the deleted list.

tp_WebId: The site identifier (section 2.2.1.11) of the site containing the list.

tp_Id: The list identifier (section 2.2.1.5) of the deleted list.

3.1.5.13.3 Empty Deleted Aliased Lists Result Set

The Empty Deleted Aliased Lists Result Set MUST be returned only if *@ListDeletedAliases* is set to 1 and the document specified by *@Url* has a **Document Store** type (section 2.2.2.4) of 0 (File). Deletion of a file does not cause the deletion of lists, and the Empty Deleted Aliased Lists Result Set is returned to indicate that the input parameter *@ListDeletedAliases* cannot be satisfied.

The Empty Deleted Aliased Lists Result Set has zero rows with a schema of two unnamed columns.

3.1.5.14 proc_DisableRbs

The **proc_DisableRbs** stored procedure is invoked to prepare the back-end database server to stop storing content in remote blob storage, and reclaim any back-end database server resources allocated with **proc_EnableRbs**. When remote BLOB storage is disabled, the administrative property in the database for remote BLOB storage, **RbsEnabled**, MUST be added and updated to a value of NULL.

```
    PROCEDURE proc_DisableRbs (
    @RequestGuid uniqueidentifier = null OUTPUT
    );
```

@RequestGuid: The optional request identifier for the current request

Return values: The **proc_DisableRbs** stored procedure returns an integer return code, which MUST be in the following table.

Value	Description
1060	The back-end database server does not support remote blob storage.
108	The back-end database server cannot disable remote blob storage because content is still stored in remote blob storage.
0	Successful execution.

The **proc_DisableRbs** stored procedure MUST NOT return any result sets.

3.1.5.15 proc_DirtyDependents

The **proc_DirtyDependents** stored procedure is invoked to mark all items that depend on a given document, configuration setting, navigation structure, or usage statistic as "dirty", so that subsequent action can be taken to update them as necessary. In this context, a "dependent" item is an item that requires an update to its metadata when another item is modified. **proc_DirtyDependents** follows the full dependency chain until all dependent items are marked.

```
PROCEDURE proc_DirtyDependents(
      @SiteId                        uniqueidentifier,
      @DepType                       tinyint,
      @DepDesc                       nvarchar(270),
      @DepDescLike                   nvarchar(260)      =NULL,
      @RequestGuid                   uniqueidentifier   =NULL      OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) for a site collection that contains the items with dependencies.

@DepType: The dependency type. The following values are valid.

Value	Description
1	Document dependency. This updates items dependent on the specified document. The *@DepDesc* parameter is the store-relative form URL of the document that has changed.
3	Configuration dependency. This updates items dependent on changes to system configuration metadata, as specified in [MC-FPSEWM] section 2.2.2.3. The *@DepDesc* parameter is the metakey for the metadata that has changed.
4	Navigation dependency. This updates items dependent on changes to navigation structures. The *@DepDesc* parameter contains the **Web-Navigation-URL**, as specified in [MC-FPSEWM] section 2.2.2.2.34, for a navigation structure node.
7	Usage dependency. This updates items dependent on changes to site usage statistics. The *@DepDesc* parameter is the store-relative form URL of the site.

@DepDesc: This specifies the dependency description parameter, which varies according to the value of *@DepType*, as described in the preceding table.

@DepDescLike: This parameter is unused and MUST be ignored.

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

Return Values: The **proc_DirtyDependents** stored procedure returns an integer return code, which MUST be one that is listed in the following table.

Value	Description
0	Successful execution.

The **proc_DirtyDependents** stored procedure returns no result sets.

3.1.5.16 proc_EnableRbs

The **proc_EnableRbs** stored procedure is invoked to prepare the back-end database server to store content in remote blob storage.

```
PROCEDURE proc_EnableRbs (
@RequestGuid uniqueidentifier = null OUTPUT,
);
```

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

Return values: The **proc_EnableRbs** stored procedure returns an integer return code, which MUST be in the following table.

Value	Description
1060	The back-end database server does not support remote blob storage.
0	Successful execution.

The **proc_EnableRbs** stored procedure MUST NOT return any result sets.

3.1.5.17 proc_EnumLists

The **proc_EnumLists** stored procedure returns a list of the lists in a site, along with their associated metadata.

```
PROCEDURE proc_EnumLists (
      @WebId                         uniqueidentifier,
      @Collation                     nvarchar(48),
      @BaseType                      int                  =NULL,
      @BaseType2                     int                  =NULL,
      @BaseType3                     int                  =NULL,
      @BaseType4                     int                  =NULL,
      @ServerTemplate                int                  =NULL,
      @FMobileDefaultViewUrl         bit                  =NULL,
      @FRootFolder                   bit                  =NULL,
      @ListFlags                     int                  =NULL,
      @FAclInfo                      int                  =NULL,
      @Scopes                        varbinary(max)       =NULL,
      @FRecycleBinInfo               bit                  =NULL,
      @UserId                        int                  =NULL,
      @FGP                           bit                  =NULL,
      @RequestGuid                   uniqueidentifier     =NULL OUTPUT
);
```

@WebId: The site identifier (section 2.2.1.11) for the site containing the requested lists.

@Collation: This parameter determines the sort order in which the lists are returned in the List Information Result Set (section 3.1.5.17.1) (based on the **tp_Title** column). This MUST be the **Windows collation name** of one of the valid **Collation Order Enumeration** (section 2.2.3.4) values, with the case-insensitive and accent-sensitive flags set. For example, the collation order "Latin1_General" with the case-insensitive and accent-sensitive flags set, has a Windows collation **Name** of "Latin1_General_CI_AS".

@BaseType: A parameter that restricts the returned lists by **List Base** type (section 2.2.3.11). If this parameter is set one of the **List Base** type values, the lists returned MUST be restricted to lists with this **List Base** type and with the **List Base** types specified by *@BaseType2*, *@BaseType3*, and *@BaseType4*, if any. If this parameter is set to "-1" or NULL, then the lists matching any **List Base** type value will be returned, and parameters *@BaseType2*, *@BaseType3*, and *@BaseType4* MUST be ignored by **proc_EnumLists**.

@BaseType2, @BaseType3, @BaseType4: Additional parameters used to specify additional **List Base** types for lists returned, depending on the value of the *@BaseType* parameter. If any of these additional parameters are set to "-1" or NULL, then the additional parameter MUST be ignored.

@ServerTemplate: The identifier for the **List Server Template** (section 2.2.3.12) that defines the base structure of the lists to be returned. If this parameter is set to one of the **List Server Template** values, then those lists returned MUST be restricted to those whose **List Server Template** value matches *@ServerTemplate*. If this parameter is set to "-1" or NULL, then lists matching any **List Server Template** will be returned.

@FMobileDefaultViewUrl: A bit specifying whether information about the mobile default view URL is requested. If this parameter is set to "1", then the information MUST be returned in the **tp_MobileDefaultViewUrl** column in the List Information Result Set. Otherwise, the **tp_MobileDefaultViewUrl** column MUST be returned with NULL values.

@FRootFolder: A bit specifying whether information about the root folder URL is requested. If this parameter is set to "1", then the information MUST be returned in the **tp_RootFolder** column in the List Information Result Set. Otherwise, the **tp_RootFolder** column MUST be returned with NULL values.

@ListFlags: A **List Flags** (section 2.2.2.5) value restricting the data returned by the List Information Result Set. If this parameter is not NULL, then the stored procedure MUST only return data in the List Information Result Set for lists with a **tp_Flags** value exactly matching the *@ListFlags* value. Otherwise, this parameter MUST be ignored.

@FAclInfo: A flag specifying whether ACL information is requested. If this parameter is set to "1", then the ACL information MUST be returned in the **tp_ACL** column in the List Information Result Set. Otherwise, the **tp_ACL** column MUST be returned with NULL values.

@Scopes: A concatenation of one or more scope identifiers (section 2.2.1.8), represented as 16-byte binary strings with no delimiters, each specifying a scope identifier. If this parameter is not NULL, then this specifies that the stored procedure MUST only return data in the List Information Result Set for lists matching those scopes specified by this parameter. Otherwise, this parameter MUST be ignored.

@FRecycleBinInfo: A bit specifying whether to return information about the contents of the implementation-specific **Recycle Bin**. If this parameter is set to "1", then the Recycle Bin Information Result Set (section 3.1.5.17.2) MUST be returned. Otherwise, the NULL Result Set (section 3.1.5.17.3) MUST be returned.

@UserId: A user identifier (section 2.2.1.13) to specify the information to return in the Recycle Bin Information Result Set. If the *@FRecycleBinInfo* parameter is set to "1", then the stored procedure MUST only return data in the Recycle Bin Information Result Set. Otherwise, this parameter MUST be ignored.

@FGP: A bit restricting the data returned by the List Information Result Set. If this parameter is set to 1, then the stored procedure MUST only return data in the List Information Result Set for lists that are using per-list permissions, which are used to manage permissions more finely by setting unique permissions on a per-list basis. Otherwise, this parameter MUST be ignored.

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

Return Values: The **proc_EnumLists** stored procedure returns an integer return code, which MUST be 0. This stored procedure MUST return two result sets in the order specified in the following sections.

3.1.5.17.1 List Information Result Set

The List Information Result Set returns information about lists contained in the specified site. The List Information Result Set MUST be returned and MUST contain zero or more rows, one for each list that matches the specified input parameters.

```
          tp_DocTemplateUrl              nvarchar(386),
          tp_DefaultViewUrl              nvarchar(386),
          tp_MobileDefaultViewUrl        nvarchar(256),
          tp_Id                          uniqueidentifier,
          tp_Title                       nvarchar(255),
          tp_Description                 nvarchar(max),
          tp_ImageUrl                    nvarchar(255),
          tp_Name                        nvarchar(38),
          tp_BaseType                    int,
          tp_FeatureId                   uniqueidentifier,
          tp_ServerTemplate              int,
          tp_Created                     datetime,
          tp_Modified                    datetime,
          tp_LastDeleted                 datetime,
          tp_Version                     int,
          tp_Direction                   int,
          tp_ThumbnailSize               int,
          tp_WebImageWidth               int,
          tp_WebImageHeight              int,
          tp_Flags                       bigint,
          tp_ItemCount                   int,
          tp_AnonymousPermMask           bigint,
          tp_RootFolder                  nvarchar(260),
          tp_ReadSecurity                int,
          tp_WriteSecurity               int,
          tp_Author                      int,
          tp_EventSinkAssembly           nvarchar(255),
          tp_EventSinkClass              nvarchar(255),
          tp_EventSinkData               nvarchar(255),
          tp_EmailAlias                  nvarchar(128),
          tp_WebFullUrl                  nvarchar(256),
          tp_WebId                       uniqueidentifier,
          tp_SendtoLocation              nvarchar(512),
          tp_ScopeId                     uniqueidentifier,
          {tp_MaxMajorVersionCount}      int,
          {tp_MaxMajorwithMinorVersionCount}int,
          tp_DefaultWorkflowId           uniqueidentifier,
          tp_HasInternalFGP              bit,
          tp_NoThrottleListOperations    bit,
          HasRelatedLists                bit,
          tp_ValidationFormula           nvarchar(1024),
          tp_ValidationMessage           nvarchar(1024),
          TitleResource                  nvarchar(256),
          DescriptionResource            nvarchar(256),
          tp_ACL                         varbinary(max);
```

tp_DocTemplateUrl: Contains the URL in store-relative form of the **document template** associated with the list, if any, or NULL if none.

tp_DefaultViewUrl: Contains the URL in store-relative form default view of the list. This value MUST NOT be NULL.

tp_MobileDefaultViewUrl: When *@FMobileDefaultViewUrl* is set to "1", this contains the URL in store-relative form of the default view for **mobile devices**, if any, or NULL if none has been set. Otherwise, this MUST be NULL.

tp_Id: Contains the list identifier (section 2.2.1.5) for the list.

tp_Title: Contains the display name (a short user-provided text description) to identify the list.

tp_Description: Contains user-provided text describing the list.

tp_ImageUrl: Contains the URL in store-relative form holding an image associated with the list.

tp_Name: Contains a string representation of the **tp_Id** GUID delimited by braces.

tp_BaseType: Contains the **List Base** type (section 2.2.3.11) value from which the list is derived.

tp_FeatureId: Contains a feature identifier (section 2.2.1.4) for a feature associated with the list.

tp_ServerTemplate: Contains the value of the of the **list template** that defines the base structure of the list. This value can either be in the **List Server Template** (section 2.2.3.12) enumeration or can be a custom value as defined in the disk-based template (a template known to all web front-end clients) of the list. A custom list template value MUST be unique and MUST be than 10000.

tp_Created: Contains the time in **UTC**, specifying when the list was created.

tp_Modified: Contains the time in UTC, specifying when the list was last modified.

tp_LastDeleted: Contains a time stamp, in UTC format, specifying when a list item was last deleted from the list. If no list item has been deleted, contains the value of **tp_Created**.

tp_Version: Contains an implementation-specific, internal version counter used in tracking modifications to the list's settings.

tp_Direction: Contains a value specifying the direction of text flow for front-end web server elements. Valid values are in the following table.

Value	Description
0	No explicit direction is specified.
1	Text flow SHOULD be left to right.
2	Text flow SHOULD be right to left.

tp_ThumbnailSize: The width, in pixels, specified for use when creating thumbnail images of list items within this list. This value MUST be NULL for lists that do not have a **List Server Template** value of 109. Thumbnails images are generated by the front-end web server for documents and they are implementation-specific capabilities.

tp_WebImageWidth: The width, in pixels, specified for use when creating images of list items within this list. This value MUST be NULL for lists that do not have a **List Server Template** value of 109.

tp_WebImageHeight: The height, in pixels, specified for use when creating images of list items within this list. This value MUST be NULL for lists that do not have a **List Server Template** value of 109.

tp_Flags: Contains a **List Flags** (section 2.2.2.5) value describing list properties.

tp_ItemCount: Contains a count of list items in the list.

tp_AnonymousPermMask: Contains the **WSS Rights Mask** (section 2.2.2.14) that applies to an **anonymous user** for the list.

tp_RootFolder: When the *@FRootFolder* parameter is set to "1", this contains the URL in store-relative form of the root folder of the list. Otherwise, this MUST be NULL.

tp_ReadSecurity: Contains a value identifying the **security policy** for **read-only mode** for list items. If this value is set to "1", then users with read-only mode permissions can read all list items. Otherwise, users with read-only mode permissions can read only those list items they create.

tp_WriteSecurity: Contains a value specifying the security policy in use for write access to list items. Valid values are in the following table.

Value	Description
1	Users with write permissions have write access to all list items.
2	Users with write permissions have write access to those list items they create.
4	Users have no write access to any list items.

tp_Author: Contains the user identifier (section 2.2.1.13) of the user who created the list.

tp_EventSinkAssembly: Contains the **assembly name** of the **event sink** handler if an event sink handler is registered for the list or NULL if none exists.

tp_EventSinkClass: Contains the assembly class name of the event sink handler if an event sink handler is registered for the list or NULL if none exists.

tp_EventSinkData: Contains string data specific to the implementation of the event sink associated with this list or NULL if none exists.

tp_EmailAlias: Contains the email address of the list. This alias is used to allow files to be sent directly to the list through an implementation-specific email handling feature. Can be NULL if the list is not an **email enabled list**.

tp_WebFullUrl: Contains the URL in store-relative form of the site that contains the list.

tp_WebId: Contains the site identifier (section 2.2.1.11) of the site that contains the list.

tp_SendToLocation: Contains an implementation-specific string of the URL used to copy list items to alternative locations. This parameter can be NULL.

tp_ScopeId: Contains the scope identifier (section 2.2.1.8) for the list.

{tp_MaxMajorVersionCount}: Contains the number of major versions that will be retained for this document if versioning is enabled.

{tp_MaxMajorwithMinorVersionCount}: Contains the number of major versions that will have their associated minor versions retained for this document if versioning is enabled.

tp_DefaultWorkflowId: Contains the **workflow** identifier (section 2.2.1.16) of the default workflow associated with the list or NULL if none exists.

tp_HasInternalFGP: This flag is set to "1" if there have ever been list items that have had a unique **access control list (ACL)** applied.

tp_NoThrottleListOperations: If this list is exempt from resource throttling operations, then this value MUST be 1. Otherwise, it MUST be 0.

HasRelatedLists: This value MUST be NULL.

tp_ValidationFormula: This value MUST be NULL.

tp_ValidationMessage: This value MUST be NULL.

TitleResource: This value MUST be NULL.

DescriptionResource: This value MUST be NULL.

tp_ACL: When the *@FAclInfo* parameter is set to "1", this MUST contain a binary array of the ACL for this list if one is defined; otherwise, it MUST be NULL.

3.1.5.17.2 Recycle Bin Information Result Set

The Recycle Bin Information Result Set returns information about list items from the specified list that the specified user marked as deleted in the Recycle Bin.

If the *@FRecycleBinInfo* parameter is "1", the Recycle Bin Information Result Set MUST be returned and MUST contain a single row with two columns specified as follows.

```
          {RecycleBinCount}              int,
          {RecycleBinSize}               int;
```

{RecycleBinCount}: Contains a count of the number of items in the Recycle Bin matching the specified user and list.

{RecycleBinSize}: Contains the total size, in bytes, of items in the Recycle Bin matching the specified user and list.

3.1.5.17.3 NULL Result Set

The NULL Result Set is an empty placeholder returned when Recycle Bin information is not requested. The NULL Result Set MUST only be returned if the *@FRecycleBinInfo* parameter is not 1 and MUST contain no rows, in a schema consisting of a single, unnamed NULL column.

3.1.5.18 proc_FetchChunkFromDocStreams

The **proc_FetchChunkFromDocStreams** stored procedure is called to read a specific range of data from a document's content.

```
    PROCEDURE proc_FetchChunkFromDocStreams (
          @Offset               int,
          @Length               int,
          @SiteId               uniqueidentifier,
          @DocId                uniqueidentifier,
          @InternalVersion      int,
          @RequestGuid          uniqueidentifier     =NULL OUTPUT
    );
```

@Offset: The offset, in bytes, into the document content to begin reading.

@Length: The number of bytes to read from the document content.

@SiteId: The site collection identifier (section 2.2.1.9) for a site collection containing the document.

@DocId: The **document identifier** (section 2.2.1.2) of the document to read from.

@InternalVersion: The internal version number of the document to read from.

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

Return Values: The **proc_FetchChunkFromDocStreams** stored procedure returns an integer return code which MUST be listed in the following table.

Value	Description
0	Successful execution.
30	There was an IO error reading the data.

Result Sets: The **proc_FetchChunkFromDocStreams** stored procedure MUST return one result set.

3.1.5.18.1 Document Stream Chunk Result Set

The Document Stream Chunk Result Set contains a range of bytes read from a document's content.

```
        {Content}                         varbinary(max);
```

{Content}: The data read from the document content.

3.1.5.19 proc_FetchDocForHttpGet

The **proc_FetchDocForHttpGet** stored procedure is called to fetch a document stream and provide information necessary to render the document on a front-end web server.

```
    PROCEDURE proc_FetchDocForHttpGet(
          @DocSiteId                      uniqueidentifier,
          @DocDirName                     nvarchar(256),
          @DocLeafName                    nvarchar(128),
          @LooksLikeAttachmentFile        bit,
          @IfModifiedSince                datetime,
          @FetchType                      int,
          @ValidationType                 int,
          @ClientVersion                  int,
          @ClientId                       uniqueidentifier,
          @PageView                       tinyint,
          @FetchBuildDependencySet        bit,
          @SystemID                       varbinary(512),
          @CurrentVirusVendorID           int,
          @PrefetchListScope              bit,
          @ChunkSize                      int,
          @DGCacheVersion                 bigint,
          @MaxCheckinLevel                tinyint,
          @HonorLevel                     bit,
          @CurrentFolderUrl               nvarchar(260),
          @ThresholdRowCount              int,
          @Level                          tinyint            OUTPUT,
          @RequestGuid                    uniqueidentifier = NULL OUTPUT
    );
```

@DocSiteId: The site collection identifier (section 2.2.1.9) for a site collection containing the document.

@DocDirName: The directory name of the requested document.

@DocLeafName: The leaf name of the requested document.

@LooksLikeAttachmentFile: Boolean flag that specifies if the document is an attachment to a list item or an attachment folder.

@IfModifiedSince: Specifies the datetime in **UTC** of the last modified time stamp of the document. If *@ValidationType* is "2" or "3", then the document MUST only be fetched if this value is earlier than the current last modified time stamp value in the back-end database server.

@FetchType: Specifies the type of request. This parameter MUST be set to 0 to specify information is requested to satisfy a normal **HTTP GET** request, including the content of the document. This parameter MUST be set to 1 if information is needed only to satisfy an **HTTP HEAD** request, which does not include the content of the document. All nonzero values except 1 MUST also be treated as "0".

@ValidationType: This parameter MUST have a value of "0", "1", "2", or "3". If this parameter is "0", then *@ClientVersion*, *@ClientId*, and *@IfModifiedSince* MUST be ignored and the document MUST be fetched.

@ClientVersion: Specifies the internal version number of the document to fetch. If *@ValidationType* is "1" or "3", then the document MUST only be fetched if this value is equal to the current internal version number of the document on the back-end database server.

@ClientId: Specifies the **document identifier** (section 2.2.1.2) of the document to fetch. If *@ValidationType* is "1" or "3", then the document MUST only be fetched if this value is equal to the current document identifier of the document on the back-end database server.

@PageView: Non-NULL values indicate that the document is a view web page, an implementation-specific web page that renders an item or items in a list or document library. This also indicates that information is needed to enable front-end web server rendering based on metadata about the view page, the item or items being rendered, the user's browsing context, the overall site's navigation scheme, and the user's security privileges. Information requested includes site metadata, containing list metadata, users and site **groups** metadata, Web Parts, and related list metadata. In addition, a value of 1 indicates that the metadata SHOULD be security trimmed to the user specified by *@SystemId*. The value 0 or other non-NULL values indicate that information is requested as seen by all users. A NULL value indicates that the document is not a view web page.

@FetchBuildDependencySet: If this parameter is "1" and the document being fetched has a publishing level of published, then the Document Build Dependency Set Result Set (section 3.1.5.19.11) and Document Build Dependency Metadata Result Set (section 3.1.5.19.12) MUST be returned. Otherwise they MUST NOT be returned.

@SystemID: The **security identifier (SID)**, as specified in [MS-DTYP] section 2.4.2.1, of the current user, or NULL to indicate an anonymous user.

@CurrentVirusVendorID: Specifies an identifier for an anti-virus scanner. If this value is not NULL and does not match the current anti-virus scanner registered for the document, the document MUST NOT be fetched.

@PrefetchListScope: If this parameter is set to "1", the metadata of the list scope that contains the document MUST be returned.

@ChunkSize: Specifies the maximum amount of data, in bytes, to fetch from the document stream. If *@ChunkSize* is greater than the size of the entire document stream, then only the first *@ChunkSize* bytes MUST be fetched. Otherwise, all bytes of the document stream MUST be fetched.

@DGCacheVersion: Specifies an internal version number for the Domain Group Map Cache of the site collection that contains this document being request. If this parameter is "-2", then the Domain Group Cache BEDS Update Result Set (section 3.1.5.19.3) and the Domain Group Cache WFE Result Set (section 3.1.5.19.4) MUST NOT be returned. Otherwise, the Domain Group Cache BEDS Update Result Set MUST be returned and the Domain Group Cache WFE Update Result MUST only be returned if *@DCCacheVersion* is less than the current internal version number of the Domain Group Map Cache.

@MaxCheckinLevel: Specifies the publishing level of the document to fetch. If *@HonorLevel* is equal to "0", this parameter MUST be ignored.

@HonorLevel: If this parameter is not equal to "0", then the document with publishing level specified in *@MaxCheckInLevel* MUST be fetched. Otherwise, the document with the maximum publishing level available to the current user MUST be fetched.

@CurrentFolderUrl: The URL of the folder that the user was browsing before the specified document was requested. This is used to preserve the browsing context when the fetch request is redirected to a view page. When fetching an item in a list or a **file** in a document library, an attachment to an item, a folder, or a **list view**, the specified document will be a view page, and *@CurrentFolderUrl* will be used to render the browsing context.

@ThresholdRowCount: Specifies the maximum number of security scopes to be fetched. If *@PrefetchListScope* is not equal to "1", then this parameter MUST be ignored.

@Level: An output parameter specifying the publishing level of the document to be fetched.

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

Return Values: The **proc_FetchDocForHttpGet** stored procedure returns an integer return code, which MUST be listed in the following table.

Value	Description
0	Successful execution.
2	The document specified by *@DocSiteId*, *@DocDirName*, *@DocLeafName* (and *@MaxCheckinLevel*, if provided) was not found, or the document is an attachment folder; or the document is a folder, but no welcome page, **home page**, view web page, or other redirect web page was found for it.
18	Successful execution; based on values of *@ValidationType*, *@ClientId*, *@ClientVersion*, and *@IfModifiedSince*, the document was not fetched.
146	The document does not exist, but a welcome page was found, as specified by the Welcome Page Redirect Information Result Set (section 3.1.5.19.6).
1168	The site collection specified by *@DocSiteId* was not found.
1271	The site is read locked.

The **proc_FetchDocForHttpGet** stored procedure MUST return zero or more result sets upon successful execution. If the document is not found, zero result sets MUST be returned. The following result sets MUST be returned only if the document has a **Document Store** type (section 2.2.2.4) of 0 (File):

- Site Collection Audit Mask Result Set (section 3.1.5.19.9)
- List Audit Mask Result Set (section 3.1.5.19.10)
- Document Build Dependency Set Result Set (section 3.1.5.19.11)
- Document Build Dependency Metadata Result Set (section 3.1.5.19.12)
- Domain Group Cache Versions Result Set (section 3.1.5.19.2)
- Site Metadata Result Set (section 3.1.5.19.13)
- Event Receivers Result Set (section 3.1.5.19.14)
- Web Event Receiver Result Set (section 3.1.5.19.15)
- Site Features List Result Set (section 3.1.5.19.16)
- Web Parts Metadata, Personalized Result Set (section 3.1.5.19.17)
- Web Parts Metadata, Nonpersonalized Result Set (section 3.1.5.19.18)
- List Metadata Result Set (section 3.1.5.19.19)
- List Event Receivers Result Set (section 3.1.5.19.20)
- List Security Information Result Set (section 3.1.5.19.21)
- Site Collection Custom Action Result Set (section 3.1.5.19.22)

- Site Custom Actions Result Set (section 3.1.5.19.23)
- List Custom Actions Result Set (section 3.1.5.19.24)
- List Web Parts Result Set (section 3.1.5.19.25)
- Content Type Order Result Set (section 3.1.5.19.26)
- Current Folder Scope Result Set (section 3.1.5.19.27)
- Navigation Context Security Information Result Set (section 3.1.5.19.28)
- NULL Navigation Context Security Information Result Set (section 3.1.5.19.29)
- Empty Navigation Context Security Information Result Set (section 3.1.5.19.30)

Other **proc_FetchDocForHttpGet** result sets return conditionally, as described in each result set section.

3.1.5.19.1 HTTP Document Metadata Result Set

The HTTP Document Metadata Result Set returns the core document metadata, including the effective security permission and anonymous permission mask. This result set MUST be returned if the specified document exists.

```
          {Size}                        int,
          {DocFlags}                    int,
          {FullUrl}                     nvarchar(260),
          {WebId}                       uniqueidentifier,
          {FirstUniqueWebId}            uniqueidentifier,
          {SecurityProvider}            uniqueidentifier,
          {Dirty}                       bit,
          {TimeLastWritten}             datetime,
          {CharSet}                     int,
          {Version}                     int,
          {DocId}                       uniqueidentifier,
          {LeafName}                    nvarchar(128),
          InDocLibrary                  bit,
          IsAttachment                  bit,
          NeedManageListRight           int,
          {SiteFlags}                   int,
          Acl                           varbinary(max),
          AnonymousPermMask             bigint,
          {ListIdForPermissionCheck}    uniqueidentifier,
          {PermCheckedAgainstUniqueList} int,
          DraftOwnerId                  int,
          ListFlags                     bigint,
          Level                         tinyint,
          {IsCurrentVersion}            bit,
          {Type}                        tinyint,
          {VirusVendorID}               int,
          {VirusStatus}                 int,
          {VirusInfo}                   nvarchar(255),
          {ContentModifiedSince}        bit,
          {ProgId}                      nvarchar(255),
          {DoclibRowId}                 int,
          {Language}                    int,
          {DirName}                     nvarchar(256),
          {UIVersion}                   int,
          {ContentVersion}              int,
          {RbsCollectionId}             int
    ;
```

{Size}: The size, in bytes, of the document stream.

{DocFlags}: A **Doc Flags** (section 2.2.2.3) value describing the document.

{FullUrl}: The complete store-relative form URL for the requested document.

{WebId}: The site identifier (section 2.2.1.11) of the site containing the document.

{FirstUniqueWebId}: The site identifier of the site whose security permissions are the effective security permission for the site containing the specified document.

{SecurityProvider}: COM CLSID of the **security provider** for this site.

{Dirty}: A bit that specifies this document MUST have implementation-specific processing performed before its **stream** can be returned. If this document does not require processing, this value MUST be 0. If this document does not have a stream, this value MUST be NULL.

{TimeLastWritten}: The datetime, in **UTC**, when the document stream was last modified.

{CharSet}: A **character set** associated with the document. This value MUST be NULL or a valid **Windows code page** identifier.

{Version}: The internal version number of the document being returned.

{DocId}: The **document identifier** (section 2.2.1.2) of the requested document.

{LeafName}: The leaf name of the requested document.

InDocLibrary: If the document is in a document library, this value MUST be "1".

IsAttachment: If the document is an attachment, this value MUST be "1".

NeedManageListRight: If the user is required to have the Manage List right in order to read the document, this value MUST be "1".

{SiteFlags}: A **Site Collection Flags** (section 2.2.2.9) value describing the configuration of the site collection containing the document.

Acl: The **WSS ACL Format** (section 2.2.4.6) permissions for the document.

AnonymousPermMask: A **WSS Rights Mask** (section 2.2.2.14) that indicates the rights granted to an anonymous user on this document.

{ListIdForPermissionCheck}: The GUID for the list containing the document. This value MUST be NULL if this document is not in a list. **WSS Rights Mask**.

{PermCheckedAgainstUniqueList}: MUST be 0.

DraftOwnerId: The identifier for the user who published the document as a draft version. If the document is not a draft version, then this value MUST be NULL.

ListFlags: A **List Flags** (section 2.2.2.5) value describing the list that contains the document. If the document is not in a list, then the value MUST be 0.

Level: The publishing level value of this document.

{IsCurrentVersion}: If the document being returned is the current version, as defined by the implementation, then this value MUST be "1", else this value MUST be "0".

{Type}: The **Document Store** type (section 2.2.2.4) of this document.

{VirusVendorID}: The identifier of the anti-virus vendor that processed this document. This value MUST be NULL if the document has not been processed by an anti-virus scanner.

{VirusStatus}: The **Virus Status** (section 2.2.3.18) of the document. This value MUST be NULL if the requested document does not exist or if it has not been processed by an anti-virus scanner.

{VirusInfo}: A anti-virus scanner specific message returned by the anti-virus scanner when it last processed the document.

{ContentModifiedSince}: A bit indicating whether the document has been modified, depending on the validation type in use. MUST be set to 1 if any of the following are **true:** the document is a dynamic document type; the document requires a dependency update; validation type is "None" (0); validation type is "E-tag" (1) and the value of *@ClientVersion* disagrees with the document version in the store, or the value of *@ClientId* disagrees with the document identifier in the store; validation type is "Last modified" (2) and the last modification date of the document in the store is more recent than specified in *@IfModifiedSince*. In all other cases, **ContentModifiedSince** MUST be set to 0.

{ProgId}: Specifies an implementation-specific preferred application to open the document.

{DoclibRowId}: The identifier of the list item which represents this document if it belongs in a list. If the requested document is not contained in a list, then this value MUST be NULL.

{Language}: The LCID of the **locale** of the site.

{DirName}: The directory name of the requested document.

{UIVersion}: The display version number of the document being returned.

{ContentVersion}: The version number of the document stream being returned.

{RbsCollectionId}: The identifier for the remote blob storage collection for the site collection, or zero if remote blob storage is not configured for this database.

3.1.5.19.2 Domain Group Cache Versions Result Set

The Domain Group Cache Versions Result Set contains information about the version numbers associated with the Domain Group Map Caches on the front-end web server and on the back-end database server for the specified site collection.

The Domain Group Cache Versions Result Set MUST be returned if the specified document exists and MUST contain one row of version number data. If the specified *@DGCacheVersion* value is -2, then all columns returned MUST have the value -2, indicating that the value MUST NOT be used for comparison. The Domain Group Cache Versions Result Set is defined in Domain Group Cache Versions Result Set, section 2.2.5.4.

3.1.5.19.3 Domain Group Cache BEDS Update Result Set

The Domain Group Cache BEDS Update Result Set contains information to be used in recomputing the Domain Group Map Cache.

The Domain Group Cache BEDS Update Result Set MUST be returned if *@DGCacheVersion* is not -2 ("Skip") and the real Domain Group Map Cache version is more recent than the cached version on the back-end database server; that is, if the value of **RealVersion** is greater than the value of **CachedVersion** in the Domain Group Cache Versions Result Set (section 2.2.5.4).

If the Domain Group Cache BEDS Update Result Set is returned, it indicates that the copy of the Domain Group Map Cache on the back-end database server is out of date and MUST be recomputed to ensure that proper security checks can be made.

When returned, the Domain Group Cache BEDS Update Result Set MUST have a single row. The Domain Group Cache BEDS Update Result Set is defined in Domain Group Cache BEDS Update Result Set (section 2.2.5.3).

3.1.5.19.4 Domain Group Cache WFE Update Result Set

The Domain Group Cache WFE Update Result Set contains the binary data needed to refresh the domain group map cache.

The Domain Group Cache WFE Update Result Set MUST be returned only if *@DGCacheVersion* is not -2 ("Skip") and the cached version on the back-end database server is up-to-date; that is, if the value of **RealVersion** is not greater than the value of **CachedVersion** in the Domain Group Cache BEDS Update Result Set (section 3.1.5.19.3).

The Domain Group Cache WFE Update Result Set is defined in the Common Result Sets Domain Group Cache WFE Update Result Set (section 2.2.5.5).

3.1.5.19.5 User Information Result Set

The user Information Result Set returns information about the user specified in *@UserId*.

```
tp_Id                           int,
tp_SiteAdmin                    bit,
tp_IsActive                     bit,
tp_Login                        nvarchar(255),
tp_Email                        nvarchar(255),
tp_Title                        nvarchar(255),
tp_Notes                        nvarchar(1023),
tp_ExternalTokenLastUpdated     datetime,
tp_Token                        varbinary(max),
tp_Flags                           int,
UserId                          int,
SiteSecurityVersion             bigint;
```

tp_Id: The user identifier (section 2.2.1.13).

tp_SiteAdmin: Indicates whether the specified user is a site collection administrator on the site collection specified by *@DocSiteId*.

tp_IsActive: MUST be set to "1" if the specified user is an active user in the site collection specified by *@DocSiteId*.

tp_Login: The login name of the specified user.

tp_Email: The email address of the specified user.

tp_Title: The display name of the specified user.

tp_Notes: Notes about the specified user.

tp_ExternalTokenLastUpdated: The date and time, in **UTC** format, when the **External Group Token** (section 2.2.4.2) for the specified user was last updated.

tp_Token: A **WSS user Token** (section 2.2.4.10) value specifying the site **group** membership of the specified user.

tp_Flags: A **WSS user Flags** value for the specified user.

UserId: The site membership identifier of the specified user. This parameter can be NULL if the user has not been added as a **member** to the site whose permissions are in effect on the document.

SiteSecurityVersion: The current security information version of the site collection containing this document.

3.1.5.19.6 Welcome Page Redirect Information Result Set

The Welcome Page Redirect Information Result Set returns whether or not the document is a site or a folder with a configured welcome page, or the welcome page itself. The Welcome Page Redirect Information Result Set contains information about the welcome page.

```
{RedirectType}                  tinyint,
{RedirectUrl}                   nvarchar(260),
WelcomePageParameters           nvarchar(max),
{ContentTypeId}                 varbinary(512);
```

{RedirectType}: The **Redirect** type (section 2.2.3.15) of the URL. MUST be 0, indicating a welcome page redirect URL.

{RedirectUrl}: The URL of the welcome page.

WelcomePageParameters: This MUST contain any URL parameters configured for the welcome page. This value can contain a query string starting with "?" or a hash parameter starting with "#".

{ContentTypeId}: This value MUST be NULL.

3.1.5.19.7 Non-Welcome Page Redirect Information Result Set

The Non-Welcome Page Redirect Information Result Set returns if the document is a site or a folder, where a welcome page is not configured. Depending on the specified document URL, the redirect can be to a home page, a list view web page, or to a provisioning page URL. The Non-Welcome Page Redirect Information Result Set contains information about the redirect URL. The Non-Welcome Page Redirect Information Result Set MUST return a single row.

```
{RedirectType}                  tinyint,
{RedirectUrl}                   nvarchar(260),
{WelcomePageParameters}         nvarchar(max),
{ContentTypeId}                 varbinary(512);
```

{RedirectType}: The **Redirect** type (section 2.2.3.15) of the URL. This parameter MUST NOT be 0. For all other valid values, see **Redirect** type (section 2.2.3.15).

{RedirectUrl}: The full redirect URL.

{WelcomePageParameters}: MUST be NULL.

{ContentTypeId}: The document's content type identifier.

3.1.5.19.8 Document Content Stream Result Set

The Document Content Stream Result Set contains the document's binary **stream** and associated metadata. This row set MUST be returned only if *@FetchType* is NOT set to 1, otherwise, the row set MUST NOT be returned.

The Document Content Stream Result Set MUST return zero or one rows. If the document is modified, then a single row MUST be returned. A document is considered modified subject to the semantics indicated by the input parameter *@ValidationType*, or if its Virus Vendor ID has been updated since the client last retrieved it. When the document is modified , the return code 18 MUST be returned upon successful completion of **proc_FetchDocForHttpGet**. Otherwise, zero rows MUST be returned.

```
{Size}                          int,
{Content}                       varbinary(max),
{RbsResRefrence}                varbinary(800),
{Version}                       int,
{Id}                            uniqueidentifier;
{SetupPathVersion}              tinyint,
{SetupPath}                     nvarchar(255),
{Dirty}                         bit,
{DocFlags}                      int,
{IsHistoryversion}              bit
{Level}                         tinyint,
{InternalVersion}               int
```

{Size}: The size of the document stream, in bytes.

{Content}: The document's content stream. For a ghosted document, or for a document using external storage, this MUST be NULL. Otherwise, if the content is larger than the value specified in the *@ChunkSize* parameter, only the first *@ChunkSize* bytes MUST be returned, and the front-end web server can request individual chunks of content in a subsequent request.

{RbsResRefrence}: This value is used in remote blob storage and is opaque to the back-end database server. For additional information regarding remote blob storage, see [MS-WSSO] section 2.1.2.3.8.

{Version}: An integer value tracking the document's internal version number.

{Id}: The **document identifier** (section 2.2.1.2) of the requested document.

{SetupPathVersion}: For a ghosted document, this parameter governs the setup path location to which the **SetupPath** fragment is relative. This value MUST be NULL if the document does not exist, and it is undefined for a document that was never ghosted. The value MUST be one of the following.

Value	Description
2	Relative to the install location of Windows SharePoint Services 2.0 on the front-end web server (for example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\60).
3	Relative to the install location of Windows SharePoint Services 3.0 on the front-end web server (for example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\12).
4	Relative to the install location of Microsoft SharePoint Foundation 2010 on the front-end web server (for example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\14).

{SetupPath}: For a document that has ever been ghosted, this contains the setup path fragment relative to the base setup path where the content stream of this document can be found, as described in **{SetupPathVersion}**. Otherwise, this parameter MUST be NULL. This parameter also contains the setup user, which specifies the author of the ghosted document.

{Dirty}: This parameter MUST be set to 1 if this document has had dependency update processing performed; otherwise, it MUST be set to 0. If the document does not have a content stream, then the value is implementation-dependent and MUST be ignored.

{DocFlags}: A **Doc Flags** (section 2.2.2.3) value describing the document.

{IsHistoricalVersion}: This parameter MUST be set to 0

{Level}: A publishing level value specifying the publishing status of this document.

{InternalVersion}: An integer value tracking the document's internal version number.

3.1.5.19.9 Site Collection Audit Mask Result Set

The Site Collection Audit Mask Result Set contains the information about the **Audit Flags** (section 2.2.2.1) associated with the site collection containing the specified document.

The Site Collection Audit Mask Result Set is defined in Site Audit Mask Result Set, section 2.2.5.20.

3.1.5.19.10 List Audit Mask Result Set

The List Audit Mask Result Set contains information about the **Audit Flags** (section 2.2.2.1) associated with the list containing the document.

The List Audit Mask Result Set MUST return one row if the document is contained in a list; otherwise, zero rows MUST be returned.

```
tp_Id                          uniqueidentifier,
tp_AuditFlags                  int,
tp_InheritAuditFlags           int,
{GlobalAuditMask}              int,
{URL}                          nvarchar(516)
```

tp_Id: The list identifier (section 2.2.1.5) of the list containing the document.

tp_AuditFlags: An **Audit Flags** value determining the operations to be tracked on the list.

tp_InheritAuditFlags: An **Audit Flags** value determining the operations to be tracked on the document as inherited from the document's container.

{GlobalAuditMask}: An **Audit Flags** value determining the operations to be tracked across the site collection that contains the document.

{URL}: The URL of the list containing the document.

3.1.5.19.11 Document Build Dependency Set Result Set

The Document Build Dependency Set Result Set returns a build dependency set for a published document. The Document Build Dependency Set Result Set MUST return only when *@FetchBuildDependencySet* is set to 1 and when the document is published.

```
BuildDependencySet             varbinary(max);
```

BuildDependencySet: A binary array holding implementation-specific information about dependent documents. NULL indicates that there is no information about dependencies. A **BuildDependencySet** of size 0 indicates a document with no dependencies.

3.1.5.19.12 Document Build Dependency Metadata Result Set

The Document Build Dependency Metadata Result Set contains information about each document in the specified document's build dependency set. The Document Build Dependency Metadata Result Set MUST return only if *@FetchBuildDependencySet* is set to 1, and the binary array returned in the Document Build Dependency Set Result Set (section 3.1.5.19.11) is not NULL, and is not of zero-length.

```
DirName                        nvarchar(256),
LeafName                       nvarchar(128),
WebsFullUrl                    nvarchar(256),
FirstUniqueAncestorWebId       uniqueidentifier,
SecurityProvider               uniqueidentifier,
Acl                            varbinary(max),
AnonymousPermMask              bigint,
WebId                         uniqueidentifier,
ListId                        uniqueidentifier,
UniqueList                    bit,
InDocLibrary                  bit,
DraftOwnerId                  int,
ListFlags                     bigint,
ProgId                        nvarchar(255),
SetupPathVersion              tinyint,
SetupPath                     nvarchar(255),
TimeLastWritten               datetime,
MasterUrl                     nvarchar(260),
CustomMasterUrl               nvarchar(260),
Version                       int,
Id                            uniqueidentifier,
BuildDependencySet            varbinary(max);
```

DirName: The directory name of the document.

LeafName: The leaf name of the document.

WebsFullUrl: The store-relative form URL of the document.

FirstUniqueAncestorWebId: The site identifier (section 2.2.1.11) of the closest site in this site's ancestor chain that does not inherit security settings from its parent site.

SecurityProvider: COM CLSID of the **external security provider** for this site. This MUST be NULL for sites using the native security implementation.

Acl: The binary serialization of the **WSS ACL Format** (section 2.2.4.6) ACL for this site.

AnonymousPermMask: A **WSS Rights Mask** (section 2.2.2.14) that indicates the rights granted to a user that is anonymous, or has no specific rights, to the document.

WebId: The site identifier for a site containing the document.

ListId: The list identifier (section 2.2.1.5) for the list containing the document.

UniqueList: This value MUST be zero.

InDocLibrary: MUST be set to 1 if the document is contained within a document library. Otherwise, MUST be set to 0.

DraftOwnerId: The user identifier (section 2.2.1.13) for the owner of the last checked-in draft version of the document.

ListFlags: A **List Flags** (section 2.2.2.5) value describing the list that contains this document. This value MUST be NULL if the document is not stored in a list.

ProgId: Designates a preferred application to open this document. This value MUST be NULL if the document does not exist or the parser did not specify a ProgId when the document was saved.

SetupPathVersion: For a ghosted document, this governs the setup path location to which the **SetupPath** fragment is relative. This value MUST be NULL if the requested document does not exist and is undefined for documents that were never ghosted. The following are values are valid.

Value	Description
2	The **SetupPath** is relative to the install location of Windows SharePoint Services 2.0 on the front-end web server (for example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\60).

Value	Description
3	The **SetupPath** is relative to the install location of Windows SharePoint Services 3.0 on the front-end web server (for example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\12).
4	Relative to the install location of Microsoft SharePoint Foundation 2010 on the front-end web server (for example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\14).

SetupPath: For a document that is now or once was ghosted, this contains the setup path fragment relative to the base setup path described earlier by the **SetupPathVersion** value, where the content **stream** of this document can be found. This value MUST be NULL in the case of a document that does not exist or a document that was never ghosted.

TimeLastWritten: A time stamp, in **UTC** format, specifying when any changes were made to the document stream. Does not reflect changes made strictly to the metainfo or other item properties.

MasterUrl: The URL for the master page registered on the site for use in pages of the site when rendered on the front-end web server.

CustomMasterUrl: The URL for an alternate master page registered on the site for use in pages of the site rendered on the front-end web server.

Version: A counter incremented any time a change is made to this document.

Id: The **document identifier** (section 2.2.1.2).

BuildDependencySet: A **varbinary(max)** value specifying further document dependencies.

3.1.5.19.13 Site Metadata Result Set

The Site Metadata Result Set contains metadata for the site containing the specified document. The Site Metadata Result Set MUST return only if the input parameter *@PageView* is not NULL, as part of a series of result sets describing view web page document metadata.

The Site Metadata Result Set is defined in section 2.2.5.22.

3.1.5.19.14 Event Receivers Result Set

The Event Receivers Result Set contains information about the **event receivers** defined for the **site collection** containing the specified document.

The Event Receivers Result Set is part of a series of result sets describing view webpage-related metadata. It MUST return only if the input parameter *@PageView* is not NULL, and it will not return if **proc_FetchDocForHttpGet** returns a return code of 1271, indicating that the site collection was locked.

The Event Receivers Result Set MUST contain one row per event receiver registered for the site collection. The Event Receivers Result Set is defined in section 2.2.5.9.

3.1.5.19.15 Web Event Receiver Result Set

The Web Event Receivers Result Set contains information about the **event receivers** defined for the site containing the specified document.

The Web Event Receivers Result Set is part of a series of result sets describing view webpage-related metadata. It MUST return only if the input parameter *@PageView* is not NULL, and it will not return if **proc_FetchDocForHttpGet** returns a return code of 1271, indicating that the site was **locked**.

The Web Event Receivers Result Set MUST contain one row per event receiver registered for the site. The Web Event Receivers Result Set is defined in Event Receiver Result Set, section 2.2.5.9.

3.1.5.19.16 Site Features List Result Set

The Site Features List Result Set returns information about available features. If the Site Features List Result Set returns, it MUST return **twice:** first, for site collection features, and then for site features, for the site and site collection that contain the specified document.

The Site Features List Result Set MUST return only if *@PageView* is not NULL, and it will not return if **proc_FetchDocForHttpGet** returns a code of 1271, indicating that the site collection was locked.

The Site Feature List Result Set is defined in section 2.2.5.21.

3.1.5.19.17 WebParts Metadata, Personalized Result Set

The Web Parts Metadata, Personalized Result Set contains the core metadata about the Web parts appearing on the specified document, personalized for the user specified in *@UserId*.

The Web Parts Metadata, Personalized Result Set is part of a series of result sets describing view Web page-related metadata. It MUST return only if the input *@PageView* is NOT 0.

The Web Parts Metadata, Personalized Result Set will not return if **proc_FetchDocForHttpGet** returns a code of 1271, indicating that the site collection was locked.

The Web Parts Metadata, Personalized Result Set MUST contain one row per Web part.

```
tp_Id                          uniqueidentifier,
{tp_ZoneID}                    nvarchar(64),
tp_WebPartTypeId               uniqueidentifier,
tp_Assembly                    nvarchar(255),
tp_Class                       nvarchar(255),
tp_SolutionId                  uniqueidentifier,
Hash                           nvarchar(50),
ValidatorsHash                  nvarchar(64),
ValidationErrorUrl,             nvarchar(4000),
ValidationErrorMessage,         nvarchar(4000),
{tp_IsIncluded}                bit,
{tp_FrameState}                tinyint,
tp_AllUsersProperties          varbinary(max),
{tp_PerUsersProperties}        varbinary(max),
{tp_PartOrder}                 int,
{tp_Flags}                     int,
{AllUsers}                     int,
tp_Version                     int,
tp_Cache                       varbinary(max),
{Per_tp_Cache}                 varbinary(max),
{tp_ListId}                    nvarchar(38),
tp_Type                        tinyint,
tp_Source                      nvarchar(max),
Tp_BaseViewId                  int,
tp_View                        nvarchar(max);
```

tp_Id: The web part identifier (section 2.2.1.15) of the Web part. This value MUST NOT be NULL.

tp_ZoneId: The name of a Web part zone. This value can be NULL.

tp_WebPartTypeId: A 16-byte value uniquely identifying the type of the Web part. MUST NOT be NULL.

tp_Assembly: The assembly name of the implementation of the **Web Part**.

tp_Class: The **fully qualified class name** of the implementation of the Web part.

SolutionId: The solution identifier of the solution. MUST be NULL if there is no solution associated with the Web part.

Hash: The implementation-specific hash of the content of the sandboxed solution. This value MUST be NULL if there is no solution associated with the Web part.

ValidatorsHash: The implementation-specific hash of the sandboxed solution validators that validated the sandboxed solution. This value MUST be NULL if there is no solution associated with the Web part.

ValidationErrorUrl: The URL to render if the sandboxed solution validator failed validation. This value MUST be NULL if there is no solution associated with the Web part.

ValidationErrorMessage: The error message string to render if the sandboxed solution validator failed validation. This value MUST be NULL if there is no solution associated with the Web part.

{tp_IsIncluded}: If this bit flag is set to 1, it indicates that the Web part is visible. Otherwise, it MUST be 0.

{tp_FrameState}: A value that indicates the frame state of the Web part. This value MUST be one of the following.

Value	Description
0	Normal. The Web part is displayed in its normal state, with title, content, and placement within the page.
1	Minimized. The Web part is collapsed.

tp_AllUsersProperties: Web Part Properties specified for all users. This value can be NULL.

{tp_PerUsersProperties}: Web Part Properties specified for per-user basis. This value can be NULL.

{tp_PartOrder}: Ordinal number that indicates the location of the Web part in relation to other Web parts in the same zone. This value can be NULL.

{tp_Flags}: A **View Flags** (section 2.2.2.12) value that specifies view-related settings for this Web part. This value can be equal to 0.

{AllUsers}: A flag that indicates whether customization or personalization is in effect on this Web parts page. This value MUST be one of the following.

Value	Description
1	User not specified; changes apply to all users of this Web part instance. Also indicates customization mode.
2	Personalization is in effect; changes apply to the **tp_UserId** in the Personalization table.
3	Personalization is in effect; changes apply to the **tp_UserId** in the Web Parts table.

tp_Version: A counter incremented any time a change is made to the schema or other properties of this list and used for internal conflict detection.

tp_Cache: Private data cache for the Web part. This value can be NULL.

{Per_tp_Cache}: Contains the private data cache for documents with a checkout level of "published". Otherwise, this MUST be NULL.

{tp_ListId}: The list identifier (section 2.2.1.5) of the list to which this Web part refers, enclosed in braces. If not referencing a list, then this value MUST be NULL.

tp_Type: The **Page** type (section 2.2.3.14) of this Web part. This value can be NULL.

tp_Source: Properties of the Web part, as specified by a compatible HTML editing application. This value can be NULL.

tp_BaseViewId: The ID of the base view for the list view web part. This value can be NULL.

tp_View: Contains implementation-specific XML used when processing this Web part. If this Web part is not a view, then this MUST be NULL.

3.1.5.19.18 Web Parts Metadata, Nonpersonalized Result Set

The Web Parts Metadata, Nonpersonalized Result Set contains the core metadata about the Web parts appearing on the specified document.

The Web Parts Metadata, Nonpersonalized Result Set is part of a series of result sets describing view Web page-related metadata. It MUST return only if the input *@PageView* is 0.

The Web Parts Metadata, Nonpersonalized Result Set will not return if **proc_FetchDocForHttpGet** returns a code of 1271, indicating that the site collection was locked.

The Web Parts Metadata, Nonpersonalized Result Set MUST contain one row per Web part.

```
tp_Id                          uniqueidentifier,
tp_ZoneID                      nvarchar(64),
tp_WebPartTypeId               uniqueidentifier,
tp_Assembly                    nvarchar(255),
tp_Class                       nvarchar(255),
tp_SolutionId                  uniqueidentifier,
Hash                           nvarchar(50),
ValidatorsHash                  nvarchar(64),
ValidationErrorUrl,             nvarchar(4000),
ValidationErrorMessage,         nvarchar(4000),
tp_IsIncluded                  bit,
tp_FrameState                  tinyint,
tp_AllUsersProperties          varbinary(max),
tp_PerUsersProperties          varbinary(max),
tp_PartOrder                   int,
{tp_Flags}                     int,
{AllUsers}                     int,
tp_Version                     int,
tp_Cache                       varbinary(max),
{Per_tp_Cache}                 varbinary(max),
{tp_ListId}                    nvarchar(38),
tp_Type                        tinyint,
tp_Source                      nvarchar(max),
tp_BaseViewId                  int,
tp_View                        nvarchar(max);
```

tp_Id: The web part identifier (section 2.2.1.15) of the Web part. This value MUST NOT be NULL.

tp_ZoneId: The name of a Web part zone. This value can be NULL.

tp_WebPartTypeId: A 16-byte value uniquely identifying the type of the Web part. MUST NOT be NULL.

tp_Assembly: The assembly name of the implementation of the Web Part.

tp_Class: The fully qualified class name of the implementation of the Web Part.

SolutionId: The solution identifier of the solution. MUST be NULL if there is no solution associated with the Web part.

Hash: The implementation-specific hash of the contents of the sandboxed solution. This value MUST NOT be NULL.

ValidatorsHash: The implementation-specific hash of the sandboxed solution validators that validated the sandboxed solution. This value MUST NOT be NULL.

ValidationErrorUrl: The URL to render if the sandboxed solution validator failed validation. This value MUST be NULL if there is no solution associated with the Web part.

ValidationErrorMessage: The error message string to render if the sandboxed solution validator failed validation. This value MUST be NULL if there is no solution associated with the Web part.

tp_IsIncluded: If 1, this indicates that the Web part is visible. MUST NOT be NULL.

tp_FrameState: A value that indicates the frame state of the Web part. This value MUST be one of the following.

Value	Description
0	Normal. The Web part is displayed in its normal state, with title, content, and placement within the page.
1	Minimized. The Web part is collapsed.

tp_AllUsersProperties: Properties specified for all users. This value can be NULL.

tp_PerUsersProperties: Properties specified for per-user basis. This value can be NULL.

tp_PartOrder: Ordinal number that indicates the location of the Web part in relation to other Web parts in the same zone. This value can be NULL.

{tp_Flags}: A **View Flags** (section 2.2.2.12) value that specifies view-related settings for this Web part. This value MUST NOT be NULL.

{AllUsers}: A flag that indicates whether customization or personalization is in effect on this Web parts page. This value MUST be equal to 1, indicating customization.

tp_Version: A counter incremented any time a change is made to the schema or other properties of this list and used for internal conflict detection.

tp_Cache: Private data cache for the Web part. This value can be NULL.

{Per_tp_Cache}: Private data cache for published documents. This value MUST be NULL.

{tp_ListId}: The list identifier (section 2.2.1.5) of the list to which this Web part refers, enclosed in braces. If not referencing a list, this value MUST be NULL.

tp_Type: The **Page** type (section 2.2.3.14) of this Web part. This value can be NULL.

tp_Source: Properties of the Web part as specified by a compatible HTML editing application. This value can be NULL.

tp_BaseViewId: The ID of the base view for the list view Web part, the value can be NULL.

tp_View: An **nvarchar(max)** value containing implementation-specific XML used when processing this Web part. If this Web part is not a view, this MUST be NULL.

3.1.5.19.19 List Metadata Result Set

The List Metadata Result Set contains the metadata for the lists associated with the Web parts that are included on the specified document.

The List Metadata Result Set returns only if there are such Web parts (at least one row returns in the previously returned result set: Web Parts Metadata, Personalized (section 3.1.5.19.17), or Web Parts Metadata, Nonpersonalized (section 3.1.5.19.18)).

The List Metadata Result Set is part of a series of result sets describing view Web page-related metadata. It MUST be returned only if the input parameter *@PageView* is not NULL, and it MUST NOT be returned if **proc_FetchDocForHttpGet** returns a code of 1271, indicating that the site collection was locked.

The List Metadata Result Set MUST return one row for each associated list. The List Metadata Result Set is defined in List Metadata Result Set, section 2.2.5.12.

3.1.5.19.20 List Event Receivers Result Set

The List Event Receivers Result Set contains the event receivers registered on the lists associated with the Web parts that appear on the specified document.

The List Event Receivers Result Set MUST return only if there are such lists (the List Metadata Result Set (section 3.1.5.19.19) returns with at least one row.)

The List Event Receivers Result Set MUST contain one row per event receiver registered. The List Event Receivers Result Set can be empty. The List Event Receivers Result Set is defined in Event Receivers Result Set, section 2.2.5.9.

3.1.5.19.21 List Security Information Result Set

The List Security Information Result Set contains permissions information for the lists associated with the Web parts that appear on the specified document.

The List Security Information Result Set MUST return only if requested (input parameter *@PrefetchListScope* is set to 1) and if such lists exist (the List Metadata Result Set (section 3.1.5.19.19) returns with at least one row).

The List Security Information Result Set returns one row per each unique scope associated with the lists.

```
ListId                          uniqueidentifier,
ScopeId                         uniqueidentifier,
Acl                             varbinary(max),
AnonymousPermMask               bigint;
```

ListId: The list identifier (section 2.2.1.5) of the list.

ScopeId: The scope identifier (section 2.2.1.8) of the Permissions Scope that applies to the list. This MUST be set to zero if the list does not have Fine grain permission.

Acl: Contains the ACL of the Permissions Scope that applies to the list. This MUST be NULL if **ScopeId** is zero.

AnonymousPermMask: Contains the **WSS Rights Mask** (section 2.2.2.14) on the list in effect for anonymous users. This MUST be 0 if **ScopeId** is zero.

3.1.5.19.22 SiteCollection Custom Action Result Set

The Site Collection Custom Actions Result Set MUST return 1 row for each custom action defined for the site collection containing the specified document.

The Site Collection Custom Actions Result Set is part of a series of result sets describing view webpage-related metadata. It MUST return only if the input parameter *@PageView* is not NULL, and it will not return if **proc_FetchDocForHttpGet** returns a return code of 1271, signifying that the site collection was locked.

If there are no custom actions defined for the site collection, this result set MUST NOT return any rows. This result set is defined in Custom Actions From Scope Result Set, section 2.2.5.2.

3.1.5.19.23 Site Custom Actions Result Set

The Site Custom Actions Result Set MUST return 1 row for each custom action defined for the site containing the specified document.

The Site Custom Actions Result Set is part of a series of result sets describing view webpage-related metadata. It MUST return only if the input parameter *@PageView* is not NULL, and it will not return if **proc_FetchDocForHttpGet** returns a return code of 1271, signifying that the site collection was locked.

If there are no custom actions defined for the site, this result set MUST NOT return any rows. This result set is defined in Custom Actions From Scope Result Set, section 2.2.5.2.

3.1.5.19.24 List Custom Actions Result Set

The List Custom Action Result Set contains information about the custom actions related to the lists associated with the specified document.

The List Custom Actions Result Set is part of a series of result sets describing view webpage-related metadata. It MUST return only if the input parameter *@PageView* is not NULL, and it will not return if **proc_FetchDocForHttpGet** returns a return code of 1271, signifying that the site collection was locked.

The List Custom Actions Result Set MUST return only if such lists exist (the List Metadata Result Set (section 2.2.5.12) returns with at least one row).

If there are no custom actions defined for the list, this result set MUST NOT return any rows. This result set is defined in Custom Actions From Scope Result Set, section 2.2.5.2.

3.1.5.19.25 List Web Parts Result Set

The List Web Parts Result Set contains information about the Web parts related to the lists associated with the specified document.

The List Web Parts Result Set MUST return only if such lists exist (the List Metadata Result Set (section 2.2.5.12) returns with at least one row).

The List Web Parts Result Set MUST contain one row per Web part registered for each list. The List Web Parts Result Set can be empty. The List Web Parts Result Set is defined in List Web Parts Result Set, section 2.2.5.13.

3.1.5.19.26 Content Type Order Result Set

The Content Type Order Result Set provides the information necessary for the implementation-specific rendering of a list view web page. The information necessary to render the content types in the configured order, if such exists, is provided in a binary array value.

The Content Type Order Result Set MUST return only if *@PageView* is not NULL, and it will not return if **proc_FetchDocForHttpGet** returns a code of 1271, indicating that the site collection was locked.

```
        {CurrentFolderURL}              varchar,
        {MetaInfo}                    varbinary(max)
```

{CurrentFolderURL}: The URL of the current folder as specified in the *@CurrentFolderURL* input parameter.

{MetaInfo}: The information about content type order, embedded in a binary array. If no information is available for the current folder, or if there is no current folder specified, or if the document is not contained in a list, then this parameter MUST be NULL and the column is unnamed. Otherwise, the column MUST be named "MetaInfo".

3.1.5.19.27 Current Folder Scope Result Set

The Current Folder Scope Result set contains scope information about the current folder as specified by the input parameter *@CurrentFolderURL*. It is used by the front-end web server to security trim the information rendered to the user in a view webpage based on the user's access.

The Current Folder Scope Result Set MUST return only if *@PageView* is not NULL, and it MUST NOT return if **proc_FetchDocForHttpGet** returns a code of 1271, indicating that the site collection was locked. In addition, the Current Folder Scope Result Set MUST return only if the document is contained in a list and resides in a folder that is not the list's root folder.

If the Current Folder Scope Result Set returns, it MUST return one row.

```
        {FolderScopeId}               uniqueidentifier,
        {FolderId}                    int;
```

{FolderScopeId}: The scope identifier (section 2.2.1.8) of the security scope effective for the specified folder.

{FolderId}: The integer identifier of the folder document within the containing list.

3.1.5.19.28 Navigation Context Security Information Result Set

The Navigation Context Security Information Result Set contains security information about the site containing the specified document and about all sites in its navigation hierarchy.

The Navigation Context Security Information Result Set MUST return only upon successful execution, if *@PageView* is not NULL and the information is not larger than 1,800 bytes.

If the Navigation Context Security Information Result Set returns, it MUST return one row for each unique scope in the site's navigation hierarchy, excluding the site's own scope.

The Navigation Context Security Information Result Set is defined in Unique Permissions Result Set, section 2.2.5.24.

3.1.5.19.29 NULL Navigation Context Security Information Result Set

The NULL Navigation Context Security Information Result Set, with NULL values in three unnamed columns, returns to indicate that the navigation context security information is larger than 1,800 bytes.

The NULL Navigation Context Security Information Result Set MUST return only upon successful execution if *@PageView* is not NULL and the security information about the site or the parent site in the site's navigation hierarchy was larger than 1,800 bytes.

If the NULL Navigation Context Security Information Result Set returns, it MUST return one row, as defined in NULL Unique Permissions Result Set, section 2.2.5.15.

3.1.5.19.30 Empty Navigation Context Security Information Result Set

The Empty Navigation Context Security Information Result Set, holding zero rows with a single, unnamed NULL column, is returned to indicate that the navigation context security information is not available or is not up-to-date. The Empty Navigation Context Security Information Result Set MUST return only upon successful execution, and if neither the Navigation Context Security Information (section 3.1.5.19.28) nor the NULL Navigation Context Security Information (section 3.1.5.19.29) result sets return.

If the Empty Navigation Context Security Information Result Set returns, it MUST return zero rows, as defined using T-SQL syntax in Empty Result Set, section 2.2.5.8.

3.1.5.20 proc_FetchDocForRead

The **proc_FetchDocForRead** stored procedure is invoked to request the metadata information and document stream of a document.

```
PROCEDURE proc_FetchDocForRead(
      @DocSiteId                      uniqueidentifier,
      @DocWebId                       uniqueidentifier,
      @DocDirName                     nvarchar(256),
      @DocLeafName                    nvarchar(128),
      @DocFullUrl                     nvarchar(260),
      @LooksLikeAttachmentFile        bit,
      @GetContent                     bit,
      @GetWebListForNormalization     bit,
      @bGetContainingList             bit,
      @bCheckout                      bit,
      @GetCurrentMetaInfo             bit,
      @GetLinkInfo                    bit,
      @UserId                         int,
      @Version                        int,
      @ChunkSize                      int,
      @MaxLevel                       tinyint,
      @Level                          tinyint OUTPUT,
      @UIVersion                      int OUTPUT,
      @RequestGuid                    uniqueidentifier = NULL OUTPUT
);
```

@DocSiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the requested document.

@DocWebId: The site identifier (section 2.2.1.11) of the site containing the requested document.

@DocDirName: The directory name of the requested document.

@DocLeafName: The leaf name of the requested document.

@DocFullUrl: This parameter MUST be NULL and MUST be ignored.

@LooksLikeAttachmentFile: Specifies whether the requested document appears to the front-end web server to be an attachment to a list item. If this flag is set to 1, then this stored procedure MUST determine whether the current user has sufficient permissions to retrieve the requested document as an attachment. In addition, if this flag is set to 1, then the **NeedManageListRight** column of the Attachment State Result Set (section 3.1.5.20.12) will indicate if the **ManageLists** bit of the **WSS Rights Mask** (section 2.2.2.14) is set.

@GetContent: A bit flag specifying whether to return the document stream of the document or not. If this flag is set to 1, then this stored procedure MUST return the document stream of the requested document in the Document Information and Content Result Set (section 3.1.5.20.10) or the Document Version Information and Content Result Set (section 3.1.5.20.11).

@GetWebListForNormalization: A bit flag specifying whether to return the subsite of the site specified by *@DocWebId*. If this flag is set to 1, then this stored procedure MUST return a list of subsites in the Subsite List Result Set (section 3.1.5.20.1).

@bGetContainingList: Specifies whether to return the event receivers information about the list containing the requested document. If this flag is set to 1, then this stored procedure MUST return the Event Receivers Result Set (section 3.1.5.20.6).

@bCheckout: Specifies whether the current user is requesting to **check out** the document. If this flag is set to 1, and the *@LooksLikeAttachmentFile* parameter is set to 1, then this stored procedure MUST determine whether the current user has sufficient permissions to check out the requested document as an attachment. The result of this is returned by the Attachment State Result Set.

@GetCurrentMetaInfo: If this parameter is set to 1, then the Document Version Metadata Result Set (section 3.1.5.20.4) MUST return the document's metainfo in the **{MetaInfo}** column. If this parameter is not set to 1, then the result set MUST return NULL as the document's metainfo in the **{MetaInfo}** column.

@GetLinkInfo: Specifies whether to return the link information about the requested document. If this flag is set to 1, then this stored procedure MUST return the Link Information Result Set (section 2.2.5.11).

@UserId: The user identifier (section 2.2.1.13) for the current user requesting the information.

@Version: Specifies the user interface (UI) version number of the document that is being requested. A value of -1 specifies the most recent version of the document.

@ChunkSize: Specifies the maximum size, in bytes, of the document content to be returned in the Document Information and Content Result Set. For a ghosted document, or for a document using external storage, this MUST be NULL. Otherwise, if the content is larger than the value specified in the *@ChunkSize* parameter, only the first *@ChunkSize* bytes MUST be returned, and the front-end web server can request individual chunks of content in a subsequent request.

@MaxLevel: A **Publishing Level** type (section 2.2.2.6) value that indicates the maximum publishing level of the document to be returned in the *@Level* parameter if multiple levels of the document are available and the current user is not the **owner** of the draft or does not have the document checked out.

@Level: The **Publishing Level** type value of the requested version of the document visible to the current user, returned as an output parameter. This value MUST be returned as NULL if the document does not exist.

@UIVersion: Output parameter. The user interface version number associated with the document. This value MUST be NULL in the case of a document that does not exist, or if the *@GetContent* flag is not 1.

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

Return Values: This stored procedure returns an integer return code that MUST be in the following table.

Value	Description
0	Successful execution.
3	The document does not exist.

This stored procedure returns a **Publishing Level** type value in the parameters as described earlier. This stored procedure MUST return multiple result sets. Some of the result sets MUST NOT be returned depending upon input parameters or calculations performed in this stored procedure, and all result sets that are returned MUST be sent in the order defined as follows.

3.1.5.20.1 Subsite List Result Set

The Subsite List Result Set contains an unordered list of store-relative form URLs for all subsites within the site collection whose parent site is specified in the *@DocWebId* parameter.

The Subsite List Result Set MUST only be returned if *@GetWebListForNormalization* is set to "1". The Subsite List Result Set MUST contain one row for each subsite with the specified parent site, and it MUST contain no rows if there are no such subsites.

The Subsite List Result Set is defined in URL Result Set (section 2.2.5.25).

3.1.5.20.2 Link Info Single Doc Result Set

The Link Info Single Doc Result Set contains information about links to or within the requested document. Entries are present both for forward links and for backward links. The Link Info Single Doc Result Set MUST only be returned if *@GetLinkInfo* is set to "1", and MUST contain one row for each link that is referenced.

The Link Info Single Doc Result Set is defined in Link Information Result Set, section 2.2.5.11.

3.1.5.20.3 Document Metadata Result Set

The Document Metadata Result Set contains the metadata about the most current version of the requested document visible to the current user.

The Document Metadata Result Set MUST only be returned if *@Version* is negative. There MUST be one row in the Document Metadata Result Set if the document exists; otherwise, there MUST be no rows.

The Document Metadata Result Set is defined in Document Metadata Result Set, section 2.2.5.6.

3.1.5.20.4 Document Version Metadata Result Set

The Document Version Metadata Result Set contains the metadata about the requested version of the document.

The Document Version Metadata Result Set MUST only be returned if *@Version* is not negative. If the document exists, the Document Version Metadata Result Set MUST contain one row; otherwise, it MUST contain zero rows.

The Document Version Metadata Result Set is defined in Document Version Metadata Result Set, section 2.2.5.7.

3.1.5.20.5 NULL Result Set

The NULL Result Set is a placeholder that returns no data.

The NULL Result Set MUST only be returned if *@Version* is negative, the requested document exists, and *@DocWebId* is NULL. The NULL Result Set MUST contain zero rows in a schema containing a single unnamed column.

3.1.5.20.6 Event Receivers Result Set (1)

The Event Receivers Result Set contains information about the event receivers defined for this document.

The Event Receivers Result Set MUST only be returned if one of these conditions is **true:**

- The requested document exists and either *@Version* is not negative, OR
- @Version is negative and *@DocWebId* is not NULL

The Event Receivers Result Set MUST contain one row for each event receiver registered with an **Event Host** type (section 2.2.3.5) of 3 (list item) for this document.

The Event Receivers Result Set is defined in Event Receivers Result Set, section 2.2.5.9.

3.1.5.20.7 List Metadata Result Set

The List Metadata Result Set contains the metadata associated with the list that contains the requested document.

The List Metadata Result Set MUST only be returned if the *@bGetContainingList* parameter is set to "1" and the document is contained within a list. The List Metadata Result Set MUST contain one row.

The List Metadata Result Set is defined in List Metadata Result Set, section 2.2.5.12.

3.1.5.20.8 Empty List Result Set

The Empty List Result Set contains a single, unnamed column with no rows to indicate that the document is not contained within a list. The Empty List Result Set MUST only be returned if the *@bGetContainingList* parameter is set to "1" and the document is not contained within a list.

The Empty List Result Set is defined in Empty Result Set, section 2.2.5.8.

3.1.5.20.9 Event Receivers Result Set (2)

The Event Receivers Result Set contains the event receivers associated with the list that contains the requested document.

The Event Receivers Result Set MUST only be returned if the *@bGetContainingList* parameter is set to "1" and the document is contained within a list. The Event Receivers Result Set MUST contain one row for each event receiver registered with an **Event Host** type (section 2.2.3.5) of 2 (List) for the List.

The Event Receivers Result Set is defined in Event Receivers Result Set, section 2.2.5.9.

3.1.5.20.10 Document Information and Content (Read) Result Set

The Document Information and Content (Read) Result Set contains information about the content of the document stream for the requested document.

The Document Information and Content (Read) Result Set MUST only be returned if *@GetContent* is set to "1" and *@Version* is negative. If the document exists, the Document Information and Content (Read) Result Set MUST contain one row; otherwise, it MUST contain zero rows.

The Document Information and Content (Read) Result Set is similar to the Document Information and Content (Update) Result Set (section 3.1.5.21.14), except for column naming.

```
{Size}                          int,
{Content}                       varbinary(MAX),
{RbsId}                         varbinary(800),
ETagVersion                     int,
Id                              uniqueidentifier,
SetupPathVersion                tinyint,
SetupPath                       nvarchar(255),
Dirty                           bit,
DocFlags                        int,
{IsHistoricalVersion}          bit,
Level                          tinyint,
InternalVersion                int,
DoclibRowId                    int,
VirusVendorID                  int,
VirusStatus                    int,
VirusInfo                      nvarchar(255)
```

{Size}: The size of the requested document in bytes.

{Content}: The document stream content of the document. For a ghosted document, content MUST be NULL. If the content is larger than the value specified in the *@ChunkSize* parameter, then the amount of data returned from the document stream will be equal to the number of bytes specified by the *@ChunkSize* parameter starting at the first byte of the document stream. The front-end web server can then request individual chunks of content in subsequent requests.

{RbsId}: If remote blob storage is enabled and the document's content is contained in a remote data store, then this MUST be the remote blob storage identifier for the document's content. If remote blob storage is disabled or the document's content is not contained in a remote data store, then this MUST contain NULL. For additional information regarding remote blob storage, see [MS-WSSO] section 2.1.2.3.8.

ETagVersion: An internal version counter incremented any time a change is made to this document, used for internal conflict detection. This value MUST be set to the current **ETagVersion** value for the document in the **Docs View** (section 2.2.7.4), except in the Document Version Information and Content Result Set (section 3.1.5.20.11), where this value MUST be NULL.

Id: The **document identifier** (section 2.2.1.2) of this document.

SetupPathVersion: For a ghosted document, this governs the setup path location to which the **SetupPath** fragment is relative. This value MUST be NULL if the document does not exist and is undefined for a document that was never ghosted.<11>

SetupPath: For a document that is now or once was ghosted, **SetupPath** MUST contain the setup path fragment relative to the base setup path described by the **SetupPathVersion** value, where the content **stream** of this document can be found. This value MUST be NULL in the case of a document that does not exist or a document that was never ghosted.

Dirty: A bit set to "1" to indicate that this document MUST have dependency update processing performed before its document stream is returned to an external agent. This field's value is implementation-dependent if this document does not have a document stream and MUST be ignored if the document does not have a document stream.

DocFlags: A **Doc Flags** (section 2.2.2.3) value describing the document. This value MUST be NULL if the document does not exist.

{IsHistoricalVersion}: MUST be 0.

Level: The **Publishing Level** type (section 2.2.2.6) value of the requested version of the document.

InternalVersion: An integer value specifying the implementation-related internal version number of this version of the document.

DocLibRowId: The identifier of the row in the document library that represents this document. This value MUST be NULL if the document is not stored in a list.

VirusVendorId: The identifier of the virus scanner that processed this document. This value MUST be NULL if the document does not exist or if this document has not been processed by a virus scanner.

VirusStatus: A **Virus Status** (section 2.2.3.18) value specifying the current virus scan status of this document. This value MUST be NULL if the document does not exist or if this document has not been processed by a virus scanner.

VirusInfo: A string containing a provider-specific message returned by the virus scanner when it last processed the document. This value MUST be NULL if the document does not exist or if the document has not been processed by a virus scanner.

3.1.5.20.11 Document Version Information and Content Result Set

The Document Version Information and Content Result Set contains information about the content of the document stream for the requested document.

The Document Version Information and Content Result Set MUST only be returned if *@GetContent* is set to "1" and *@Version* is not negative.

The Document Version Information and Content Result Set MUST contain one row if the specified document exists; otherwise, it MUST contain no rows.

```
{Size}                        int,
{Content}                     varbinary(max),
{RbsId}                      varbinary(800),
Version                       int,
Id                            uniqueidentifier,
SetupPathVersion              tinyint,
SetupPath                     nvarchar(255),
DocFlags                      int,
{IsHistoricalVersion}         bit,
{Level}                       tinyint,
DoclibRowId                   int,
VirusVendorID                 int,
VirusStatus                   int,
VirusInfo                     nvarchar(255)
```

{Size}: The size of the requested document in bytes.

{Content}: The document stream content of the document. For a ghosted document, content MUST be NULL. If the content is larger than the value specified in the *@ChunkSize* parameter, then the number of bytes specified by the *@ChunkSize* parameter starting at the first byte of the document stream MUST be returned. The front-end web server can then request individual chunks of content in subsequent requests.

{RbsId}: If remote blob storage is enabled and the document's content is contained in a remote data store, then this MUST be the remote blob storage identifier for the document's content. If remote blob storage is disabled or the document's content is not contained in a remote data store, then this MUST contain NULL. For additional information regarding remote blob storage, see [MS-WSSO] section 2.1.2.3.8.

Version: An internal version counter incremented any time a change is made to this document, used for internal conflict detection. This value MUST be set to the current internal version counter value for the document in the **Docs View** (section 2.2.7.4), except in the Document Version Information and Content Result Set, where this value MUST be NULL.

Id: The **document identifier** (section 2.2.1.2) of this document.

SetupPathVersion: For a ghosted document, this governs the setup path location to which the **SetupPath** fragment is relative. This value MUST be NULL if the document does not exist and is undefined for a document that was never ghosted.<12>

SetupPath: For a document that is now or once was ghosted, **SetupPath** MUST contain the setup path fragment relative to the base setup path described by the **SetupPathVersion** value, where the content **stream** of this document can be found. This value MUST be NULL in the case of a document that does not exist or a document that was never ghosted.

DocFlags: A **Doc Flags** (section 2.2.2.3) value describing the document. This value MUST be NULL if the document does not exist.

{IsHistoricalVersion}: MUST be 1.

{Level}: MUST be 1.

DocLibRowId: The identifier of the row in the document library that represents this document. This value MUST be NULL if the document is not stored in a list.

VirusVendorId: The identifier of the virus scanner that processed this document. This value MUST be NULL if the document does not exist or if this document has not been processed by a virus scanner.

VirusStatus: A **Virus Status** (section 2.2.3.18) value specifying the current virus scan status of this document. This value MUST be NULL if the document does not exist or if this document has not been processed by a virus scanner.

VirusInfo: A string containing a provider-specific message returned by the virus scanner when it last processed the document. This value MUST be NULL if the document does not exist or if the document has not been processed by a virus scanner.

3.1.5.20.12 Attachment State Result Set

The Attachment State Result Set contains the information about the attachment state of the requested document. The Attachment State Result Set MUST be returned and MUST contain one row.

The Attachment State Result Set is defined using T-SQL syntax, as follows.

```
    {IsAttachment}                bit,
    {NeedManageListRight}         bit,
    {Level}                       tinyint,
```

{IsAttachment}: This specifies whether the document is associated with an attachment. This value MUST be 1 if the document is an attachment, a list item attachment folder, or the list attachment folder itself. Otherwise, this value MUST be 0. See **Attachments Flag** (section 2.2.3.1) for more information.

{NeedManageListRight}: This specifies whether operations on this document require the user to have the ManageLists bit of the **WSS Rights Mask** (section 2.2.2.14) set. This value MUST be 0 if the document is not stored in a list.

{Level}: A **Publishing Level** type (section 2.2.2.6) value specifying the publishing status of the document. This value MUST be NULL if the document does not exist.

3.1.5.20.13 Audit Mask Result Set

The Audit Mask Result Set contains the information about the **Audit Flags** (section 2.2.2.1) associated with this document. The Audit Mask Result Set MUST be returned and MUST contain a single row of data if the document exists.

The Audit Mask Result Set is defined in Site Audit Mask Result Set, section 2.2.5.20.

3.1.5.21 proc_FetchDocForUpdate

The **proc_FetchDocForUpdate** stored procedure requests document content and metadata information. It also updates the **CacheParseId** flag for the requested document if the *@CacheParse* parameter is set to "1".

```
PROCEDURE proc_FetchDocForUpdate(
      @DocSiteId                      uniqueidentifier,
      @DocWebId                       uniqueidentifier,
      @DocDirName                     nvarchar(256),
      @DocLeafName                    nvarchar(128),
      @UserId                         int,
      @Version                        int,
      @GetContent                     bit,
      @GetWebListForNormalization     bit,
      @CacheParse                     bit,
      @GetWebPartInfo                 bit,
      @GetFileFormatInfo              bit,
      @bGetContainingList             bit,
      @GetCurrentMetaInfo             bit,
      @LooksLikeAttachmentFile        bit,
      @ChunkSize                      int,
      @Level                          tinyint          OUTPUT,
      @RequestGuid                    uniqueidentifier = NULL OUTPUT
);
```

@DocSiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the requested document.

@DocWebId: The site identifier (section 2.2.1.11) of the site containing the requested document.

@DocDirName: The directory name of the requested document.

@DocLeafName: The leaf name of the requested document.

@UserId: The user identifier (section 2.2.1.13) for the current user requesting the information.

@Version: Specifies the user interface (UI) version number of the document being requested. A value of "-1" specifies the most recent version of the document.

@GetContent: A bit flag specifying whether or not to return the document stream of the document. This flag MUST be set to "1" to return either the Document Information and Content (Update) Result Set (section 3.1.5.21.14), the Document Version 1 Information and Content Result Set (section 3.1.5.21.15), or the Document Version 2 Information and Content Result Set (section 3.1.5.21.16).

@GetWebListForNormalization: A bit flag specifying whether to return the subsite of the site specified by *@DocWebId*. If this flag is set to "1", then **proc_FetchDocForUpdate** MUST return a list of child sites in the Subsite List Result Set (section 3.1.5.21.1).

@CacheParse: A bit flag specifying whether to update the **CacheParseId** flag of the requested document. If this flag is set to "1", then **proc_FetchDocForUpdate** MUST update the **CacheParseId** flag in the **Docs View** (section 2.2.7.4) for the requested document.

@GetWebPartInfo: A bit flag specifying whether to return the Web parts information for the requested document. If this flag is set to "1", then **proc_FetchDocForUpdate** MUST return the Web parts information for the requested document in the Web Part Info Result Set (section 3.1.5.21.8) and the Zone ID Result Set (section 3.1.5.21.9).

@GetFileFormatInfo: If this parameter is set to NULL or 1, then the File Format Metadata Result Set (section 3.1.5.21.10) MUST be returned. Otherwise, the File Format Metadata Result Set MUST NOT be returned.

@bGetContainingList: Specifies whether to return the event receivers information about the list containing the requested document. If this flag is set to "1", then **proc_FetchDocForUpdate** MUST return the Event Receivers Result Set (section 3.1.5.21.13) for the list.

@GetCurrentMetaInfo: If this parameter is set to 1, then the Document Version Metadata Result Set (section 3.1.5.21.4) MUST return the document's metadata in the **{MetaInfo}** column. If this parameter is not set to 1, then the result set MUST return NULL as the document's metadata in the **{MetaInfo}** column.

@LooksLikeAttachmentFile: Specifies whether the requested document appears to the front-end Web server to be an attachment to a list item. If this flag is set to 1, then this stored procedure MUST determine whether the current user has sufficient permissions to retrieve the requested document as an attachment. In addition, if this flag is set to 1, then the **NeedManageListRight** column of the Attachment State Result Set (section 3.1.5.21.17) will indicate if the **ManageLists** bit of the **WSS Rights Mask** (section 2.2.2.14) is set.

@ChunkSize: Specifies the maximum size, in bytes, of the document stream to be returned in the Document Information and Content (Update) Result Set. For a ghosted document, or for a document using external storage, this MUST be NULL. Otherwise, if the content is larger than the value specified in the *@ChunkSize* parameter, only the first *@ChunkSize* bytes MUST be returned, and the front-end Web server can request individual chunks of content in a subsequent request.

@Level: The **Publishing Level** type (section 2.2.2.6) value of the requested version of the document visible to the current user, returned as an output parameter. This value MUST be returned as NULL if the document does not exist.

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

Return Values: The **proc_FetchDocForUpdate** stored procedure MUST return an integer return code of 0.

The **proc_FetchDocForUpdate** stored procedure returns a **Publishing Level** type value in the parameters as specified earlier. **proc_FetchDocForUpdate** MUST return multiple result sets. Some of the result sets MUST NOT be returned, depending upon input parameters or calculations performed in **proc_FetchDocForUpdate**, and all result sets that are returned MUST be sent in the order defined as follows.

3.1.5.21.1 Subsite List Result Set

The Subsite List Result Set contains an unordered list of store-relative form URLs for all subsites within the site collection whose parent Site is specified in the *@DocWebId* parameter.

The Subsite List Result Set MUST only be returned if *@GetWebListForNormalization* is set to "1". The Subsite List Result Set MUST contain one row for each subsite within the specified parent site, and MUST contain no rows if there are no such subsites.

The Subsite List Result Set is defined in URL Result Set, section 2.2.5.25.

3.1.5.21.2 ACL and Permission Result Set

The ACL and Permission Result Set contains information about the permissions associated with the security scope in effect for the document. The ACL and Permission Result Set MUST be returned and MUST contain one row. If the document does not exist, then the values of both columns MUST be NULL. The ACL and Permission Result Set is defined in ACL and Permission Result Set, section 2.2.5.1.

3.1.5.21.3 Document Metadata Result Set

The Document Metadata Result Set contains the metadata about the most current version of the requested document visible to the current user.

The Document Metadata Result Set MUST only be returned if *@Version* is negative. If the document exists, the Document Metadata Result Set MUST contain one row; otherwise, it MUST contain no rows.

The Document Metadata Result Set is defined in Document Metadata Result Set, section 2.2.5.6.

3.1.5.21.4 Document Version Metadata Result Set

The Document Version Metadata Result Set contains the metadata about the requested version of the document.

The Document Version Metadata Result Set MUST only be returned if *@Version* is not negative. If the document exists, the Document Version Metadata Result Set MUST contain one row; otherwise, it MUST contain no rows.

The Document Version Metadata Result Set is in Document Version Metadata Result Set, section 2.2.5.7.

3.1.5.21.5 NULL Result Set

The NULL Result Set is a placeholder that returns no data.

The NULL Result Set MUST only be returned if *@Version* is negative, the requested document exists, and *@DocWebId* is NULL. The NULL Result Set MUST contain zero rows in a schema containing a single unnamed column.

3.1.5.21.6 Event Receivers Result Set (1)

The Event Receivers Result Set contains information about the event receivers defined for this document.

The Event Receivers Result Set MUST only be returned if the requested document exists in the site specified by *@DocWebId*. The Event Receivers Result Set MUST contain one row for each event receiver registered with an **Event Host** type (section 2.2.3.5) of 3 (list item) for this document.

The result set is defined in Event Receivers Result Set, section 2.2.5.9.

3.1.5.21.7 Link Info Single Doc Fixup Result Set

The Link Info Single Doc Fixup Result Set contains information about forward links within the requested document. The Link Info Single Doc Fixup Result Set MUST only be returned if Link information was requested by setting *@CacheParse* to "1".

```
          LinkDirName                    nvarchar(256),
          LinkLeafName                   nvarchar(128),
          LinkType                       tinyint,
          LinkSecurity                   tinyint,
          LinkDynamic                    tinyint,
          LinkServerRel                  bit,
          LinkStatus,                    tinyint,
          PointsToDir                    bit,
          WebPartId                      uniqueidentifier,
          LinkNumber                     int,
          WebId                          uniqueidentifier,
          Search                         nvarchar(max),
          FieldId                        uniqueidentifier;
```

LinkDirName: Contains the directory name of the link.

LinkLeafName: Contains the leaf name of the link.

LinkType: The **LinkType** value of the link in the specified document.

LinkSecurity: A **LinkSecurity** value specifying whether the scheme of the link is HTTP or HTTPS.

LinkDynamic: A **LinkDynamic** value that specifies whether this link is one of several special link types.

LinkServerRel: MUST be 1 to indicate that the link URL is server-relative URL. Otherwise, MUST be 0 to indicate the link URL is not a server-relative URL.

LinkStatus: A **Document Store** type (section 2.2.2.4) indicating the type of document the link points to. If the Link is a forward link for a document that doesn't exist, then this MUST be NULL. This value MUST be NULL if this link entry refers to a location that exists outside the site collection where the document is stored, or if it refers to a location that could not be verified. See **Document Store** type (section 2.2.2.4) for a list of valid values.

PointsToDir: If the link pointed to a directory where a welcome page existed (for example, pointing to http://server when a welcome page like http://server/default.aspx existed), then the link is automatically changed to be the URL to the welcome page itself. This bit MUST be set to 1 if this operation has been performed so that it can be distinguished from an explicit link to the welcome page.

WebPartId: If this link corresponds to a web part within a web part zone, which is therefore logically part of this document, but not present in the HTML source code of this document, then this is the GUID of the web part to which the link belongs.

LinkNumber: An ordinal value denoting the relative order of the link within the source of the document, web part, or field being processed.

WebId: MUST be NULL.

Search: MUST be NULL.

FieldId: If the link is for a list item field within this document, this is the GUID of the field to which the link belongs.

3.1.5.21.8 Web Part Info Result Set

The Web Part Info Result Set MUST only be returned when *@GetWebPartInfo* is set to "1". If web part data for the requested document exists, one row MUST be returned for each web part, otherwise zero rows MUST be returned.

```
tp_Id                           uniqueidentifier,
tp_Level                        tinyint
tp_Source                       nvarchar(max),
tp_AllUsersProperties           varbinary(max);
```

tp_Id: The web part identifier (section 2.2.1.15) of the web part.

tp_Level: Publishing Level type (section 2.2.2.6) value of the document.

tp_Source: Properties of the web part.

tp_AllUsersProperties: A list of the XML properties which are common for all users of the web part.

3.1.5.21.9 Zone ID Result Set

The Zone ID Result Set returns a list of **Web Part zone identifiers**. The Zone ID Result Set MUST only be returned when *@GetWebPartInfo* is set to "1". If web part data for the requested document exists, one row MUST be returned for each web part zone; otherwise zero rows MUST be returned.

```
        tp_ZoneID                         nvarchar(64);
```

tp_ZoneID: The Web Part zone identifier.

3.1.5.21.10 File Format Metadata Result Set

The File Format Metadata Result Set returns metadata information about the format of the content of the document. This result set MUST only be returned if the *@GetFileFormatInfo* parameter is set to NULL or 1.

```
        FileFormatMetaInfo              varbinary(max),
        FileFormatMetaInfoSize          int
        FFMConsistent                   bit
```

@FileFormatMetaInfo: This contains metadata about the format of the content of the document that is to be updated. This can be NULL.

@FileFormatMetaInfoSize: Size in bytes of the metadata returned by the *@FileFormatMetaInfo* column. This MUST be 0 if the *@FileFormatMetaInfo* is NULL.

@FFMConsistent: A bit flag specifying whether the file format metadata info is in a consistent state.

3.1.5.21.11 List Metadata Result Set

The List Metadata Result Set contains the metadata associated with the list containing the requested document.

The List Metadata Result Set MUST only be returned if the *@bGetContainingList* parameter is set to "1" and the document is contained within a list. The List Metadata Result Set MUST contain one row.

The List Metadata Result Set is defined in List Metadata Result Set, section 2.2.5.12.

3.1.5.21.12 Empty List Result Set

The Empty List Result Set contains a single unnamed column with no rows to indicate the document is not contained within a list. The Empty List Result Set MUST only be returned if the *@bGetContainingList* parameter is set to "1" and the document is not contained within a list.

The Empty List Result Set is defined in Empty Result Set, section 2.2.5.8.

3.1.5.21.13 Event Receivers Result Set (2)

The Event Receivers Result Set contains the event receivers associated with the list which contains the requested document.

The Event Receivers Result Set MUST only be returned if the *@bGetContainingList* parameter is set to "1" and the document is contained within a list. There MUST be one row in the Event Receivers Result Set for each event receiver registered with an **Event Host** type (section 2.2.3.5) of 2 (List) for the list.

The Event Receivers Result Set is defined in Event Receivers Result Set, section 2.2.5.9.

3.1.5.21.14 Document Information and Content (Update) Result Set

The Document Information and Content (Update) Result Set contains information about the document stream for the current version of the requested document.

The Document Information and Content (Update) Result Set MUST only be returned if *@GetContent* is set to "1" and *@Version* is negative. If the publishing level of the document specified by *@Level* exists, the Document Information and Content (Update) Result Set MUST contain one row; otherwise, the Document Information and Content (Update) Result Set MUST contain no rows.

The Document Information and Content (Update) Result Set is similar to the Document Information and Content (Read) Result Set, except for a number of columns that are unnamed here.

```
{Size}                          int,
{Content}                       varbinary(max)
{RbsId}                         varbinary(800),
ETagVersion                     int,
Id                              uniqueidentifier,
SetupPathVersion                tinyint,
SetupPath                       nvarchar(255),
{Dirty}                         bit,
DocFlags                        int,
{IsHistoricalVersion}           bit,
Level                           tinyint,
InternalVersion                 int
```

{Size}: The size of the requested document, in bytes.

{Content}: The document stream of the document. For a ghosted document, content MUST be NULL. If the content is larger than the value specified in the *@ChunkSize* parameter, then the amount of data returned from the document stream will be equal to the number of bytes specified by the *@ChunkSize* parameter starting at the first byte of the document stream. The front-end web server can then request individual chunks of content in subsequent requests.

{RbsId}: If remote blob storage is enabled and the document's content is contained in a remote data store, then this MUST be the remote blob storage identifier for the document's content. If remote blob storage is disabled or the document's content is not contained in a remote data store, then this MUST contain NULL. For additional information regarding remote blob storage, see [MS-WSSO] section 2.1.2.3.8.

ETagVersion: An internal version counter incremented any time a change is made to this document, and used for internal conflict detection.

Id: The **document identifier** (section 2.2.1.2) of this document.

SetupPathVersion: For a ghosted document, this governs the setup path location which the **SetupPath** fragment is relative to. This value MUST be NULL if the document does not exist, and is undefined for a document which was never ghosted.<13>

SetupPath: For a document which is now or once was ghosted, this parameter MUST contain the setup path fragment relative to the base setup path described by the **SetupPathVersion** value, where the content **stream** of this document can be found. In the case of a document which does not exist or a document which was never ghosted, this value MUST be NULL.

{Dirty}: A bit set to "1," indicating that this document MUST have dependency update processing performed before its document stream is returned to an external agent. This field's value is implementation-dependent if this document does not have a document stream, and MUST be ignored if the document does not have a document stream.

DocFlags: A **Doc Flags** (section 2.2.2.3) value describing the document. This value MUST be NULL if the document does not exist.

{IsHistoricalVersion}: MUST be 0.

Level: The **Publishing Level** type (section 2.2.2.6) value of the requested version of the document.

InternalVersion: An integer value specifying the implementation-related internal version number of this version of the document. This value MUST be **Docs View** (section 2.2.7.4) set to the current internal version counter value for the document in the **Docs View**.

3.1.5.21.15 Document Version 1 Information and Content Result Set

The Document Version 1 Information and Content Result Set contains information about the document stream for the requested version of the document.

The Document Version 1 Information and Content Result Set MUST only be returned if *@GetContent* is set to "1" and *@Version* is not negative, and the document exists with the publishing level specified by *@Level* and with the user interface (UI) version specified by *@Version*.

The Document Version 1 Information and Content Result Set MUST contain one row if it is returned.

```
{Size}                          int,
{Content}                       varbinary(max)
{RbsId}                         varbinary(800),
{Version}                       int,
{Id}                            uniqueidentifier,
{SetupPathVersion}              tinyint,
{SetupPath}                      nvarchar(255),
{Dirty}                         bit,
DocFlags                        int,
{IsHistoricalVersion}           bit,
{Level}                         tinyint,
InternalVersion                 int
```

{Size}: The size of the requested document in bytes.

{Content}: The document stream of the document. For a ghosted document, content MUST be NULL. If the content is larger than the value specified in the *@ChunkSize* parameter, then the amount of data returned from the document stream will be equal to the number of bytes specified by the *@ChunkSize* parameter starting at the first byte of the document stream. The front-end web server can then request individual chunks of content in subsequent requests.

{RbsId}: If remote blob storage is enabled and the document's content is contained in a remote data store, then this MUST be the remote blob storage identifier for the document's content. If remote blob storage is disabled or the document's content is not contained in a remote data store, then this MUST contain NULL. For additional information regarding remote blob storage, see [MS-WSSO] section 2.1.2.3.8.

{Version}: An internal version counter incremented any time a change is made to this document, and used for internal conflict detection. This value MUST be set to the current internal version counter value for the document in the **Docs View** (section 2.2.7.4), except in the Document Version 1 Information and Content Result Set and the Document Version 2 Information and Content Result Set (section 3.1.5.21.16), where this value MUST be NULL.

{Id}: The **document identifier** (section 2.2.1.2) of this document.

{SetupPathVersion}: For a ghosted document, this governs the setup path location to which the **SetupPath** fragment is relative. This value MUST be NULL if the document does not exist, and is undefined for a document which was never ghosted.<14>

{SetupPath}: For a document that is now or once was ghosted, this parameter MUST contain the setup path fragment relative to the base setup path described by the **SetupPathVersion** value, where the content **stream** of this document can be found. In the case of a document which does not exist or a document which was never ghosted, this value MUST be NULL.

{Dirty}: A bit set to "1," indicating that this document MUST have dependency update processing performed before its document stream is returned to an external agent. This field's value is implementation-dependent if this document does not have a document stream, and MUST be ignored if the document does not have a document stream.

DocFlags: A **Doc Flags** (section 2.2.2.3) value describing the document. This value MUST be NULL if the document does not exist.

{IsHistoricalVersion}: MUST be 1.

{Level}: MUST be 1.

InternalVersion: An integer value specifying the implementation-related internal version number of this version of the document. This value MUST be set to the current internal version counter value for the document in the **Docs View**.

3.1.5.21.16 Document Version 2 Information and Content Result Set

The Document Version 2 Information and Content Result Set contains information about the document stream for the nearest available publishing level of the requested version of the document.

The Document Version 2 Information and Content Result Set MUST only be returned if *@GetContent* is set to "1" and *@Version* is not negative, and the document exists with the user interface (UI) version specified by *@Version*, but with a publishing level visible to the current user that is less than the value returned in the *@Level* output parameter. There MUST be one row in the Document Version 2 Information and Content Result Set if it is returned.

```
{Size}                         int,
{Content}                      varbinary(max)
{RbsId}                        varbinary(800),
{Version}                      int,
{Id}                           uniqueidentifier,
SetupPathVersion               tinyint,
SetupPath                      nvarchar(255),
{Dirty}                        bit,
DocFlags                       int,
{IsHistoricalVersion}          bit,
{Level}                        tinyint,
InternalVersion                int
```

For the field descriptions, see Document Version 1 Information and Content Result Set, section 3.1.5.21.15.

3.1.5.21.17 Attachment State Result Set

The Attachment State Result Set contains the information about the attachment state of the requested document. The Attachment State Result Set MUST be returned and MUST contain one row.

The Attachment State Result Set is defined using T-SQL syntax, as follows.

```
{IsAttachment}                 bit,
{NeedManageListRight}          bit,
{Level}                        tinyint,
```

{IsAttachment}: This specifies whether the document is associated with an attachment. This value MUST be 1 if the document is an attachment, a list item attachment folder, or the list attachment folder itself. Otherwise, this value MUST be 0. See **Attachments Flag** (section 2.2.3.1) for more information.

{NeedManageListRight}: This specifies whether operations on this document require the user to have the **ManageLists** bit of the **WSS Rights Mask** (section 2.2.2.14) set. This value MUST be 0 if the document is not stored in a list.

{Level}: A **Publishing Level** type (section 2.2.2.6) value specifying the publishing status of the document. This value MUST be NULL if the document does not exist.

3.1.5.22 proc_FetchWelcomeNames

The **proc_FetchWelcomeNames** stored procedure lists all the names of the default welcome pages used as redirection targets when folders are requested by an HTTP GET in all site collections in the back-end database server.

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

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

Return Values: The **proc_FetchWelcomeNames** stored procedure MUST return an integer return code of 0.

The **proc_FetchWelcomeNames** stored procedure MUST return one result **set:**

3.1.5.22.1 Welcome Pages Result Set

The Welcome Pages Result Set returns the list of configured default welcome page names, with one row for each configured welcome page name.

```
      LeafName                       nvarchar(128);
```

LeafName: A string containing the configured welcome page name in leaf name format, for example, "default.aspx", "default.htm".

3.1.5.23 proc_GenerateNextId

The **proc_GenerateNextId** stored procedure returns an identifier to be used for a new list item in a specified list, and to increment the value of the identifier returned by the next call to **proc_GenerateNextId** for the same list.

```
PROCEDURE proc_GenerateNextId (
      @WebId                         uniqueidentifier,
      @ListId                        uniqueidentifier,
      @NumIds                        int = 1,
      @RequestGuid                   uniqueidentifier = NULL     OUTPUT
);
```

@WebId: The site identifier (section 2.2.1.11) of the site containing the list.

@ListId: The list identifier (section 2.2.1.5) of the List.

@NumIds: **proc_GenerateNextId** MUST increment the value of the list item identifier (section 2.2.1.6) returned by the next call by this value. This value MUST be a positive number (greater than zero). If the parameter is not provided, 1 is used.

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

Return Values: The **proc_GenerateNextId** stored procedure returns a positive integer return code, which is the available list item identifier for the specified list. The **proc_GenerateNextId** stored procedure MUST NOT return a result set.

3.1.5.24 proc_GetAllRolesForUser

The **proc_GetAllRolesForUser** stored procedure retrieves the role identifiers for the roles assigned to a set of **principals** in a given security scope.

```
PROCEDURE proc_GetAllRolesForUser (
@SiteId uniqueidentifier,
@ScopeId uniqueidentifier,
@PrincipalIdsCsv nvarchar(max),
@RequestGuid uniqueidentifier = null OUTPUT,
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the list.

@ScopeId: The scope identifier (section 2.2.1.8) of the security scope for which roles are to be returned.

@PrincipalIdsCsv: The principal identifiers for which roles are to be returned. This parameter MUST be an **nvarchar(max)** that conforms to the following **Augmented Backus-Naur Form (ABNF):** as specified in [RFC5234].

```
PrincipalIdsCsv = *PrincipalId
PrincipalId = 1*DIGIT
```

Each **PrincipalId** MUST be greater than 0 and less than 2,147,483,648 when interpreted as a decimal integer.

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

Result Sets:

This procedure MUST return the user Roles Result Set (section 3.1.5.24.1).

3.1.5.24.1 User Roles Result Set

```
RoleId int NOT NULL,
```

RoleId: A **role identifier** assigned to one or more of the **principal** identifiers specified by the **PrincipalId** listed by the *@PrincipalIdsCsv* parameter.

3.1.5.25 proc_GetAuditMask

The **proc_GetAuditMask** stored procedure is invoked to identify **Audit Flags** (section 2.2.2.1) information for a specified object (a page, file, document, or site collection).

```
PROCEDURE proc_GetAuditMask(
        @ItemType                     tinyint,
        @SiteId                       uniqueidentifier,
        @DirName                      nvarchar(256),
        @LeafName                     nvarchar(128),
        @RequestGuid                  uniqueidentifier = NULL OUTPUT
);
```

@ItemType: The **Audit Item** type (section 2.2.3.2) of the object.

@SiteId: The site collection identifier (section 2.2.1.9) for the site collection containing the object. This is used to get the global audit mask.

@DirName: The directory name of the object.

@LeafName: The leaf name of the object.

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

Return Values: The **proc_GetAuditMask** stored procedure returns an integer return code, which MUST be listed in the following table.

Value	Description
0	Successful execution.
2	An object of the specified **Audit Item** type could not be found at the specified location.

The **proc_GetAuditMask** stored procedure MUST return one result set.

3.1.5.25.1 Audit Mask Result Set

The Audit Mask Result Set contains the context-sensitive identifier for the specified object, and the **Audit Flags** (section 2.2.2.1) set and inherited on that object. The Audit Mask Result Set MUST be returned and MUST contain a single row. The Audit Mask Result Set is defined in Site Audit Mask Result Set, section 2.2.5.20.

3.1.5.26 proc_GetAuditMaskOutput

The **proc_GetAuditMaskOutput** stored procedure is invoked to get **Audit Flags** (section 2.2.2.1) information about an object (a page, file, document, or site collection).

```
PROCEDURE proc_GetAuditMaskOutput(
        @ItemType                     tinyint,
        @SiteId                       uniqueidentifier,
        @DirName                      nvarchar(256),
        @LeafName                     nvarchar(128),
        @Id                           uniqueidentifier              OUTPUT,
        @AuditFlags                   int                           OUTPUT,
        @InheritAuditFlags            int                           OUTPUT,
        @GlobalAuditMask              int                           OUTPUT,
        @RequestGuid                  uniqueidentifier = NULL       OUTPUT
);
```

@ItemType: The **Audit Item** type (section 2.2.3.2) of the object.

@SiteId: A site collection identifier (section 2.2.1.9) for the site collection containing the object. This is used to get the global audit mask information.

@DirName: The directory name of the object.

@LeafName: The leaf name of the object.

@Id: A **document identifier** (section 2.2.1.2) identifying the object, returned as an output parameter.

@AuditFlags: An **Audit Flags** value of the operations to be audited which are set directly on the specified object, returned as an output parameter.

@InheritAuditFlags: An **Audit Flags** value of the operations to be audited on the specified object which are inherited from its containing list, site, or site collection, returned as an output parameter.

@GlobalAuditMask: An **Audit Flags** value of the operations to be audited on the specified object which are inherited from the specified site collection, returned as an output parameter.

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

Return Values: The **proc_GetAuditMaskOutput** stored procedure returns an integer return code, which MUST be listed in the following table.

Value	Description
0	Successful execution.
2	An object of the specified **Audit Item** type could not be found at the specified location.

The **proc_GetAuditMaskOutput** stored procedure MUST return no result sets.

3.1.5.27 proc_GetBlobIdsDocs

The **proc_GetBlobIdsDocs** stored procedure is invoked to read a batch of document stream information that is stored in external storage for the current version of each document.

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

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection which contains the documents identified by *@DocId*.

@DocId: The **document identifier** (section 2.2.1.2) of the starting document of the batch to be generated.

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

Return Values: The **proc_GetBlobIdsDocs** stored procedure MUST return an integer return code of 0.

The **proc_GetBlobIdsDocs** stored procedure MUST return one Document Stream External Storage Information Result Set (section 3.1.5.27.1).

3.1.5.27.1 Document Stream External Storage Information Result Set

The Document Stream External Storage Information Result Set MUST contain from 0 up to 1000 rows of document stream information, one row for each document stream found in external storage for the current version of a document with a **document identifier** (section 2.2.1.2) greater than or equal to the value specified by *@DocId*, and MUST be ordered by site collection, document identifier and publishing level.

```
        Content                        varbinary(max),
        ID                             uniqueidentifier,
        {Version}                      int;
```

Content: The document stream content.

ID: The document identifier of the document associated with this document stream.

{Version}: This parameter MUST be the constant integer 0.

3.1.5.28 proc_GetBlobIdsVersions

The **proc_GetBlobIdsVersions** stored procedure is invoked to read a batch of document stream information that is stored in external storage for **historical versions** of each document.

```
    PROCEDURE proc_GetBlobIdsVersions (
          @SiteId                        uniqueidentifier,
          @DocId                         uniqueidentifier,
          @Version                       int,
          @RequestGuid                   uniqueidentifier = NULL       OUTPUT
    );
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection that contains the documents historical versions.

@DocId: The **document identifier** (section 2.2.1.2) of the starting document to include in the batch.

@Version: A version number less than any historical version number for the initial document to return. For an initial call to **proc_GetBlobIdsVersions**, this can be 0. Otherwise, to obtain the next batch of information, this can be the version number of the last document returned by a previous call to **proc_GetBlobIdsVersions** before the starting document to include in the batch.

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

Return Values: The **proc_GetBlobIdsVersions** result set MUST return a return code of 0.

The **proc_GetBlobIdsVersions** result set MUST return one result set as follows.

3.1.5.28.1 Document Version External Storage Information Result Set

The Document Version External Storage Information Result Set MUST contain from 0 up to 1000 rows of document stream information, one row for each document stream found in external storage for the historical versions of documents with a **document identifier** (section 2.2.1.2) greater than or equal to the value specified by *@DocId*, and MUST be ordered by site collection, document identifier, and version.

```
        Content                        varbinary(max),
        ID                             uniqueidentifier,
        Version                        int;
```

Content: The document stream content of the historical version of the document.

ID: The document identifier of the document with a historical version. This value MUST be greater than or equal to *@DocId*.

Version: The version number of the historical version of the document. If **ID** is equal to *@DocId*, then the **Version** value MUST be greater than *@Version*.

3.1.5.29 proc_GetContainingList

The **proc_GetContainingList** stored procedure is invoked to get metadata and event receiver information about the list containing a specified URL.

```
PROCEDURE proc_GetContainingList(
      @SiteId                        uniqueidentifier,
      @WebId                         uniqueidentifier,
      @Url                           nvarchar(260),
      @RequestGuid                   uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the list.

@WebId: The site identifier (section 2.2.1.11) of the site containing the list.

@Url: The URL in store-relative form of a list item or document within the list. The URL is used to derive the location of the containing list. The leaf name part of this parameter is ignored because it could point to a nonexistent document or list item.

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

Return Values: This stored procedure returns an integer return code which MUST be listed in the following table.

Value	Description
0	Successful execution.
1	The list was found in a different site other than that specified by the *@WebId* parameter or the site specified by the *@WebId* parameter was not found.
3	The list does not exist in the site that was specified by the *@WebId* parameter.

This stored procedure MUST return one result set if the containing list was not found, and MUST return two result sets if the containing list was found, as defined below.

3.1.5.29.1 List Metadata Result Set

The List Metadata Result Set contains the metadata associated with the list containing the value in the *@Url* parameter.

The List Metadata Result Set MUST only be returned if a containing list is found for the value for the *@Url* parameter and MUST contain one row.

This result set is defined in List Metadata Result Set, section 2.2.5.12.

3.1.5.29.2 Empty Result Set

The Empty Result Set MUST only be returned if a containing list is not found for the value in the *@Url* parameter.

This result set is defined in Empty Result Set, section 2.2.5.8.

3.1.5.29.3 Event Receivers Result Set

The Event Receivers Result Set contains information about the event receivers defined for the list that contains the value in the *@Url* parameter.

The Event Receivers Result Set MUST only be returned if a containing list is found for the value in the *@Url* parameter. This result set MUST contain one row for each event receiver registered with an **Event Host** type (section 2.2.3.5) of 2 for the list.

This result set is defined in Event Receivers Result Set, section 2.2.5.9.

3.1.5.30 proc_GetDocsMetaInfo

The **proc_GetDocsMetaInfo** stored procedure is invoked to request document metadata information for up to ten documents within a specified site.

```
PROCEDURE proc_GetDocsMetaInfo(
      @DocSiteId                      uniqueidentifier,
      @WebFullUrl                     nvarchar(260),
      @GetDocsFlags                   int,
      @UserId                         int,
      @DirName1                       nvarchar(256)      = NULL,
      @LeafName1                      nvarchar(128)      = NULL,
      @AttachmentsFlag1               tinyint            = NULL,
      @Level1                         tinyint            = NULL,
      @DirName2                       nvarchar(256)      = NULL,
      @LeafName2                      nvarchar(128)      = NULL,
      @AttachmentsFlag2               tinyint            = NULL,
      @Level2                         tinyint            = NULL,
      @DirName3                       nvarchar(256)      = NULL,
      @LeafName3                      nvarchar(128)      = NULL,
      @AttachmentsFlag3               tinyint            = NULL,
      @Level3                         tinyint            = NULL,
      @DirName4                       nvarchar(256)      = NULL,
      @LeafName4                      nvarchar(128)      = NULL,
      @AttachmentsFlag4               tinyint            = NULL,
      @Level4                         tinyint            = NULL,
      @DirName5                       nvarchar(256)      = NULL,
      @LeafName5                      nvarchar(128)      = NULL,
      @AttachmentsFlag5               tinyint            = NULL,
      @Level5                         tinyint            = NULL,
      @DirName6                       nvarchar(256)      = NULL,
      @LeafName6                      nvarchar(128)      = NULL,
      @AttachmentsFlag6               tinyint            = NULL,
      @Level6                         tinyint            = NULL,
      @DirName7                       nvarchar(256)      = NULL,
      @LeafName7                      nvarchar(128)      = NULL,
      @AttachmentsFlag7               tinyint            = NULL,
      @Level7                         tinyint            = NULL,
      @DirName8                       nvarchar(256)      = NULL,
      @LeafName8                      nvarchar(128)      = NULL,
      @AttachmentsFlag8               tinyint            = NULL,
      @Level8                         tinyint            = NULL,
      @DirName9                       nvarchar(256)      = NULL,
      @LeafName9                      nvarchar(128)      = NULL,
      @AttachmentsFlag9               tinyint            = NULL,
      @Level9                         tinyint            = NULL,
      @DirName10                      nvarchar(256)      = NULL,
      @LeafName10                     nvarchar(128)      = NULL,
      @AttachmentsFlag10              tinyint            = NULL,
      @Level10                        tinyint            = NULL,
      @RequestGuid                    uniqueidentifier   = NULL      OUTPUT
);
```

@DocSiteId: The site collection identifier (section 2.2.1.9) for the site collection containing the specified documents.

@WebFullUrl: The store-relative form URL of the site containing the specified documents.

@GetDocsFlags: A bit mask with a flag that specifies whether to return link information. If this parameter has the bit 0x00000020 set, then link information MUST be returned in the Link Info Result Set (section 3.1.5.30.5).

@UserId: The user identifier (section 2.2.1.13) for the current user requesting the information.

The next four parameters are duplicated 10 times, with each set of parameters referring to a document to be fetched. Each instance of these individual parameter names is differentiated by a suffix with a value of 1 through 10, which replaces the placeholder "#" symbol shown below.

@DirName#**:** The directory name of the specified document. A NULL value signifies that no document is being fetched in this slot, and the next three parameters MUST be ignored.

@LeafName#: The leaf name of the specified document. If *@DirName#* is not NULL, this MUST NOT be NULL.

@AttachmentsFlag#: An **Attachments Flag** (section 2.2.3.1) value which specifies the type of security checks to be performed by **proc_GetDocsMetaInfo** based on whether it appears to be an attachment.

@Level#: A **Publishing Level** type (section 2.2.2.6) value specifying the maximum publishing level of the current version of the document to be returned to the front-end web server if multiple current versions of the document are available.

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

Return Values: The **proc_GetDocsMetaInfo** stored procedure returns an integer return code that MUST be 0.

The **proc_GetDocsMetaInfo** stored procedure MUST return multiple result **sets:**

3.1.5.30.1 Individual URL Security Result Set

The Individual URL Security Result Set contains security information about the specified document. The Individual URL Security Result Set to be used to check if the current user has permission to see this document's metadata. If the document does not exist, but the specified URL is within a list or document library, security information is returned from the security scope for the specified document location.

One Individual URL Security Result Set or NULL Individual URL Security Result Set MUST be returned for each document whose corresponding *@DirName#* parameter is not NULL. If all *@DirName#* parameters are set to NULL, Individual URL Security Result Sets MUST NOT be returned.

Each Individual URL Security Result Set MUST only be returned if the specified document location is contained within a List or document library. Otherwise, the NULL Individual URL Security Result Set (section 2.2.5.14) MUST be returned instead. If returned, the Individual URL Security Result Set MUST contain a single row.

The result set is defined in Individual URL Security Result Set, section 2.2.5.10.

3.1.5.30.2 NULL Individual Url Security Result Set

The NULL Individual URL Security Result Set indicates that the specified document location is not contained within a List or document library.

One Individual URL Security Result Set or NULL Individual URL Security Result Set MUST be returned for each document whose corresponding *@DirName#* parameter is not NULL. If all *@DirName#* parameters are set to NULL, NULL Individual URL Security Result Sets MUST NOT be returned.

Each NULL Individual URL Security Result Set MUST only be returned if the specified document location is not contained within a List or document library. Otherwise, the Individual URL Security Result Set (section 2.2.5.10) MUST be returned instead. If returned, the NULL Individual URL Security Result Set MUST contain a single row.

The NULL Individual URL Security Result Set is defined in NULL Individual URL Security Result Set, section 2.2.5.14.

3.1.5.30.3 Server Time Result Set

The Server Time Result Set returns the current time from the back-end database server in **UTC**. The Server Time Result Set MUST be returned. The Server Time Result Set MUST only contain a single row. The Server Time Result Set is defined in Server Time Result Set, section 2.2.5.18.

3.1.5.30.4 Subsite List Result Set

The Subsite List Result Set contains an unordered list of store-relative form URLs for all subsites whose parent site is specified in the *@WebFullUrl* parameter.

The Subsite List Result Set MUST be returned. It MUST contain one row for each subsite with the specified parent site, and MUST contain no rows if there are no such subsites.

The Subsite List Result Set is defined in URL Result Set, section 2.2.5.25.

3.1.5.30.5 Link Info Result Set

The Link Info Result Set returns information about each forward link from, and each backward link to, the current version of the specified documents within the site collection. The Link Info Result Set MUST only be returned if link information was requested by the *@GetDocsFlags* parameter. The Link Info Result Set MUST contain one row per link for each specified document.

The Link Info Result Set MUST be ordered by the DocId column, and is defined in Single Doc Link Information Result Set, section 2.2.5.19.

3.1.5.30.6 Multiple Document Metadata Result Set

The Multiple Document Metadata Result Set contains the metadata for the specified documents. The Multiple Document Metadata Result Set MUST contain one row for each document where the corresponding *@DirName#* parameter was not set to NULL, and the rows MUST be ordered by the **DocId** column.

```
DocId                          uniqueidentifier,
{FullUrl}                      nvarchar(385),
Type                           tinyint,
MetaInfoTimeLastModified       datetime,
MetaInfo                       varbinary(max),
Size                           int,
TimeCreated                    datetime,
TimeLastModified               datetime,
ETagVersion                    int,
DocFlags                       int,
{ListType}                     int,
tp_Name                        nvarchar(38),
{ListTitle}                    nvarchar(255),
{CacheParseId}                 uniqueidentifier,
GhostDirName                   nvarchar(256),
GhostLeafName                  nvarchar(128),
tp_Login                       nvarchar(255),
CheckoutDate                   datetime,
CheckoutExpires                datetime,
VirusStatus                    int,
VirusInfo                      nvarchar(255),
SetupPathVersion               tinyint,
SetupPath                      nvarchar(255),
SetupPathUser                  nvarchar(255),
NextToLastTimeModified         datetime,
UIVersion                      int,
CheckinComment                 nvarchar(1023),
WelcomePageUrl                 nvarchar(260),
WelcomePageParameters          nvarchar(max),
tp_Flags                       bigint,
Acl                            varbinary(max),
AnonymousPermMask              bigint,
DraftOwnerId                   int,
Level                          tinyint,
ParentVersion                  int,
TransformerId                  uniqueidentifier,
ParentLeafName                 nvarchar(128),
ProgId                         nvarchar(255),
DoclibRowId                    int,
tp_DefaultWorkflowId           uniqueidentifier,
ListId                         uniqueidentifier,
ItemChildCount                 int,
FolderChildCount               int,
MetaInfoVersion                int,
{CurVerMetaInfo}               varbinary(max),
ContentVersion                 int,
{bContentVersionIsDirty}       bit;
```

DocId: If the specified document exists, then this MUST be the **document identifier** (section 2.2.1.2), otherwise, this MUST be a generated document identifier value for the specified document.

{FullUrl}: The store-relative form URL for the specified document. This value MUST be NULL if the document does not exist. Otherwise, the size of the **nvarchar** type returned SHOULD vary depending on whether or not the store-relative form URL has only a directory name, only a leaf name, or both. If it has an empty directory name, then it SHOULD be returned as **nvarchar(128)**. If it has an empty leaf name, then it SHOULD be returned as **nvarchar(256)**. Otherwise, it MUST be returned as **nvarchar(385)**. Since a front-end web server cannot determine which type will be returned in advance, it MUST allow for a data type returned as **nvarchar(385)**.

Type: The **Document Store** type (section 2.2.2.4) of this document. This value MUST be NULL if the requested document does not exist.

MetaInfoTimeLastModified: A datetime with a timestamp in **UTC** format specifying the last time the metainfo value of this document was changed. This value MUST be NULL if the document does not exist.

MetaInfo: A METADICT for the document. The METADICT format is specified in [MS-FPSE] section 2.2.2.2.11. This value MUST be NULL if the document does not exist.

Size: The number of bytes in the document stream. This value MUST be NULL if the document does not exist.

TimeCreated: A datetime with a timestamp in UTC format specifying when this document was created. This value MUST be NULL if the document does not exist.

TimeLastModified: A datetime with a timestamp in UTC format specifying when the document was last modified. This corresponds to the **TimeCreated** (for historical versions) or **TimeLastModified** (for current versions) of the document. This value MUST be NULL if the document does not exist.

ETagVersion: A counter incremented any time a change is made to this document, used for internal conflict detection. This value MUST be NULL if the document does not exist.

DocFlags: A **Doc Flags** (section 2.2.2.3) value describing the document. This value MUST be NULL if the document does not exist.

{ListType}: A combination of the **List Base** type (section 2.2.3.11) and **List Server Template** (section 2.2.3.12) values of the associated list for this document, where the **List Server Template** value is multiplied by 256 and added to the value of the **List Base** type. This value MUST be NULL if the document does not exist or if it is not in a list.

tp_Name: The identifier of the list that contains this document. This value MUST be NULL if the document does not exist or if it is not in a list.

{ListTitle}: If this document is the root folder of a list, then this contains the display name of the list. This value MUST be NULL if the document does not exist or if it is not in a list.

{CacheParseId}: This value MUST be NULL.

GhostDirName: The directory name as passed to **proc_GetDocsMetaInfo** in the *@DirName#* parameter.

GhostLeafName: The leaf name as passed to **proc_GetDocsMetaInfo** in the *@LeafName#* parameter.

tp_Login: If the document exists and is currently checked out, then **tp_Login** is the login name of the user to whom it is checked out. In all other cases, this is NULL.

CheckoutDate: A datetime with a timestamp in UTC format specifying when this document was checked out. If the document is currently checked in or the document does not exist, this MUST be NULL.

CheckoutExpires: A datetime with a timestamp in UTC format specifying when the **short-term lock** for this document will expire. If this date is in the past, then the document SHOULD be treated as unlocked. This value can be NULL if no short-term lock has been placed on the document.

VirusStatus: A **Virus Status** (section 2.2.3.18) value specifying the current virus state of this document. This value can be NULL if the document has not been processed by a virus scanner. This value MUST be NULL if the document does not exist.

VirusInfo: A string containing a provider-specific message returned by the virus scanner when it last processed the document. This value MUST be NULL if the document does not exist or if the document has not been processed by a virus scanner.

SetupPathVersion: For a ghosted document, this specifies the setup path location that the **SetupPath** fragment is relative to. This value MUST NOT be NULL. The following are valid values.

Value	Description
2	This is relative to the install location of Windows SharePoint Services 2.0 on the front-end web server (for example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\60).
3	This is relative to the install location of Windows SharePoint Services 3.0 on the front-end web server (for example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\12).
4	This is relative to the install location of Microsoft SharePoint Foundation 2010 on the front-end web server (for example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\14).

SetupPath: For a document that is or has been ghosted, this contains the **SetupPath** fragment relative to the base setup path specified by the **SetupPathVersion** value, where the document **stream** of this document can be found. This value MUST be NULL for a document that does not exist or was never ghosted.

SetupPathUser: If this document is now or once was ghosted, then this contains the login name of the user that created the ghosted document. This value MUST be NULL for a document that does not exist or was never ghosted.

NextToLastTimeModified: The value of **TimeLastModified** from the previous time when the document was last saved. This value MUST be NULL for a document that does not exist.

UIVersion: The UI version number of the document. This value MUST be NULL for a document that does not exist.

CheckinComment: An optional user-supplied description provided when a document is checked in or published. This value MUST be NULL for a document that does not exist or is not checked in.

WelcomePageUrl: If this document is a folder, this specifies an optional page to redirect to when the folder is requested with an HTTP GET operation. The URL is relative to the URL of the folder itself, and MUST be contained within that folder. Attempts to break out of the folder, such as "../../somepage.aspx", are not valid. If the document does not exist, or a welcome page is not specified, this value MUST be NULL.

WelcomePageParameters: Contains optional URL parameters to specify as part of the **WelcomePageUrl** value. These parameters are the URL substring starting at either the query string signifier "?" or the bookmark signifier "#". If the document does not exist, or parameters are not specified, this value MUST be NULL.

tp_Flags: The **List Flags** (section 2.2.2.5) value for the list that contains this document. This value MUST be NULL for documents that do not exist or are not in a List.

Acl: The binary serialization of the **WSS ACL Format** (section 2.2.4.6) access control list (ACL) for this document. The WSS ACL is either explicitly defined for the document, or inherited from the parent object of the document. This value MUST be NULL for documents that do not exist.

AnonymousPermMask: A **WSS Rights Mask** (section 2.2.2.14) indicating the rights granted to an anonymous user, or to a user who has no specific rights to this document. The value MUST be NULL for documents that do not exist.

DraftOwnerId: The user identifier (section 2.2.1.13) of the user that published this document as a draft. This value MUST be NULL if the document is not a draft version or does not exist.

Level: A **Publishing Level** type (section 2.2.2.6) value specifying the publishing status of this document. This value MUST be NULL if the document does not exist.

ParentVersion: If the document is a transformed version of another document, then this is the UI version value from the parent document. This value MUST be NULL if the document does not exist or is not the product of a document transformation.

TransformerId: If this document is a transformed version of another document, this is the Globally Unique Identifier (GUID) of the agent that performed the transformation. This value MUST be NULL if the document does not exist or is not the product of a document transformation.

ParentLeafName: If the document is a transformed version of another document, then this is the leaf name of the original document. The original document MUST be in the same folder as the transformed document. If either is moved, then the relationship between the original and transformed document is broken. This value MUST be NULL if the document is not the product of a document transformation or does not exist.

ProgId: Specifies a **ProgId**, or preferred application to open the document. The **ProgId** value is used to distinguish between different applications that save files with a given file extension (for example, different editing applications for .HTML or .XML files). This value MUST be NULL if the document does not exist or if a **ProgId** was not specified when the document was saved.

DoclibRowId: The document library row identifier for this item. If the document does not exist or is not contained in a List, this value MUST be NULL.

tp_DefaultWorkflowId: The workflow identifier (section 2.2.1.16) corresponding to the **workflow** to be invoked if the document is in a moderated list and the document is submitted for approval as part of a check in. If the document does not exist or is not contained in a list with a configured approval workflow, then this value MUST be NULL.

ListId: The list identifier (section 2.2.1.5) of the list that contains the document. If the document does not exist or is not contained in a list, then this value MUST be NULL.

ItemChildCount: The number of non-folder children of the document.

FolderChildCount: The number of folder children of the document.

MetaInfoVersion: A counter incremented any time a change is made to the **MetaInfo** for the document and used for internal conflict detection.

{CurVerMetaInfo}: This value MUST be NULL.

ContentVersion: A counter incremented any time a change is made to the binary contents of this document; used for internal conflict detection. This value MUST be NULL if the document does not exist.

{bContentVersionIsDirty}: This value specifies whether the **ContentVersion** value can be used for conflict detection. If the **ContentVersion** value can be used for conflict detection or the **ContentVersion** value is NULL, this value MUST be 0; otherwise it MUST be 1.

3.1.5.31 proc_GetLinkInfoSingleDoc

The **proc_GetLinkInfoSingleDoc** stored procedure provides link information and status for all forward links within a specified document and for all backward links within the specified site collection to the document.

```
PROCEDURE proc_GetLinkInfoSingleDoc(
      @DocSiteId                      uniqueidentifier,
      @DocDirName                     nvarchar(256),
      @DocLeafName                    nvarchar(128),
      @Level                          tinyint,
      @RequestGuid                    uniqueidentifier = NULL OUTPUT
);
```

@DocSiteId: The site collection identifier (section 2.2.1.9) for a site collection containing the document.

@DocDirName: The directory name of the document.

@DocLeafName: The leaf name of the document.

@Level: A **Publishing Level** type (section 2.2.2.6) indicating the publishing level of the document.

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

Return Values: This stored procedure MUST return an integer return code of 0.

This stored procedure MUST return a single result set, as specified in the following section.

3.1.5.31.1 Link Info Single Doc Result Set

The Link Info Single Doc Result Set contains information about links to or within the requested document. Entries are present both for forward links and for backward links . The Link Info Single Doc Result Set MUST be returned, and MUST contain one row for each forward link within the specified document, and one row for each backward link to the document within the specified site collection. The Link Info Single Doc Result Set is defined in Link Information Result Set, section 2.2.5.11.

3.1.5.32 proc_GetListCheckedOutFiles

The **proc_GetListCheckedOutFiles** stored procedure is invoked to retrieve a list of all documents with a **Document Store** type (section 2.2.2.4) of 0 (File) within a specified list, folder, or its subfolders, which are checked out and do not have checked in draft or published versions.

```
    PROCEDURE proc_GetListCheckedOutFiles(
          @SiteId                       uniqueidentifier,
          @ListUrl                      nvarchar(260),
          @RequestGuid                  uniqueidentifier   = NULL      OUTPUT
    );
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the list.

@ListUrl: The directory name which contains the document(s), preceded with a leading slash ("/").

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

Return Values: The **proc_GetListCheckedOutFiles** stored procedure returns an integer return code, which MUST be 0.

The **proc_GetListCheckedOutFiles** stored procedure MUST return one result set.

3.1.5.32.1 Checked Out Files Result Set

The Checked Out Files Result Set returns information in the form defined below about the documents which are checked out. The Checked Out Files Result Set MUST return one row for each document with a **Document Store** type (section 2.2.2.4) of 0 (File) in the specified directory which is checked out and does not have a checked in draft or published version.

```
          DirName                       nvarchar(256),
          LeafName                      nvarchar(128),
          DocLibRowId                   int,
          CheckoutUserId                int,
          {tp_Title}                    nvarchar(255),
          {tp_Email}                    nvarchar(255),
          TimeLastModified              datetime,
          Size                          int;
```

DirName: The directory name of the directory containing the document(s).

LeafName: The leaf name of the document.

DocLibRowId: The row identifier for the document within the containing document library. If the document is not contained in a list, this value MUST be NULL.

CheckoutUserId: The user identifier (section 2.2.1.13) for the user who has this document checked out.

{tp_Title}: The display name of the user specified in **CheckoutUserId**. This parameter MUST NOT be NULL. If display name is unavailable, **{tp_Title}** MUST contain an empty string.

{tp_Email}: The email address of the user specified in **CheckoutUserId**. This parameter MUST NOT be NULL. If the email address is unavailable, **{tp_Email}** MUST contain an empty string.

TimeLastModified: A date/time value in **UTC** format specifying when the document was last saved.

Size: The size of the document in bytes.

3.1.5.33 proc_GetListFields

The **proc_GetListFields** stored procedure is invoked to get the mapping of fields in a list. The mapping is represented via an XML string specifying each of the fields in the list.

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

@WebId: The site identifier (section 2.2.1.11) of the site containing the list.

@ListId: The list identifier (section 2.2.1.5) of the list.

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

Return Values: The **proc_GetListFields** stored procedure returns an integer return code which MUST be 0. The **proc_GetListFields** stored procedure MUST return a single result set as follows.

3.1.5.33.1 Fields Information Result Set

The Fields Information Result Set MUST return a single row holding a single column. The Fields Information Result Set will be returned once, and MUST contain 0 or 1 rows as follows: if no list was found matching the provided *@WebId* and *@ListId* parameters, then the result set MUST contain 0 rows. Otherwise, the result set MUST contain one row.

```
      tp_Fields                      nvarchar(max);
```

tp_Fields: Contains a **WSS Compressed** structure (section 2.2.4.8). Uncompressed it contains an implementation-specific version string followed by an XML representation of the field definitions. The field definitions include display and interaction options. The XML MUST conform to the **FieldDefinitionDatabaseWithVersion** complex type, as specified in section 2.2.8.3.5.

3.1.5.34 proc_GetListMetaDataAndEventReceivers

The **proc_GetListMetaDataAndEventReceivers** stored procedure retrieves information about the metadata, security scopes, web parts, and event receivers for a specified list.

```
PROCEDURE proc_GetListMetaDataAndEventReceivers(
      @SiteId                        uniqueidentifier,
      @WebId                         uniqueidentifier,
      @ListId                        uniqueidentifier,
      @PrefetchListScope             bit                =0,
      @ThresholdScopeCount           int                =0,
      @PrefetchRelatedFields         bit                =0,
      @PrefetchWebParts              bit                =0,
      @UserId                        int                =-1,
      @CurrentFolderUrl              nvarchar(260)      =NULL,
      @RequestGuid                   uniqueidentifier   =NULL OUTPUT
```

```
    );
```

@SiteId: The site collection identifier (section 2.2.1.9) for a site collection that contains the site specified by the *@WebId* parameter.

@WebId: The site identifier (section 2.2.1.11) for the site containing the list specified by the *@ListId* parameter.

@ListId: The list identifier (section 2.2.1.5) for the list.

@PrefetchListScope: Specifies whether to include the Unique Permissions Result Set (section 3.1.5.34.2) or NULL Unique Permissions Result Set (section 3.1.5.34.3) in the returned result sets. If 1, then the Unique Permissions Result Set or NULL Unique Permissions Result Set MUST be returned. Otherwise, the Unique Permissions Result Set and NULL Unique Permissions Result Set MUST NOT be returned.

@ThresholdScopeCount: This parameter MUST be set to a value greater than 0. If the *@PrefetchListScope* parameter is not set to 0, this parameter MUST specify the maximum number of security scopes returned by the Unique Permissions Result Set.

@PrefetchRelatedFields: If this parameter is not set to 0, the List Related Fields Result Set (section 2.2.5.26) MUST be returned.

@PrefetchWebParts: Specifies whether to include the List Web Parts Result Set (section 3.1.5.34.5) in the returned result sets. If 1, then the List Web Parts Result Set MUST be returned. Otherwise, the List Web Parts Result Set MUST NOT be returned.

@UserId: This value can refer to an existing user identifier (section 2.2.1.13) for the specified site collection, and defaults to -1, specifying that the contents of the List Web Parts Result Set MUST NOT be limited to results for a particular user. When an existing user identifier is specified, the contents of the List Web Parts Result Set MUST be limited to web parts visible to the specified user.

@CurrentFolderUrl: If this value is set to NULL or an empty string this parameter MUST be ignored. Otherwise, it MUST be the root folder URL that the specified list is contained within.

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

Return Values: This stored procedure returns an integer return code, which MUST be 0.

This stored procedure MUST return between two and five result sets, depending upon input parameters. Result sets that are returned will be sent in the following order:

3.1.5.34.1 List Metadata Result Set

The List Metadata Result Set contains the metadata associated with the list. The List Metadata Result Set MUST be returned and MUST contain one row for a valid list identifier (section 2.2.1.5) by the *@ListId* parameter. If the list identifier is invalid, the List Metadata Result Set MUST contain zero rows.

The List Metadata Result Set is defined in List Metadata Result Set, section 2.2.5.12.

3.1.5.34.2 Unique Permissions Result Set

The Unique Permissions Result Set returns any security scopes associated with list items or folders within the specified list. The Unique Permissions Result Set MUST be returned when the *@PrefetchListScope* parameter is set to 1, if at least one such security scopes exists. Otherwise, the NULL Unique Permissions Result Set (section 3.1.5.34.3) MUST be returned to indicate no such security scopes exist. The Unique Permissions Result Set MUST contain one row for each security scope defined on any list item or folder within the list, but no more rows than those specified by the *@ThresholdScopeCount* parameter.

The Unique Permissions Result Set is defined in Unique Permissions Result Set, section 2.2.5.24.

3.1.5.34.3 NULL Unique Permissions Result Set

The NULL Unique Permissions Result Set indicates that no unique security scopes exist within the specified list. The NULL Unique Permissions Result Set MUST be returned if the *@PrefetchListScope* parameter is set to 1 and no security scopes are defined for any list items or folders within the specified list.

The NULL Unique Permissions Result Set is defined in NULL Unique Permissions Result Set, section 2.2.5.15.

3.1.5.34.4 List Event Receivers Result Set

The List Event Receivers Result Set contains information about the event receivers defined for this list. The List Event Receivers Result Set MUST be returned, and MUST contain one row for each event receiver that is registered for this list, or zero rows if no event receivers are registered for this list.

The List Event Receivers Result Set is defined in Event Receivers Result Set, section 2.2.5.9.

3.1.5.34.5 List Web Parts Result Set

The List Web Parts Result Set contains information about the list web parts defined for this site. The List Web Parts Result Set MUST be returned if the *@PrefetchWebParts* parameter is set to 1. The result set MUST contain one row for each web part that is registered for this list. If there are no web parts registered for this list, then this result set MUST NOT return any rows.

The List Web Parts Result Set is defined in List Web Parts Result Set, section 2.2.5.13.

3.1.5.34.6 List Related Fields Result Set

The List Related fields Result Set returns information about all of the relationship lookup fields whose target list is the specified list. The List Related fields Result Set MUST return one row for each of the relationship lookup fields whose target list is the specified list. If there are no such fields, then it MUST return zero rows.

The List Related Fields Result Set is defined in List Related Fields Result Set, section 2.2.5.26.

3.1.5.35 proc_GetNewListItemId

The **proc_GetNewListItemId** stored procedure is called to get the next available row identifier of a **list** and the number of rows that will be copied or moved when copying or moving a **URL** to that list.

```
PROCEDURE proc_GetNewListItemId(
@SiteId uniqueidentifier,
@FullUrl nvarchar(260),
@NewListId uniqueidentifier,
@bIsCopy bit,
@NewDoclibRowId int OUTPUT,
@MaxNewRows int OUTPUT
);
```

@SiteId: The **site collection identifier** of the **site collection** which contains the URL specified by @FullUrl.

@FullUrl: The **full URL** to copy or move.

@NewListId: The list identifier (section 2.2.1.5) of the target list.

@bIsCopy: A bit flag that indicates whether to copy or move the URL specified by **@FullUrl**. The value MUST be 1 to copy or MUST be zero to move.

@NewDocLibRowId: This is an output parameter whose value indicates the next available row identifier in the target list where the URL specified by **@FullUrl** is being copied or moved to. The return value MUST be obtained by calling **proc_GenerateNextId** (section 3.1.5.23) when the URL specified by **@FullUrl** is a file or folder; otherwise the value is undefined.

@MaxNewRows: This is an output parameter whose value indicates the total number of items that will be copied or moved, as specified by **@FullUrl**. The return value MUST be 1 if the URL specified by **@FullUrl** is a file; MUST be the number of items contained in the folder if the URL specified by **@FullUrl** is a folder; otherwise the value is undefined.

Return Values: The **proc_GetNewListItemId** stored procedure returns an integer return code, which MUST be one of the values listed in the following table.

Value	Description
0	Successful execution returned when all inputs are valid and all constraints are satisfied.
3	The URL specified by **@FullUrl** was not found.

The **proc_GetNewListItemId** stored procedure MUST NOT return any result sets.

3.1.5.36 proc_getObject

The **proc_getObject** stored procedure is invoked to retrieve a single Configuration Object (section 2.2.6.1) from the Configuration Database (section 3.1.1). The **proc_getObject** stored procedure is located in the Configuration Database.

```
PROCEDURE proc_getObject(
      @Id                          uniqueidentifier,
      @RequestGuid                 uniqueidentifier = NULL     OUTPUT
);
```

@Id: Contains the GUID for the **Id** of the Configuration Object to be retrieved.

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

Return Values: The **proc_getObject** stored procedure returns an integer return code, which MUST be 0. The **proc_getObject** stored procedure MUST return a single result set as follows.

3.1.5.36.1 Object Result Set

The Object Result Set returns the Configuration Object (section 2.2.6.1) identified by *@Id*. If *@Id* is NULL or does not match the value of any configuration objects in the Configuration Database (section 3.1.1), then the Object Result Set MUST be returned and MUST return zero rows.

The Object Result Set MUST be returned and MUST return one row if the Configuration Object matches the value of *@Id*.

```
      Id                           uniqueidentifier,
      ParentId                     uniqueidentifier,
      ClassId                      uniqueidentifier,
      Name                         nvarchar(128),
      Status                       int,
      Version                      rowversion,
      Properties                   nvarchar(max);
```

Id: Contains the GUID for the requested Configuration Object (section 2.2.6.1).

ParentId: Contains the GUID for the Parent (section 2.2.6.1.4) of the Configuration Object.

ClassId: Contains the GUID for the Class (section 2.2.6.1.1) of the Configuration Object.

Name: Contains the Name (section 2.2.6.1.3) of the requested Configuration Object.

Status: Contains the Status (section 2.2.6.1.5) of the Configuration Object.

Version: Contains the Version (section 2.2.6.1.6) of the Configuration Object.

Properties: Contains the properties definition of the Configuration Object, which is opaque to the back-end database server.

3.1.5.37 proc_getObjectsByBaseClass

The **proc_getObjectsByBaseClass** stored procedure is invoked to return a list of GUIDs for child Configuration Objects (section 2.2.6.1) of the specified parent Configuration Object that inherit from the specified base Class (section 2.2.6.1.1). The **proc_getObjectsByBaseClass** is located in the Configuration Database (section 3.1.1).

```
PROCEDURE proc_getObjectsByBaseClass (
      @BaseClassId                   uniqueidentifier,
      @ParentId                      uniqueidentifier,
      @RequestGuid                   uniqueidentifier = NULL     OUTPUT
);
```

@BaseClassId: Contains the GUID for the base Class of the requested Configuration Objects.

@ParentId: Contains the GUID for the Parent (section 2.2.6.1.4) Configuration Object of the requested Configuration Objects.

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

Return Values: The **proc_getObjectsByBaseClass** stored procedure returns an integer return code which MUST be 0. The **proc_getObjectsByBaseClass** stored procedure MUST return a single result set as follows.

3.1.5.37.1 Object ID Result Set

The Object ID Result Set contains the GUIDs of the matching Configuration Objects (section 2.2.6.1), if any. The Object ID Result Set MUST contain zero rows if no matching Configuration Objects are found, and MUST contain one row for each matching Configuration Object found. The Object ID Result Set is defined in Object ID Result Set, section 2.2.5.16.

3.1.5.38 proc_getObjectsByClass

The **proc_getObjectsByClass** stored procedure is invoked to retrieve the GUIDs of Configuration Objects (section 2.2.6.1) based upon the Class type (section 2.2.6.1.1), the parent (section 2.2.6.1.4) Configuration Object, and the Name (section 2.2.6.1.3) of the Configuration Object. The **proc_getObjectsByClass** is located in the Configuration Database (section 3.1.1).

```
PROCEDURE proc_getObjectsByClass(
      @ClassId                       uniqueidentifier,
      @ParentId                      uniqueidentifier,
      @Name                          nvarchar(128),
      @RequestGuid                   uniqueidentifier = NULL     OUTPUT
```

```
    );
```

@ClassId: Contains the GUID of the Class type of the requested Configuration Objects. MUST NOT be NULL.

@ParentId: Contains the GUID for the Parent Configuration Object of the requested Configuration Objects.

@Name: Contains the Name of the requested Configuration Object.

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

Return Values: The **proc_getObjectsByClass** stored procedure returns an integer return code which MUST be listed in the following table.

Value	Description
0	Successful execution.
50105	Error: Class does not exist.

The **proc_getObjectsByClass** stored procedure MUST return a single result set as follows.

3.1.5.38.1 Object ID Result Set

The Object ID Result Set contains the GUIDs of the matching Configuration Object (section 2.2.6.1), if any. The Object ID Result Set MUST contain zero rows if no matching Configuration Object are found. A Configuration Object MUST match if *@ParentId* is NULL or *@ParentId* is equal to the GUID of the Parent (section 2.2.6.1.4) of the Configuration Object. In addition, a Configuration Object MUST match if *@Name* is null OR *@Name* is equal to the Name of the Configuration Object. The Object ID Result Set is defined in Object ID Result Set, section 2.2.5.16.

3.1.5.39 proc_GetSiteFlags

The **proc_GetSiteFlags** stored procedure is invoked to return the **Site Collection Flags** (section 2.2.2.9) set for a specified site collection.

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

@WebSiteId: The site collection identifier (section 2.2.1.9) of the site collection for which to retrieve **Site Collection Flags**.

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

Return Values: This stored procedure MUST return an integer return code of 0.

This stored procedure MUST return the following result set.

3.1.5.39.1 Site Collection Flags Result Set

The **Site Collection Flags Result Set** returns the **Site Collection Flags** (section 2.2.2.9) for a specified site collection. The Site Flags Result Set will be returned, and MUST contain one row.

```
        {SiteCollectionFlags}          int;
```

{SiteCollectionFlags}: Contains the **Site Collection Flags** for the specified site collection. This MUST be NULL if the site collection specified by the *@WebSiteId* parameter does not exist, is NULL, or is a **NULL GUID**.

3.1.5.40 proc_getSiteMap

The **proc_getSiteMap** stored procedure is invoked to determine the Site Map (section 3.1.5.40.1) information for a site collection. **proc_getSiteMap** is located in the Configuration Database (section 3.1.1).

```
    PROCEDURE proc_getSiteMap(
          @ApplicationId                uniqueidentifier,
          @Path                         nvarchar(128),
          @RequestGuid                  uniqueidentifier = NULL     OUTPUT
    );
```

@ApplicationId: The **web application** identifier containing the site collection specified by the *@Path* parameter.

@Path: The server-relative path to the root of the site collection. Set *@Path* to '/' to specify a site collection which is the root of the web application.

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

Return Values: The **proc_getSiteMap** stored procedure returns an integer return code, which MUST be 0. The **proc_getSiteMap** stored procedure MUST return a single result set.

3.1.5.40.1 Site Map Result Set

The Site Map Result Set returns the Site Map information for the site collection specified by the *@ApplicationId* and *@Path* parameter values. The Site Map Result Set MUST return zero rows if the parameters do not match any Site Map information; it MUST return one row if a match is found.

```
          Id                            uniqueidentifier,
          SubscriptionId                uniqueidentifier,
          DatabaseId                    uniqueidentifier,
          RedirectUrl                   nvarchar(512),
          Pairing                       tinyint;
```

Id: Contains the site collection identifier (section 2.2.1.9) for the requested site collection.

SubscriptionId: The GUID of the implementation-specific subscription feature for the requested site collection.

DatabaseId: Contains the GUID of the database containing the content of the requested site collection.

RedirectUrl: The URL used to access the specified site collection during an implementation-specific upgrade operation. This value MUST be NULL if the site collection upgrade to the current implementation-specific version is complete.

Pairing: Contains an implementation-specific integer value denoting that a given object is paired with a matching object of a previous version. This parameter is used during a implementation-specific upgrade operation.

Value	Description
0	This object is not paired.
1	This object is paired with an object from a previous major version of the product.

3.1.5.41 proc_getSiteMapById

The **proc_getSiteMapById** stored procedure is invoked to determine complete Site Map (section 3.1.5.40.1) information for a site collection. **proc_getSiteMapById** is located in the Configuration Database (section 3.1.1).

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

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection to get Site Map information for.

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

Return Values: The **proc_getSiteMapById** stored procedure returns an integer return code which MUST be 0. The **proc_getSiteMapById** stored procedure MUST return a single result set.

3.1.5.41.1 Site Map By Id Result Set

The Site Map By Id Result Set returns information about the **web application**, database, and URL mapped to the specified site collection. The Site Map By Id Result Set MUST contain one row if the specified site collection identifier (section 2.2.1.9) exists in the Configuration Database (section 3.1.1) on the back-end database server. Otherwise, it MUST contain zero rows.

```
          ApplicationId                  uniqueidentifier,
          DatabaseId                     uniqueidentifier,
          Id                             uniqueidentifier,
          SubscriptionId                 uniqueidentifier,
          Path                           nvarchar(128),
          RedirectUrl                    nvarchar(512),
          Pairing                        tinyint,
          HostHeaderIsSiteName           bit;
```

ApplicationId: The web application identifier hosting the content of the requested site collection.

DatabaseId: The GUID of the database containing the content of the requested site collection.

Id: The site collection identifier (section 2.2.1.9) for the requested site collection.

SubscriptionId: The GUID of the implementation-specific subscription for the requested site collection.

Path: The URL used to identify the requested site collection. For site collections that do not use a specifically-configured host domain, the **HostHeaderIsSiteName** field MUST be 0 and the **Path** value MUST be the server-relative URL of the root site of the site collection. For site collections using **host header** mode (that is, site collections identified by host header identifier rather than by server-relative URL path), the **HostHeaderIsSiteName** field MUST be 1 and the **Path** value for the site collection MUST be the **host name**, and MUST include the port number for hosts at non-default ports, and can include the port number for hosts at default ports.

RedirectUrl: The URL used to access the specified site collection during a implementation-specific upgrade operation. This value MUST be NULL if the site collection upgrade to the current implementation-specific version number is complete.

Pairing: An integer value used during an implementation-specific upgrade operation to denote that a given object is paired with a matching object of a previous implementation-specific version number. The following are valid values.

Value	Description
0	This object is not paired.
1	This object is paired with an object from a previous major version of the product.

HostHeaderIsSiteName: This value MUST be set to 1 if the requested site collection uses host header mode; otherwise it MUST be 0. The Path field MUST contain the URL for the specified **HostHeaderIsSiteName** value.

3.1.5.42 proc_GetTpWebMetaDataAndListMetaData

The **proc_GetTpWebMetaDataAndListMetaData** stored procedure retrieves metadata for a particular site or list.

```
PROCEDURE proc_GetTpWebMetaDataAndListMetaData(
      @WebSiteId                      uniqueidentifier,
      @WebId                          uniqueidentifier,
      @Url                            nvarchar(260),
      @ListId                         uniqueidentifier,
      @ViewId                         uniqueidentifier,
      @RunUrlToWebUrl                 bit,
      @DGCacheVersion                 bigint,
      @SystemId                       varbinary(512)      = NULL,
      @MetadataFlags                  int                 = 0,
      @ThresholdScopeCount            int                 = 0,
      @CurrentFolderUrl               nvarchar(260)       = NULL,
      @RequestGuid                    uniqueidentifier    = NULL OUTPUT
);
```

@WebSiteId: The site collection identifier (section 2.2.1.9) for a site collection containing the site.

@WebId: The site identifier (section 2.2.1.11) of the site for which metadata is requested, or of the site containing the list for which metadata is requested. This parameter is ignored if the *@MetadataFlags* has the **METADATA_WEB** flag set and if values in *@WebSiteId* and *@URL* are valid.

@Url: The store-relative form URL of the site or list to retrieve metadata for.

@ListId: The list identifier (section 2.2.1.5) of the List for which metadata is requested. This parameter is optional and MUST be NULL if no List metadata is required.

@ViewId: The web part identifier (section 2.2.1.15) that is registered for the list specified by the *@ListId* parameter. If this parameter is set to NULL, then the **tp_AllUsersProperties**, **tp_PerUserProperties**, and **tp_Cache** columns returned by the List Web Parts Result Set (section 2.2.5.13) will be set to NULL.

@RunUrlToWebUrl: A bit flag specifying whether the *@URL* parameter is for a site or for a list within a site. If this value is set to "1", the value in *@URL* is for a list, and MUST be converted to a store-relative form URL for the containing site.

@DGCacheVersion: The version number of the Domain Group Map Cache in the front-end Web server. This can be compared with the Domain Group Map Cache version number on the back-end database server returned in the Domain Group Cache Versions Result Set (section 3.1.5.42.2) to determine whether updates are needed. A *@DGCacheVersion* value of -2 specifies that information about the Domain Group Map Cache is not requested, and the Domain Group Cache BEDS Update Result Set (section 3.1.5.42.3) and the Domain Group Cache WFE Update Result Set (section 3.1.5.42.4) MUST NOT be returned.

@SystemId: The **SystemID** (section 2.2.1.12) of the current user requesting the metadata.

@MetadataFlags: A bit field used to determine the types of metadata returned by this procedure. The only valid values of the Metadata Flags bits are specified as follows.

Symbolic name, Value	Description
METADATA_NONE, 0x00000000	Do not provide any metadata.
METADATA_FETCH_VIEWS, 0x00000001	Retrieve list views. This flag is set in combination with METADATA_LISTMETADATA_NOFETCH and the *@ListID* variable and will result in the List Web Parts Result Set.
METADATA_PREFETCH_SCOPES, 0x00000002	Retrieve security scopes. If this flag is set and *@ListId* is not NULL and METADATA_LISTMETADATA_NOFETCH is not set, then the List Unique Permissions Result Set (section 3.1.5.42.18) will be returned.
METADATA_SAVE_SCOPE_DATA, 0x00000004	Save scope data. This flag is currently unused.
METADATA_FIGURE_TYPE, 0x00000008	Setting this flag in combination with NULL **ListID** will result in retrieving the **ListID** values directly from the **AllDocs** table (section 2.2.7.1).
METADATA_WEB, 0x00000010	Retrieve information pertaining to the provided site information. This flag is used in combination with other flags to further define the set or sets of information to be retrieved.
METADATA_URL, 0x00000020	Setting this flag will populate the ListId column for all results, including those which are not list items. When used in combination with a valid *@WebSiteId*, if *@ListId* is NULL or doesn't match the list contained within the site identified by *@WebSiteId*, the Document Metadata Result Set (section 3.1.5.42.27) is returned.
METADATA_USERINFOLIST, 0x00000040	Return user information. Setting this flag when METADATA_USERINFOLIST_NOFETCH is not set opens the option for List Metadata Result Set (section 2.2.5.12), List Unique Permissions Result Set, Event Receivers Result Set (section 2.2.5.9), and the List Web Parts Result Set.
METADATA_USERINFOLIST_FULL, 0x00000080	Return full user information. Setting this flag in combination with METADATA_USERINFOLIST when METADATA_USERINFOLIST_NOFETCH, is not set, along with a valid site identifier in *@WebSiteId*, where *@ListId* is NULL or doesn't match a list identifier contained within the site specified by *@WebSiteId*, returns the Event Receivers Result Set and the List Web Parts Result Set.
METADATA_WEB_PROP, 0x00000100	Return site data. Setting this flag in combination with METADATA_WEB will retrieve the Site MetaInfo Result Set (section 3.1.5.42.8).

Symbolic name, Value	Description
METADATA_WEB_FEATURES, 0x00000200	Return site features. Setting this flag in combination with METADATA_WEB will retrieve the Site Feature List Result Set (section 3.1.5.42.9).
METADATA_WEB_NAVSTRUCT, 0x00000400	Return navigation structure. Setting this flag in combination with METADATA_WEB will retrieve the Site Unique Permissions Result Set (section 3.1.5.42.10).
METADATA_LISTMETADATA_NOFETCH, 0x00000800	Do not return list metadata. Setting this flag will suppress the retrieval of the List Metadata Result Set, Unique Permissions Result Set, and the Event Receivers Result Set using the list identification parameters.
METADATA_USERINFOLIST_NOFETCH, 0x00001000	Do not return user information. Setting this flag will suppress the retrieval of the List Metadata Result Set, Unique Permissions Result Set (section 3.1.5.42.23), Event Receivers Result Set, and the List Web Parts Result Set using the user identification parameters.
METADATA_URL_REALLY, 0x00002000	Return single document metadata. Setting this flag will return the Document Metadata Result Set.
METADATA_WEB_WELCOMEPAGE, 0x00004000	Redirect welcome page. Setting this flag in combination with METADATA_WEB will retrieve the Redirect URL Result Set.
METADATA_PREFETCH_RELATEDFIELDS, 0x00008000	Return the List Related Fields Result Set (section 3.1.5.42.17) if the *@MetadataFlags* parameter does not contain METADATA_LISTMETADATA_NOFETCH.

@ThresholdScopeCount: This parameter MUST be set to a value greater than 0. If the *@MetadataFlags* parameter does not contain METADATA_LISTMETADATA_NOFETCH and contains METADATA_PREFETCH_SCOPES, this parameter MUST specify the maximum number of security scopes returned by the List Unique Permissions Result Set.

@CurrentFolderUrl: If this value is set to NULL or an empty string, this parameter MUST be ignored. Otherwise, it MUST be the root folder URL that the specified list is contained within.

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

Return Values: The **proc_GetTpWebMetaDataAndListMetaData** stored procedure returns an integer return code, which MUST be in the following table.

Value	Description
0	Successful operation.
1	Cross Site attack detected.
2	The attempt to retrieve a redirected URL failed.
3	The specified URL was not found.
1168	The site collection specified by *@WebSiteId* was not found.
1271	Access to the site is blocked because the site collection is locked.

The **proc_GetTpWebMetaDataAndListMetaData** stored procedure returns up to 27 result sets, depending on the state of the input parameters.

3.1.5.42.1 Web Url Result Set

The Web Url Result Set contains the store-relative form URL of the root of the Site which contains the *@Url* parameter.

The Web Url Result Set MUST be returned when *@MetadataFlags* has the METADATA_WEB flag set, *@RunUrlToWebUrl* is set to 1, and *@WebSiteId* is a valid site collection identifier (section 2.2.1.9), and MUST contain one row of data. If *@MetadataFlags* has the METADATA_WEB flag set, *@RunUrlToWebUrl* is set to 1, and *@WebSiteId* is not found, **proc_GetTpWebMetaDataAndListMetaData** MUST NOT return any result sets.

```
          {WebUrl}                      nvarchar(260);
```

{WebUrl}: The store-relative form URL of the site containing the *@Url* parameter.

3.1.5.42.2 Domain Group Cache Versions Result Set

The Domain Group Cache Versions Result Set contains information about the version numbers associated with the Domain Group Map Caches on the front-end web server and on the back-end database server for the specified site collection.

The Domain Group Cache Versions Result Set MUST be returned when *@Url* specifies a valid location within or of a site and *@MetadataFlags* has the METADATA_WEB flag set, and MUST contain one row of version number data. If *@MetadataFlags* has the METADATA_WEB flag set, and *@Url* does not specify either a valid site, or if *@RunUrlToWebUrl* is set to 1, a valid location within a site, then **proc_GetTpWebMetaDataAndListMetaData** MUST NOT return this or any further result sets.

If the specified *@DGCacheVersion* value is -2, then all columns returned MUST have the value -2, indicating that the value MUST NOT be used for comparison. The Domain Group Cache Versions Result Set is defined in Domain Group Cache Versions Result Set, section 2.2.5.4.

3.1.5.42.3 Domain Group Cache BEDS Update Result Set

The Domain Group Cache BEDS Update Result Set contains information to be used in recomputing the domain group map cache, which contains the mapping of domain groups to the **security groups** they are members of. The presence of the Domain Group Cache BEDS Update Result Set means the database's copy of the domain group map cache is out of date and MUST be recomputed to ensure that proper security checks can be made.

The Domain Group Cache BEDS Update Result Set MUST be returned only if the Domain Group Cache Versions Result Set is returned, *@DGCacheVersion* does not contain "-2", and the value of **RealVersion** is greater than the **CachedVersion** in the results in the Domain Group Cache Versions Result Set (section 2.2.5.4).

The Domain Group Cache BEDS Update Result Set schema is defined in Domain Group Cache BEDS Update Result Set, section 2.2.5.3.

3.1.5.42.4 Domain Group Cache WFE Update Result Set

The Domain Group Cache WFE Update Result Set contains the binary data needed to refresh the domain group map cache. The presence of the Domain Group Cache WFE Update Result Set means the domain group map cache on the database is up to date, and the front-end web server cache can be refreshed if necessary.

The Domain Group Cache WFE Update Result Set MUST be returned only if the Domain Group Cache Versions Result Set is returned, and *@DGCacheVersion* does not contain "-2", and the value of **RealVersion** is less than or equal to **CachedVersion** in the results in Domain Group Cache Versions

Result Set (section 2.2.5.4). The Domain Group Cache WFE Update Result Set MUST contain one row of data.

The Domain Group Cache WFE Update Result Set schema is defined in Domain Group Cache WFE Update Result Set, section 2.2.5.5.

3.1.5.42.5 Site Metadata Result Set

The Site Metadata Result Set contains specialized site metadata. The Site Metadata Result Set MUST be returned when the Domain Group Cache Versions Result Set (section 2.2.5.4) is returned, and it MUST contain one row.

The Site Metadata Result Set is defined in Site Metadata Result Set, section 2.2.5.22.

3.1.5.42.6 Site Collection Event Receivers Result Set

The Site Collection Event Receivers Result Set contains information about the event receivers defined for the site collection specified by the *@WebSiteId* parameter. The Site Collection Event Receivers Result Set MUST be returned when execution is successful, and the List Metadata Result Set has been returned.

The Site Collection Event Receivers Result Set MUST contain one row for each event receiver with an **Event Host** type (section 2.2.3.5) of 0 registered for the site collection specified by the *@WebSiteId* parameter.

The Site Collection Event Receivers Result Set is defined in section Event Receivers Result Set, section 2.2.5.9.

3.1.5.42.7 Site Event Receivers Result Set

The Event Receivers Result Set contains information about the event receivers defined for this site. The Event Receivers Result Set MUST be returned when the Domain Group Cache Versions Result Set is returned, and it MUST contain one row for each event receiver registered for this site. If this Event Receivers Result Set is returned, and the site collection specified by *@WebSiteId* is locked, then **proc_GetTpWebMetaDataAndListMetaData** MUST NOT return any further result sets.

The Event Receivers Result Set is defined in Event Receivers Result Set, section 2.2.5.9.

3.1.5.42.8 Site MetaInfo Result Set

The Site MetaInfo Result Set contains the METADICT of the site. The Site MetaInfo Result Set MUST be returned when execution is successful, and *@MetadataFlag* has the METADATA_WEB flag set and the METADATA_WEB_PROP flag set, and MUST contain one row if the *@WebId* parameter is valid. If the *@MetadataFlag* has the METADATA_WEB flag set and the METADATA_WEB_PROP flag set, and the site specified by *@WebId* does not exist, then the result set MUST contain no rows, and **proc_GetTpWebMetaDataAndListMetaData** MUST NOT return any further result sets.

The Site MetaInfo Result Set is defined in Site MetaInfo Result Set, section 2.2.5.23.

3.1.5.42.9 Site Feature List Result Set

The Site Feature List Result Set contains the list of feature identifiers (section 2.2.1.4) of the site. The Site Feature List Result Set MUST be returned and MUST contain one row for each feature when execution is successful, and all of the following conditions are met:

- *@MetadataFlag* has the METADATA_WEB flag set
- *@MetadataFlag* has the METADATA_WEB_FEATURES flag set

- *@WebSiteId* is valid
- *@WebId* is valid

The Site Feature List Result Set MUST be returned twice with the same schema, once for the default features for the site collection, and again with the features for the specified site.

The Site Feature List Result Set is defined in Site Feature List Result Set, section 2.2.5.21.

3.1.5.42.10 Site Unique Permissions Result Set

The Unique Permissions Result Set contains security scope and WSS ACL (section 2.2.4.6) information of the site specified by *@WebId*. The Unique Permissions Result Set MUST be returned when execution is successful, and all of the following conditions are met:

- *@MetadataFlags* has the METADATA_WEB flag set.
- *@MetadataFlags* has the METADATA_WEB_NAVSTRUCT flag set.
- The specified site contains cached navigation scope information.

The Unique Permissions Result Set is defined in Unique Permissions Result Set, section 2.2.5.24.

3.1.5.42.11 Site NULL Unique Permissions Result Set

The NULL Unique Permissions Result Set MUST be returned when execution is successful, and all of the following conditions are met:

- *@MetadataFlags* has the METADATA_WEB flag set.
- *@MetadataFlags* has the METADATA_WEB_NAVSTRUCT flag set.
- The site specified by *@WebId* or its parent site's cached navigation scope is over 900 characters long.

The NULL Unique Permissions Result Set is defined in NULL Unique Permissions Result Set, section 2.2.5.15.

3.1.5.42.12 Empty Result Set

An Empty Result Set containing no rows MUST be returned when execution is successful, and all of the following conditions are met:

- *@MetadataFlags* has the METADATA_WEB flag set.
- *@MetadataFlags* has the METADATA_WEB_NAVSTRUCT flag set.
- The site specified by *@WebId* does not contain cached navigation scope information, or the site is marked as **dirty**.

The Empty Result Set is defined in Empty Result Set, section 2.2.5.8.

3.1.5.42.13 Redirect Url Result Set

The Redirect Url Result Set contains redirect URL information generated from the *@Url* parameter. If execution is successful, and *@MetadataFlags* has the METADATA_WEB flag set and the METADATA_WEB_WELCOMEPAGE flag set, and the site location specified by *@Url* has a **Redirect** type (section 2.2.3.15) of 255 (None), then **proc_GetTpWebMetaDataAndListMetaData** MUST NOT return this or any further result sets. Otherwise, the Redirect Url Result Set MUST be returned when execution is successful, and all of the following conditions are met:

- *@MetadataFlags* has the METADATA_WEB flag set.
- *@MetadataFlags* has the METADATA_WEB_WELCOMEPAGE flag set.
- The site location has been marked as a welcome page.

The Redirect Url Result Set MUST contain one row with the redirect URL and related information for the specified site location.

```
          {RedirectType}                  tinyint,
          {RedirectUrl}                   nvarchar(260),
          WelcomePageParameters           nvarchar(max);
```

{RedirectType}: Type of item this URL is redirected to. See **Redirect** type (section 2.2.3.15) for all possible values for this column.

{RedirectUrl}: The **store-relative URL** generated from the provided *@Url* parameter.

WelcomePageParameters: The URL parameters for the welcome page found with this redirect URL. This column can contain a query string starting with "?" or a hash parameter starting with "#" (or both).

3.1.5.42.14 No Welcome Redirect Url Result Set

The No Welcome Redirect Url Result Set contains redirect URL information generated from the *@Url* parameter. The No Welcome Redirect Url Result Set MUST be returned when execution is successful, and all of the following conditions are met:

- *@MetadataFlags* has the METADATA_WEB flag set.
- *@MetadataFlags* has the METADATA_WEB_WELCOMEPAGE flag set.
- The site location specified by *@Url* has not been marked as a welcome page, that is, it has a **RedirectType** that is not 0.

The No Welcome Redirect Url Result Set will return exactly one row with the Redirect URL for this folder.

```
          {RedirectType}                  tinyint,
          {RedirectUrl}                   nvarchar(260),
          {WelcomePageParameters}         nvarchar(max);
```

{RedirectType}: Type of item this URL is redirected to. See the **Redirect** type (section 2.2.3.15) for all possible values for this column.

{RedirectUrl}: The complete redirect URL generated from the provided *@Url* parameter.

{WelcomePageParameters}: This MUST be NULL and can be ignored.

3.1.5.42.15 List Identifier Result Set

The List Identifier Result Set contains identifying information for the file, list or list item specified by *@Url*. If execution is successful, and *@ListId* is NULL, *@MetadataFlags* has the METADATA_FIGURE_TYPE flag set, and the site containing *@Url* is not the site specified by *@WebId*, then **proc_GetTpWebMetaDataAndListMetaData** MUST NOT return this or any further result sets. Otherwise, the List Identifier Result Set MUST be returned when execution is successful, and all of the following conditions are met:

- *@ListId* is NULL and
- *@MetadataFlags* has the METADATA_FIGURE_TYPE flag set and
- *@WebId* is a valid site and contains the file, list or list item specified by *@Url*

```
{ListIdSelected}                 uniqueidentifier,
{TypeSelected}                   int,
{ItemIdSelected}                 int;
```

{ListIdSelected}: The list identifier (section 2.2.1.5) of the specified list, or of the list containing the specified list item. This value MUST be NULL if *@Url* does not specify a list or list item.

{TypeSelected}: Item type value for the specified list. The value of **TypeSelected** MUST be one of the **RedirectType** values specified in **Redirect** type (section 2.2.3.15).

{ItemIdSelected}: Document Library Row Identifier of the list item taken from the **Docs View** (section 2.2.7.4) for the list. This value MUST be NULL if *@Url* does not specify a list item.

3.1.5.42.16 List Metadata Result Set (1)

The List Metadata Result Set contains the metadata associated with the list specified by *@ListId* or if *@ListId* is NULL, the containing List specified by *@Url*. The List Metadata Result Set MUST be returned when execution is successful, and *@ListId* specifies a list or *@Url* specifies a list or list item, and *@MetadataFlags* does not have the METADATA_LISTMETADATA_NOFETCH flag set. The List Metadata Result Set MUST contain one row if *@WebId* specifies a valid site containing the specified list.

The List Metadata Result Set is defined in List Metadata Result Set, section 2.2.5.12.

3.1.5.42.17 List Related Fields Result Set

The List Related Fields Result Set returns information about all the relationship lookup fields whose target list is the specified list. The List Related Fields Result Set MUST return one row for each of the relationship lookup fields whose target list is the specified list. If there are no such fields, then it MUST NOT return any rows.

The List Related Fields Result Set is defined in section List Related Fields Result Set, section 2.2.5.26.
.

3.1.5.42.18 List Unique Permissions Result Set

The Unique Permissions Result Set contains all security scope and WSS ACL (section 2.2.4.6) information from the specified list. The Unique Permissions Result Set MUST be returned when execution is successful, the List Metadata Result Set (section 2.2.5.12) has been returned, and all of the following conditions are met:

- *@MetadataFlags* has the METADATA_PREFETCH_SCOPES flag set and
- *@MetadataFlags* has the METADATA_LISTMETADATA_NOFETCH flag not set

The Unique Permissions Result Set is defined in Unique Permissions Result Set, section 2.2.5.24.

3.1.5.42.19 List NULL Unique Permissions Result Set

The NULL Unique Permissions Result Set MUST be returned when execution is successful, the List Metadata Result Set has been returned, and all of the following conditions are met:

- @MetadataFlags has the METADATA_PREFETCH_SCOPES flag set.

- @MetadataFlags has the METADATA_LISTMETADATA_NOFETCH flag not set.
- Permissions associated with the specified list do not exist.

The NULL Unique Permissions Result Set is defined in NULL Unique Permissions Result Set, section 2.2.5.15.

3.1.5.42.20 Event Receivers Result Set (1)

The Site Event Receivers Result Set contains information about the event receivers defined for the site containing the list specified by the *@ListId* parameter, or if the *@ListId* parameter is NULL, the containing list specified by the *@Url* parameter. The Site Event Receivers Result Set MUST be returned when execution is successful, and the List Metadata Result Set has been returned.

The Site Event Receivers Result Set MUST contain one row for each event receiver with an **Event Host** type (section 2.2.3.5) of 2 registered for the site specified by *@WebId*.

The Site Event Receivers Result Set is defined in Event Receivers Result Set, section 2.2.5.9.

3.1.5.42.21 List Web Parts Result Set (1)

The List Web Parts Result Set contains information about the List Web Parts defined for the list specified by *@ListId*, or if *@ListId* is NULL, the containing list specified by *@Url*. The List Web Parts Result Set MUST be returned when execution is successful, the List Metadata Result Set has been returned, and *@MetadataFlags* has the METADATA_FETCH_VIEWS flag set.

The List Web Parts Result Set MUST contain one row for each Web Part registered for this List.

The List Web Parts Result Set is defined in List Web Parts Result Set, section 2.2.5.13.

3.1.5.42.22 List Metadata Result Set (2)

The List Metadata Result Set contains the metadata associated with a specified **user information list**. The List Metadata Result Set MUST be returned when execution is successful, and all of the following conditions are met:

- *@WebSiteId* contains a valid site collection identifier (section 2.2.1.9) and the site collection has a user information list.
- The list identifier (section 2.2.1.5) specified by *@ListId*, or if *@ListId* is NULL, the list identifier of the list at (or that contains) the location specified by *@Url* doesn't match the list identifier for the user information list.
- *@MetadataFlags* has the METADATA_USERINFOLIST flag set, or *@Url* specifies a list or a location within a list, and.
- *@MetadataFlags* doesn't have the METADATA_USERINFOLIST_NOFETCH flag set.

The List Metadata Result Set MUST contain one row if it is returned.

The List Metadata Result Set is defined in List Metadata Result Set, section 2.2.5.12.

3.1.5.42.23 Unique Permissions Result Set

The Unique Permissions Result Set contains all security scope and WSS ACL (section 2.2.4.6) information from a specified user information list. The Unique Permissions Result Set MUST be returned when execution is successful, and all of the following conditions are met:

- *@WebSiteId* contains a valid site collection identifier (section 2.2.1.9) and the site collection has a user information list.

- The list identifier (section 2.2.1.5) specified by *@ListId*, or if *@ListId* is NULL, the list identifier of the list at (or that contains) the location specified by *@Url* doesn't match the list identifier for the user information list.
- *@MetadataFlags* has the METADATA_USERINFOLIST flag set, or *@Url* specifies a list or a location within a list, and
- *@MetadataFlags* doesn't have the METADATA_USERINFOLIST_NOFETCH flag set.
- *@MetadataFlags* has the METADATA_PREFETCH_SCOPES flag set.

The Unique Permissions Result Set is defined in Unique Permissions Result Set, section 2.2.5.24.

3.1.5.42.24 NULL Unique Permissions Result Set

The NULL Unique Permissions Result Set MUST be returned when execution is successful, and all of the following conditions are met:

- *@WebSiteId* contains a valid site collection identifier (section 2.2.1.9) and the site collection has a user information list.
- The list identifier (section 2.2.1.5) specified by *@ListId*, or if *@ListId* is NULL, the list identifier of the list at (or that contains) the location specified by *@Url* doesn't match the list identifier for the user information list.
- *@MetadataFlags* has the METADATA_USERINFOLIST flag set, or *@Url* specifies a list or a location within a list, and
- *@MetadataFlags* doesn't have the METADATA_USERINFOLIST_NOFETCH flag set.
- *@MetadataFlags* has the METADATA_PREFETCH_SCOPES flag set.
- No unique permissions exist for the root site of the site collection.

The NULL Unique Permissions Result Set is defined in the NULL Unique Permissions Result Set, section 2.2.5.15.

3.1.5.42.25 Event Receivers Result Set (2)

The Event Receivers Result Set contains information about the event receivers defined for the site containing the specified user information list. The Event Receivers Result Set MUST be returned when execution is successful, and all of the following conditions are met:

- *@WebSiteId* contains a valid site collection identifier (section 2.2.1.9) and the site collection has a user information list.
- The list identifier (section 2.2.1.5) specified by *@ListId*, or if *@ListId* is NULL, the list identifier of the list at (or that contains) the location specified by *@Url* doesn't match the list identifier for the user information list.
- *@MetadataFlags* has the METADATA_USERINFOLIST flag set, or *@Url* specifies a list or a location within a list.
- *@MetadataFlags* has the METADATA_USERINFOLIST_FULL flag set.
- *@MetadataFlags* doesn't have the METADATA_USERINFOLIST_NOFETCH flag set.

The Event Receivers Result Set MUST contain one row for each event receiver with an **Event Host** type (section 2.2.3.5) of 2 (List) that is registered for the root site of the site collection specified by *@WebSiteId*.

The Event Receivers Result Set is defined in Event Receivers Result Set, section 2.2.5.9.

3.1.5.42.26 List Web Parts Result Set (2)

The List Web Parts Result Set contains information about the Web Parts defined for the specified user information list. The List Web Parts Result Set MUST be returned if execution is successful, and all the following conditions are **met:**

- *@WebSiteId* contains a valid site collection identifier (section 2.2.1.9) and the site collection has a user information list.
- The list identifier (section 2.2.1.5) specified by *@ListId*, or if *@ListId* is NULL, the list identifier of the list at (or that contains) the location specified by *@Url* doesn't match the list identifier for the user information list.
- *@MetadataFlags* has the METADATA_USERINFOLIST flag set, or *@Url* specifies a list or a location within a list.
- *@MetadataFlags* has the METADATA_USERINFOLIST_FULL flag set.
- *@MetadataFlags* doesn't have the METADATA_USERINFOLIST_NOFETCH flag set.

The List Web Parts Result Set MUST contain one row for each Web Part registered for this List.

The List Web Parts Result Set is defined in List Web Parts Result Set, section 2.2.5.13.

3.1.5.42.27 Document Metadata Result Set

The Document Metadata Result Set contains the metadata for a single document when the provided *@Url* parameter refers to a single document. The Document Metadata Result Set MUST be returned when execution is successful, and either of the following conditions sets are met:

- *@MetadataFlags* has the METADATA_URL flag set.
- The document is not contained within a list.

  OR

- *@MetadataFlags* has the METADATA_URL_REALLY flag set.

The Document Metadata Result Set MUST return one row when *@Url* is valid. The Document Metadata Result Set MUST be ordered by the **Id** column, and is defined in Document Metadata Result Set, section 2.2.5.6.

3.1.5.42.28 NULL Result Set

The NULL Result Set returns no data. The NULL Result Set MUST be returned if the Document Metadata Result Set (section 2.2.5.6) is not empty, otherwise the NULL Result Set MUST NOT be returned. The NULL Result Set MUST return zero rows in a schema containing a single unnamed column.

3.1.5.43 proc_GetUniqueScopesInList

The **proc_GetUniqueScopesInList** stored procedure is invoked to get the **WSS ACL Format** (section 2.2.4.6) information for all the unique security scopes of folders and list items contained in a specified list.

```
PROCEDURE proc_GetUniqueScopesInList(
      @SiteId                         uniqueidentifier,
      @WebId                          uniqueidentifier,
      @ListId                         uniqueidentifier,
      @GetAll                         bit,
      @ThresholdRowCount              int,
      @RequestGuid                    uniqueidentifier   =NULL      OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the list.

@WebId: The site identifier (section 2.2.1.11) of the site that contains the list.

@ListId: The list identifier (section 2.2.1.5) of the list for which the **WSS ACL Format** information is requested.

@GetAll: A bit specifying whether to fetch **WSS ACL Format** information for all unique security scopes. If this bit is set to "0", **WSS ACL Format** information for most *@ThresholdRowCount* security scopes MUST be returned in the Unique Permissions Result Set (section 2.2.5.24), else **WSS ACL Format** information for all unique security scopes MUST be returned in the Unique Permissions Result Set.

@ThresholdRowCount: This parameter specifies the number of security scopes to return when *@GetAll* is set to "0". The value MUST NOT be 0.

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

Return Values: The **proc_GetUniqueScopesInList** stored procedure returns an integer return code which MUST be 0.

The **proc_GetUniqueScopesInList** stored procedure MUST return a single result set.

3.1.5.43.1 Unique Permissions Result Set

The Unique Permissions Result Set returns the **WSS ACL Format** (section 2.2.4.6) information for all the unique security scopes of folders and list items contained in a specified list. The Unique Permissions Result Set MUST be returned if the list has 1 or more folders or list items with unique security scopes. The Unique Permissions Result Set MUST return one row for every folder or list item with a unique security scope. If the *@GetAll* bit is set to "0", then this result set MUST return at most *@ThresholdRowCount* rows.

The Unique Permissions Result Set is defined in Unique Permissions Result Set, section 2.2.5.24.

3.1.5.43.2 NULL Unique Permissions Result Set

The NULL Unique Permissions Result Set MUST return one row if the list has no folders or list items with unique security scopes. The NULL Unique Permissions Result Set is defined in NULL Unique Permissions Result Set, section 2.2.5.15.

3.1.5.44 proc_GetVersion

The **proc_GetVersion** stored procedure is invoked to get the component version number string associated with the specified version identifier GUID (using the format major.minor.phase.build; for example, 3.0.106.0).

```
  PROCEDURE proc_GetVersion(
        @VersionId                      uniqueidentifier,
        @Version                        nvarchar(64)       OUTPUT
  );
```

@VersionId: The version identifier of the component version number to retrieve.

@Version: An output parameter containing the component version number string matching the specified version identifier. This value MUST be unchanged from the input value if the specified *@VersionId* value does not have a matching component version number.

Return Values: The **proc_GetVersion** stored procedure returns an integer return code which MUST be listed in the following table.

Value	Description
0	Successful execution.

The front-end web server can define one or more version numbers. Each of the version numbers is identified by a Version ID. When a database is created by the front-end web server on the back-end database server, the front-end web server stores the version numbers with different version identifiers in the **Versions** table (section 2.2.7.11). When the front-end web server connects to a back-end database server, the front-end web server SHOULD retrieve the Version IDs using stored procedure **proc_GetVersion**<15> and make sure the versions are within an acceptable range defined by the front-end web server; otherwise, the front-end web server MUST refuse to connect to the back-end database server.

The version identifiers and the acceptable version numbers for this protocol used when communicating with a content database are listed in the following table.

Version Id GUID	Acceptable version range
"00000000-0000-0000-0000-000000000000"	14.0.4006.1010 - 14.0.4006.9999
"6333368D-85F0-4EF5-8241-5252B12B2E50"	4.0.116.0 - 4.0.116.0
"1A707EF5-45B2-4235-9327-021E5F9B8BB0"	4.0.6.0 - 4.0.6.0
"25EB5CEE-15BD-4954-BD4E-2624D5878D8C"	14.0.4006.1010 - 14.0.4006.9999

When the protocol is used to communicate with a configuration database, the list of version ids used are the following.

Version Id GUID	Acceptable version range
"00000000-0000-0000-0000-000000000000"	14.0.4006.1010 - 14.0.4006.9999
"F4D348C4-A6E9-4ED5-BDB2-2358B74EF902"	4.0.116.0 - 4.0.116.0
"60B1F2BE-5130-45AB-AF1D-EDD34E626B5D"	4.0.6.0 - 4.0.6.0

The acceptable version numbers specified in the preceding tables can change when the front-end web server is updated. As a result, the updated front-end web server might not be able to communicate with the back-end database server. In order to re-enable front-end web server to back-end database server communication, the front-end web server MUST use an upgrade process to modify any data structure on the back end database server to accommodate any changes which might have occurred to this protocol, and to update the version numbers to match the new acceptable version numbers. Upgrade logic is implementation specific.

The **proc_GetVersion** stored procedure MUST NOT return any result sets.

3.1.5.45 proc_GetWebMetaInfo

The **proc_GetWebMetaInfo** stored procedure is invoked to request metadata information for a site.

```
PROCEDURE proc_GetWebMetaInfo(
      @WebSiteId                      uniqueidentifier,
      @WebDirName                     nvarchar(256),
      @WebLeafName                    nvarchar(128),
      @DGCacheVersion                 bigint,
      @SystemId                       varbinary(512) = NULL,
      @RequestGuid                    uniqueidentifier = NULL OUTPUT
);
```

@WebSiteId: The site collection identifier (section 2.2.1.9) for the site collection containing the site whose metadata is requested.

@WebDirName: The directory name part of the site location.

@WebLeafName: The leaf name part of the site location.

@DGCacheVersion: The version number of the Domain Group Map Cache in the front-end web server. This can be compared with the domain group map cache version number on the back-end database server returned in the Domain Group Cache Versions Result Set (section 2.2.5.4) to determine whether updates are needed. If the *@DGCacheVersion* parameter is set to -2, then the information about the domain group map cache is not requested, and the Domain Group Cache BEDS Update Result Set (section 2.2.5.3) and the Domain Group Cache WFE Update Result Set (section 2.2.5.5) MUST NOT be returned.

@SystemId: The **SystemID** (section 2.2.1.12) of the current user requesting the site metadata information.

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

Return Values: This stored procedure returns an integer return code, which MUST be listed in the following table.

Value	Description
0	Successful execution.
3	A site was not found at the specified location.
1271	The site collection is locked for read and write operations.

This stored procedure MUST return 2, 4, or 5 result sets as follows.

3.1.5.45.1 Site MetaInfo Result Set

The Site MetaInfo Result Set MUST be returned. The Site MetaInfo Result Set MUST contain a single row containing the metainfo for a valid site. If the specified site is not found, zero rows MUST be returned and the stored procedure MUST NOT return any more result sets. The Site MetaInfo Result Set is defined in Site MetaInfo Result Set, section 2.2.5.23.

3.1.5.45.2 Domain Group Cache Versions Result Set

The Domain Group Cache Versions Result Set contains information about the version numbers associated with the domain group map caches on the front-end web server and on the back-end database server for the specified site collection.

The Domain Group Cache Versions Result Set MUST be returned if the specified site is found, and MUST contain one row of version number data. If the value specified by the *@DGCacheVersion* parameter is set to -2, then all columns returned MUST have the value -2, indicating that the value

MUST NOT be used for comparison. The Domain Group Cache Versions Result Set is defined in section Domain Group Cache Versions Result Set, section 2.2.5.4.

3.1.5.45.3 Domain Group Cache BEDS Update Result Set

The Domain Group Cache BEDS Update Result Set contains information to be used in recomputing the domain group map cache.

The Domain Group Cache BEDS Update Result Set returns only if the Domain Group Cache Versions Result Set (section 2.2.5.4) is returned and the *@DGCacheVersion* parameter is not set to -2 and the real domain group version is more recent than the cached version. (The value of **RealVersion** is greater than the value of **CachedVersion** in the Domain Group Cache Versions Result Set).

If the Domain Group Cache BEDS Update Result Set is returned, then it indicates that the database's copy of the domain group map cache is out of date.

When returned, the Domain Group Cache BEDS Update Result Set MUST contain one row for each domain group that is a **member** of a **permission level** in the site collection, ordered by the user identifier (section 2.2.1.13) of the domain groups.

This result set is defined in Domain Group Cache BEDS Update Result Set, section 2.2.5.3.

3.1.5.45.4 Domain Group Cache WFE Update Result Set

The Domain Group Cache WFE Update Result Set contains the binary data needed to refresh the domain group map cache.

The Domain Group Cache WFE Update Result Set returns only if the Domain Group Cache Versions Result Set (section 2.2.5.4) is returned and the *@DGCacheVersion* parameter is not set to -2 and the cached version is up to date (the value of **RealVersion** is not greater than the value of **CachedVersion** in the Domain Group Cache Versions Result Set).

If the Domain Group Cache WFE Update Result Set is returned, one row MUST be returned. The Domain Group Cache WFE Update Result Set is defined in Domain Group Cache WFE Update Result Set, section 2.2.5.5.

3.1.5.45.5 Site Metadata Result Set

The Site Metadata Result Set contains metadata for the specified site.

The Site Metadata Result Set MUST be returned if the specified site is found, and MUST contain a single row corresponding to the specified site. If the site is locked from both read and write operations, the stored procedure MUST NOT return any more result sets.

The Site Metadata Result Set is defined in Site Metadata Result Set, section 2.2.5.22.

3.1.5.45.6 Event Receivers Result Set

The Event Receivers Result Set contains information about the event receivers defined for the specified site. There MUST be one row in the Event Receivers Result Set for each event receiver that is registered for this site. The Event Receivers Result Set MUST be returned on successful execution.

The Event Receivers Result Set is defined in Event Receivers Result Set, section 2.2.5.9.

3.1.5.46 proc_GetWebMetainfoByUrl

The **proc_GetWebMetainfoByUrl** stored procedure is invoked to request site metadata information for the site containing a specified URL.

```
PROCEDURE proc_GetWebMetainfoByUrl(
      @WebSiteId                    uniqueidentifier,
      @Url                          nvarchar(260),
      @DGCacheVersion               bigint,
      @SystemId                     varbinary(512)     = NULL,
      @RequestGuid                  uniqueidentifier = NULL OUTPUT
);
```

@WebSiteId: The site collection identifier (section 2.2.1.9) for the site collection containing the site or document whose metadata is requested.

@Url: A URL in store-relative form for the site or for a document within the site.

@DGCacheVersion: The version number of the Domain Group Map Cache in the front-end web server. This can be compared with the Domain Group Map Cache version number on the back-end database server returned in the Domain Group Cache Versions Result Set (section 2.2.5.4) to determine whether updates are needed. If the *@DGCacheVersion* parameter is set to -2 then information about the Domain Group Map Cache is not requested, and the Domain Group Cache BEDS Update Result Set (section 2.2.5.3) and the Domain Group Cache WFE Update Result Set (section 2.2.5.5) MUST NOT be returned.

@SystemId: The **SystemID** (section 2.2.1.12) of the current user requesting the site metadata information.

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

Return Values: This stored procedure returns an integer return code, which MUST be listed in the following table.

Value	Description
0	Successful execution.
1168	The site collection does not exist.
1271	The site is locked for read and write operations.

This stored procedure MUST return 0, 5, or 6 result sets. Some of the result sets are returned conditionally, and all result sets returned will be sent in the order specified in the following.

3.1.5.46.1 Site URL Result Set

The Site URL Result Set MUST be returned if the values for the *@Url* and *@WebSiteId* parameters are valid. Otherwise, the stored procedure MUST NOT return any result sets. The Site URL Result Set MUST contain a single row, with a URL in store-relative form of the site containing the URL specified in the *@Url* parameter.

```
      {WebUrl}                      nvarchar(260);
```

{WebUrl}: The URL in store-relative form of the site that contains the URL specified in the *@Url* parameter.

3.1.5.46.2 Site Metainfo Result Set

The Site MetaInfo Result Set MUST contain a single row containing the metainfo for the site.

The Site Metadata Result Set is defined in Site Metadata Result Set, section 2.2.5.22.

3.1.5.46.3 Domain Group Cache Versions Result Set

The Domain Group Cache Versions Result Set contains information about the version numbers associated with the domain group map caches on the front-end web Server and on the back-end database server for the specified site collection.

The Domain Group Cache Versions Result Set MUST contain one row of version number data. If the value specified for the *@DGCacheVersion* parameter is set to -2, then all columns returned MUST have the value -2, indicating that the value MUST NOT be used for comparison. The Domain Group Cache Versions Result Set is defined in section Domain Group Cache Versions Result Set, section 2.2.5.4.

3.1.5.46.4 Domain Group Cache BEDS Update Result Set

The Domain Group Cache BEDS Update Result Set contains information to be used in re-computing the domain group map cache, which contains the mapping of domain groups to the site permission levels of which they are members. The presence of the Domain Group Cache BEDS Update Result Set means the database's copy of the domain group map cache is out of date.

The Domain Group Cache BEDS Update Result Set MUST be returned if the *@DGCacheVersion* parameter does not equal -2 and the value of RealVersion is greater than the value of CachedVersion in the results of the Domain Group Cache Versions Result Set (section 2.2.5.4). Otherwise, this result set MUST NOT be returned.

If the Domain Group Cache BEDS Update Result Set is returned, it MUST contain one row for each domain group which is a **member** of a site permission level in the site collection, ordered by the user identifier (section 2.2.1.13) of the domain groups.

This result set is defined in Domain Group Cache BEDS Update Result Set, section 2.2.5.3.

3.1.5.46.5 Domain Group Cache WFE Update Result Set

The Domain Group Cache WFE Update Result Set contains the binary data needed to refresh the domain group map cache. If the Domain Group Cache WFE Update Result Set is returned, then it indicates that the back-end database server Domain Group map cache is up to date, and the front-end web server cache can be refreshed if necessary.

The Domain Group Cache WFE Update Result Set MUST be returned on successful determination of the site if the *@DGCacheVersion* parameter is not set to -2 and the value of RealVersion is less than or equal to the value of CachedVersion in the results of the Domain Group Cache Versions Result Set (section 2.2.5.4). Otherwise, the Domain Group Cache WFE Update Result Set MUST NOT be returned.

If the Domain Group Cache WFE Update Result Set is returned, it MUST contain one row.

The Domain Group Cache WFE Update Result Set is defined in Domain Group Cache WFE Update Result Set, section 2.2.5.5.

3.1.5.46.6 Site Metadata Result Set

The Site Metadata Result Set contains specialized site metadata. If the site is locked for read and write operations, the stored procedure MUST NOT return any more result sets. There MUST be one row in the Site Metadata Result Set.

The Site Metadata Result Set is defined in Site Metadata Result Set, section 2.2.5.22.

3.1.5.46.7 Event Receivers Result Set

The Event Receivers Result Set contains information about the event receivers defined for this site. There MUST be one row in the Event Receivers Result Set for each event receiver registered for this site. The Event Receivers Result Set MUST be returned on successful completion.

The Event Receivers Result Set is defined in Event Receivers Result Set, section 2.2.5.9.

3.1.5.47 proc_ListDocumentVersions

The **proc_ListDocumentVersions** stored procedure is invoked to list all available version history information for a specified document.

```
PROCEDURE proc_ListDocumentVersions(
      @DocSiteId                      uniqueidentifier,
      @DocWebId                       uniqueidentifier,
      @DocDirName                     nvarchar(256),
      @DocLeafName                    nvarchar(128),
      @MaxLevel                       tinyint,
      @UserId                         int,
      @RequestGuid                    uniqueidentifier = NULL OUTPUT
);
```

@DocSiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the document.

@DocWebId: The site identifier (section 2.2.1.11) of the site containing the document.

@DocDirName: The directory name containing the document.

@DocLeafName: The leaf name containing the document.

@MaxLevel: A **Publishing Level** type (section 2.2.2.6) indicating the maximum publishing level value of the document that SHOULD be returned to the front-end web server in the Document Versions Result Set (section 3.1.5.47.3) if multiple publishing levels of the document are available. See the *@UserId* parameter for an exception to this.

@UserId: This parameter is the user identifier (section 2.2.1.13) of the current user. If the current user is the owner of one or more versions of the document at any publishing level, the version information MUST be returned in the Document Versions Result Set, ignoring the value specified in the *@MaxLevel* parameter.

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

Return Values: This stored procedure returns an integer return code, which MUST be included in the following table.

Value	Description
0	Successful execution.
2	The document specified is not a file (that is, it does not have a **Document Store** type (section 2.2.2.4) of 0).

This stored procedure MUST return two or three result sets upon successful completion.

3.1.5.47.1 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 within.

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 (section 3.1.5.47.2)

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 Individual URL Security Result Set (section 2.2.5.10).

3.1.5.47.2 NULL Individual URL Security Result Set

The NULL Individual URL Security Result Set indicates that the specified document URL is not contained within a list or document library. The NULL Individual URL Security Result Set MUST only be returned if the document URL is not contained within a list or document library.

The NULL Individual URL Security Result Set MUST return a single row and is defined in NULL Individual URL Security Result Set (section 2.2.5.14).

3.1.5.47.3 Document Versions Result Set

The Document Versions Result Set returns version information for a specified document. The Document Versions Result Set MUST only be returned when the specified document has a **Document Store** type (section 2.2.2.4) of 0. The Document Versions Result Set MUST contain document version information corresponding to publishing levels less than or equal to the specified publishing level, or corresponding to any publishing levels owned by the current user.

```
TimeCreated                     datetime,
Version                         int,
Size                            int,
CheckinComment                  nvarchar(1023),
MetaInfo                        varbinary(max),
{IsTip}                         bit,
Level                           tinyint,
DraftOwnerId                    int;
```

TimeCreated: A timestamp in **UTC** format specifying when this document was created.

Version: A counter incremented any time a change is made to this document, used for internal conflict detection.

Size: The number of bytes in the document stream.

CheckinComment: An optional user-provided description used when a document is being checked in or published. This value MUST be NULL for documents that have not been checked in or published.

MetaInfo: A METADICT for the document. The METADICT format is specified in [MS-FPSE], section 2.2.2.2.11.

{IsTip}: If set to 1, the document is a current version, otherwise it is a historical version.

Level: A **Publishing Level** type (section 2.2.2.6) value specifying the publishing level of the document.

DraftOwnerId: The user that published this document as a draft. This value is only non-NULL if the document is a draft version. See **Publishing Level** type (section 2.2.2.6) for more information regarding the different publishing levels.

3.1.5.48 proc_ListRbsStores

The **proc_ListRbsStores** stored procedure is invoked to enumerate the remote blob storage stores with which the back-end database server has been configured, if any.

```
PROCEDURE proc_ListRbsStores (
@RequestGuid uniqueidentifier = null OUTPUT,
);
```

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

Return values:

Value	Description
0	Default return value

Result Sets:

This procedure MUST return the List RBS Stores Result Set (section 3.1.5.48.1).

3.1.5.48.1 List RBS Stores Result Set

```
blob_store_name nvarchar(128)                    NOT NULL,
```

blob_store_name: A remote blob storage store name with which the BEDS has been configured.

3.1.5.49 proc_ListUrls

The **proc_ListUrls** stored procedure retrieves metadata for a document specified by a URL and the documents contained within it, if any.

```
PROCEDURE proc_ListUrls(
      @DirSiteId                uniqueidentifier,
      @DirWebId                 uniqueidentifier,
      @DirFullUrl               nvarchar(260),
      @AttachmentsFlag          tinyint,
      @ClientTimeStamp          datetime,
      @FetchLinkInfo            bit,
      @IncludeThicketDirs       bit,
      @IncludeListItems         bit,
      @UserId                   int,
      @ItemLimit                int,
      @RequestGuid              uniqueidentifier = NULL OUTPUT
);
```

@DirSiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the document specified by a URL.

@DirWebId: The site identifier (section 2.2.1.11) of the site containing the document.

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

@AttachmentsFlag: An **Attachments Flag** (section 2.2.3.1) value specifying whether the document is, or is contained within, a folder for attachments.

@ClientTimeStamp: A datetime that specifies a limiting time on the data in the result sets returned. See the description of the result sets below for the specific effects of this parameter value.

@FetchLinkInfo: A bit flag specifying whether to return the Link Info Result Set (section 3.1.5.49.6). If this parameter is set to 1 and the specified document is a folder, the Link Info Result Set MUST be returned.

@IncludeThicketDirs: A bit flag specifying whether to return **thicket folders** in the Contained Document Metadata Result Set (section 3.1.5.49.7). If this parameter is set to 1 and the specified document is a folder, any thicket folders MUST be included in the Contained Document Metadata Result Set.

@IncludeListItems: A bit flag specifying whether to include list items which are files in the Link Info Result Set and the Contained Document Metadata Result Set. If this parameter is set to 1 and the document is a folder, list items with a **Document Store** type (section 2.2.2.4) of 0 MUST be included in the Link Info Result Set and the Contained Document Metadata Result Set.

@UserId: The user identifier (section 2.2.1.13) for the current user.

@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 *@DirFullUrl* 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 the below return values for more information.

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

Return Values: This stored procedure returns an integer return code that 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* parameter.

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

3.1.5.49.1 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 (section 3.1.5.49.2) 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 Individual URL Security Result Set, section 2.2.5.10.

3.1.5.49.2 NULL Individual URL Security Result Set

The NULL Individual URL Security Result Set indicates that the specified document URL is not contained within a list or document library. The NULL Individual URL Security Result Set MUST only be returned if the document URL is not contained within a list or document library.

The NULL Individual URL Security Result Set MUST return a single row and is defined in NULL Individual URL Security Result Set, section 2.2.5.14.

3.1.5.49.3 Server Time Result Set

The Server Time Result Set returns the current time from the back-end database server in **UTC**. The Server Time Result Set MUST be returned and MUST contain a single row of data.

The Server Time Result Set is defined in Server Time Result Set, section 2.2.5.18.

3.1.5.49.4 Subsite List Result Set

The Subsite List Result Set contains an unordered list of URLs in store-relative form for all subsites whose **parent site** is the site specified by the *@DirWebId* parameter.

If the specified document is not a folder, the Subsite List Result Set MUST NOT be returned. Otherwise the Subsite List Result Set MUST be returned and MUST contain one row for each subsite within the specified site, and it MUST contain no rows if there are no such subsites.

The Subsite List Result Set is defined in URL Result Set, section 2.2.5.25.

3.1.5.49.5 Document Metadata Result Set

The Document Metadata Result Set contains the metadata for the specified document. If the document is not a folder, or if the *@DirFullUrl* parameter contains an empty string, the Document Metadata Result Set MUST NOT be returned. Otherwise, the Document Metadata Result Set MUST contain one row with the metadata information for the document.

```
          Id                            uniqueidentifier,
          {FullUrl}                     nvarchar(260),
          Type                          tinyint,
          MetaInfoTimeLastModified      datetime,
          MetaInfo                      varbinary(max),
          Size                          int,
          TimeCreated                   datetime,
          TimeLastModified              datetime,
          ETagVersion                   int,
          DocFlags                      int,
          {ListType}                    int,
          tp_Name                       nvarchar(38),
          {ListTitle}                   nvarchar(255),
          CacheParseId                  uniqueidentifier,
          {GhostDirName}                nvarchar(256),
          {GhostLeafName}               nvarchar(128),
          {tp_Login}                    nvarchar(255),
          {CheckoutDate}                datetime,
          {CheckoutExpires}             datetime,
          VirusStatus                   int,
          VirusInfo                     nvarchar(255),
          SetupPathVersion              tinyint,
          SetupPath                     nvarchar(255),
          SetupPathUser                 nvarchar(255),
          NextToLastTimeModified        datetime,
          UIVersion                     int,
          CheckinComment                nvarchar(1023),
          WelcomePageUrl                nvarchar(260),
          WelcomePageParameters         nvarchar(max),
          tp_Flags                      bigint,
          Acl                           varbinary(max),
          AnonymousPermMask             bigint,
          DraftOwnerId                  int,
          Level                         tinyint,
          ParentVersion                 int,
          TransformerId                 uniqueidentifier,
          ParentLeafName                nvarchar(128),
          ProgId                        nvarchar(255),
          DoclibRowId                   int,
          tp_DefaultWorkflowId          uniqueidentifier,
          ListId                          uniqueidentifier,
          ItemChildCount                  int,
          FolderChildCount                int,
          MetaInfoVersion                 int,
          {CurVerMetaInfo}                varbinary(max),
          ContentVersion                  int,
          IsDirty                         bit;
```

Id: The **document identifier** (section 2.2.1.2) of the document.

{FullUrl}: The URL in store-relative form for the document.

Type: The **Document Store** type (section 2.2.2.4) of the document.

MetaInfoTimeLastModified: A datetime with a timestamp in **UTC** format specifying the last time the **Metainfo** column value of the document was changed. This value MUST be NULL if the **Metainfo** column value of the document has never been changed.

MetaInfo: A METADICT for the document. The METADICT format is specified in [MS-FPSE] section 2.2.2.2.11. This value MUST be NULL if the document does not have any METADICT associated with it.

Size: The number of bytes in the document stream of the document. This value MUST be NULL if the document does not have a document stream.

TimeCreated: A datetime with a timestamp in UTC format specifying when the document was created.

TimeLastModified: A datetime with a timestamp in UTC format specifying when the document was last modified.

ETagVersion: A counter incremented any time a change is made to the document, and used for internal conflict detection.

DocFlags: A **Doc Flags** (section 2.2.2.3) value describing the document. This value MUST be NULL if the document does not have any **Doc Flags** associated with it.

{ListType}: A packed combination of the **List Base** type (section 2.2.3.11) and **List Server Template** (section 2.2.3.12) values of the list containing this document, consisting of the **List Server Template** value multiplied by 256 and added to the value of the **List Base** type.

tp_Name: The list identifier (section 2.2.1.5) of the list containing this document.

{ListTitle}: If the document URL is the root folder of a list, this contains the display name of the list. Otherwise, this value MUST be NULL.

CacheParseId: This value MUST be NULL.

{GhostDirName}: This value MUST be NULL.

{GhostLeafName}: This value MUST be NULL.

{tp_Login}: This value MUST be NULL.

{CheckoutDate}: This value MUST be NULL.

{CheckoutExpires}: This value MUST be NULL.

VirusStatus: A **Virus Status** (section 2.2.3.18) value specifying the current virus state of the document. This value can be NULL if the document has not been processed by a virus scanner.

VirusInfo: A string containing a provider-specific message returned by the virus scanner when it last processed the document. This value can be NULL if the document has not been processed by a virus scanner.

SetupPathVersion: If this is a ghosted document, then this specifies the setup path location that the **SetupPath** fragment is relative to. This value MUST NOT be NULL. The following are valid values.

Value	Description
2	The **SetupPath** is relative to the install location of WSS2 on the front-end web server (for example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\60).
3	The **SetupPath** is relative to the install location of WSS3 on the front-end web server (for example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\12).
4	The **SetupPath** is relative to the install location of WSS4 on the front-end web server (for example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\14).

SetupPath: If the document is now or once was ghosted, then this contains the setup path fragment relative to the base setup path described above by the **SetupPathVersion** value, where the content **stream** of the document can be found. This value can be NULL.

SetupPathUser: If the document is now or once was ghosted, then this contains the login name of the user who created the ghosted document. This value can be NULL.

NextToLastTimeModified: The value of **TimeLastModified** from the previous time when the document was last saved. If the **NextToLastTimeModified** value matches the **TimeLastModified** value when the change occurred and the client has a version of the document that it has successfully modified, then the client can safely submit the document to the server despite what appears to be an intervening edit to the document. This value MUST be NULL if the document has never been saved.

UIVersion: The user interface version number for the document.

CheckinComment: An optional user-supplied description provided when the document is checked in or published. This value can be NULL.

WelcomePageUrl: If the document is a folder, this specifies an optional page to redirect to when the folder is requested with an HTTP GET operation. The URL is relative to the URL of the folder itself, and MUST be contained within that folder. Attempts to break out of the folder, such as "../../somepage.aspx", are not valid. This value can be NULL.

WelcomePageParameters: Contains optional URL parameters to specify as part of the **WelcomePageUrl** value. These parameters start at either the query string signifier '?' or the bookmark signifier '#'. This value can be NULL.

tp_Flags: The **List Flags** (section 2.2.2.5) value for the list that contains the document.

Acl: The binary serialization of the WSS ACL for the document. The WSS ACL is either explicitly defined for the document, or inherited from the parent object of the document. This value can be NULL.

AnonymousPermMask: A **WSS Rights Mask** (section 2.2.2.14) 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 (section 2.2.1.13) 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 (section 2.2.2.6) value specifying the publishing status of the document.

ParentVersion: If the document is a transformed version of another document, then this is the user interface version value from the original document. This value MUST be NULL if the document is not the product of a document transformation.

TransformerId: If the document is a transformed version of another document, then this is the GUID of the agent that performed the transformation. This value MUST be NULL if the document is not the product of a document transformation.

ParentLeafName: If the document is a transformed version of another document, then this is the leaf name of the original document. The original document MUST be in the same folder as the transformed document. If either is moved, then the relationship is broken. This value MUST be NULL if the document is not the product of a document transformation.

ProgId: Specifies a preferred application to open the document. The **ProgID** value is used to distinguish between different applications that save files with a given **file extension** (for example, different editing applications for .HTML or .XML files). This value MUST be NULL if a **ProgID** was not specified when the document was saved.

DoclibRowId: The document library row identifier for the document. If the document is not contained in a list, then this value MUST be NULL.

tp_DefaultWorkflowId: The workflow identifier (section 2.2.1.16) corresponding to the **workflow** to be invoked if the document is in a moderated list and the document is submitted for approval as part of a check in.

ListId: The list identifier (section 2.2.1.5) of the list that contains the document. If the document is not contained in a list, then this value MUST be NULL.

ItemChildCount: MUST be the number of list items for the list that contains the document.

FolderChildCount: MUST be the number of folders that exist in the list that contains the document.

MetaInfoVersion: A counter incremented any time a change is made to the metainfo for this document and used for internal conflict detection.

{CurVerMetaInfo}: The value MUST be NULL.

ContentVersion: The version number of the document stream being returned.

IsDirty: A bit that specifies this document MUST have implementation-specific processing performed before its stream can be returned. If this document does not require processing, this value MUST be 0.

3.1.5.49.6 Link Info Result Set

The Link Info Result Set returns information about all forward links from and backward links to the documents contained within the specified document. If the specified document is not a folder, or the *@FetchLinkInfo* parameter is not set to 1, then the Link Info Result Set MUST NOT be returned. Otherwise, if the document is a folder and the *@FetchLinkInfo* parameter is set to 1, then the Link Info Result Set MUST be returned. The Link Info Result Set MUST contain one row for each link that has been modified after the value in the *@ClientTimeStamp* parameter for each contained document in the site collection which has its directory name equal to the value of the *@DirFullUrl* parameter, and the contained document has a **Document Store** type (section 2.2.2.4) of 0 and the *@IncludeListItems* parameter is set to 1, the document is in a document library, or the document can have a document stream, and for forward links, is a published or draft version which is not checked out to the specified user, or is a checked out version which is checked out to the user.

The Link Info Result Set is defined using T-SQL syntax as described in Single Doc Link Information Result Set, section 2.2.5.19.

3.1.5.49.7 Contained Document Metadata Result Set

The Contained Document Metadata Result Set contains the metadata information for the documents contained within the specified document. If the specified document is not a folder, the Contained Document Metadata Result Set MUST NOT be returned. Otherwise, the Contained Document Metadata Result Set MUST return one row for each document in the site collection whose directory name is equal to the value of the *@DirFullUrl* parameter, and is a published or draft version which is not checked out to the specified user, or is a checked out version which is checked out to the user, and either the contained document has a **Document Store** type (section 2.2.2.4) of 0 and the *@IncludeListItems* parameter is set to 1, the document is in a document library, or the document can have a document stream, or the contained document is a folder, and the folder is not a thicket supporting file, and either the folder is not a thicket folder or the *@IncludeThicketDirs* parameter is set to 1.

```
          Id                              uniqueidentifier,
          {FullUrl}                       nvarchar(260),
          Type                            tinyint,
          MetaInfoTimeLastModified        datetime,
          MetaInfo                        varbinary(max),
          Size                            int,
          TimeCreated                     datetime,
          TimeLastModified                datetime,
          ETagVersion                     int,
          DocFlags                        int,
          {ListType}                      int,
          tp_Name                         nvarchar(38),
          {ListTitle}                     nvarchar(255),
          CacheParseId                    uniqueidentifier,
          {GhostDirName}                  nvarchar(256),
          {GhostLeafName}                 nvarchar(128),
          tp_Login                        nvarchar(255),
          CheckoutDate                    datetime,
          CheckoutExpires                 datetime,
          VirusStatus                     int,
          VirusInfo                       nvarchar(255),
          SetupPathVersion                tinyint,
          SetupPath                       nvarchar(255),
          SetupPathUser                   nvarchar(255),
          NextToLastTimeModified          datetime,
          UIVersion                       int,
          CheckinComment                  nvarchar(1023),
          WelcomePageUrl                  nvarchar(260),
          WelcomePageParameters           nvarchar(max),
          tp_Flags                        bigint,
          Acl                             varbinary(max),
          AnonymousPermMask               bigint,
          DraftOwnerId                    int,
          Level                           tinyint,
          ParentVersion                   int,
          TransformerId                   uniqueidentifier,
          ParentLeafName                  nvarchar(128),
          ProgId                          nvarchar(255),
          DoclibRowId                     int,
          tp_DefaultWorkflowId            uniqueidentifier,
          ListId                          uniqueidentifier,
          ItemChildCount                  int,
          FolderChildCount                int,
          MetaInfoVersion                 int,
          {CurVerMetaInfo}                varbinary(max),
          ContentVersion                  int,
          IsDirty                         bit;
```

Id: The **document identifier** (section 2.2.1.2) of this contained document.

{FullUrl}: The URL in store-relative form for this document.

Type: The **Document Store** type (section 2.2.2.4) of the document.

MetaInfoTimeLastModified: A **datetime** with a timestamp in **UTC** format specifying the last time the **Metainfo** column value of the document was changed. This value can be NULL if the **Metainfo** column value of the document has never been changed.

MetaInfo: A METADICT for the document. The METADICT format is specified in [MS-FPSE] section 2.2.2.2.11. This value can be NULL, and MUST be NULL if the **MetaInfoTimeLastModified** value is not more recent than the *@ClientTimeStamp* parameter.

Size: The number of bytes in the document stream of the document. This value MUST be NULL if the document does not have a document stream.

TimeCreated: A **datetime** with a timestamp in UTC format specifying when the document was created.

TimeLastModified: A **datetime** with a timestamp in UTC format specifying when the document was last modified.

ETagVersion: A counter incremented any time a change is made to the document, and used for internal conflict detection.

DocFlags: A **Doc Flags** (section 2.2.2.3) value describing the document. This value can be NULL if the document does not have any **Doc Flags** associated with it.

{ListType}: A packed combination of the **List Base** type (section 2.2.3.11) and **List Server Template** (section 2.2.3.12) values of the list containing this document, consisting of the **List Server Template** value multiplied by 256 and added to the value of the **List Base** type.

tp_Name: The list identifier (section 2.2.1.5) of the list containing this document.

{ListTitle}: If the containing document is the root folder of the list that contains this document, then this contains the display name of the list. Otherwise, this value MUST be NULL.

CacheParseId: This value MUST be NULL.

{GhostDirName}: This value MUST be NULL.

{GhostLeafName}: This value MUST be NULL.

tp_Login: If this document is currently checked out, then this is the login name of the user to whom it is checked out. In all other cases, this MUST be NULL.

CheckoutDate: A datetime with a timestamp in UTC format specifying when this document was checked out. This MUST be NULL if the document has never been checked out.

CheckoutExpires: A datetime with a timestamp in UTC format specifying when the **short-term lock** for this document will expire. If this date is in the past, then this document SHOULD be treated as unlocked. This value can be NULL if no short-term lock has been placed on the document.

VirusStatus: A **Virus Status** (section 2.2.3.18) value specifying the current virus state of the document. This value can be NULL if the document has not been processed by a virus scanner.

VirusInfo: A string containing a provider-specific message returned by the virus scanner when it last processed the document. This value can be NULL if the document has not been processed by a virus scanner.

SetupPathVersion: If this is a ghosted document, then this specifies the setup path location that the **SetupPath** fragment is relative to. This value MUST NOT be NULL. The following are valid values.

Value	Description
2	The **SetupPath** is relative to the install location of WSS2 on the front-end web server (for

Value	Description
	example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\60).
3	The **SetupPath** is relative to the install location of WSS3 on the front-end web server (for example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\12).
4	The **SetupPath** is relative to the install location of WSS4 on the front-end web server (for example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\14).

SetupPath: If the document is now or once was ghosted, then this contains the setup path fragment relative to the base setup path described above by the **SetupPathVersion** value, where the content **stream** of the document can be found. This value can be NULL.

SetupPathUser: If the document is now or once was ghosted, then this contains the login name of the user who created the ghosted document. This value can be NULL.

NextToLastTimeModified: The value of **TimeLastModified** from the previous time when the document was last saved. If the **NextToLastTimeModified** value matches the **TimeLastModified** value when the change occurred and the client has a version of the document that it has successfully modified, then the client can safely submit the document to the front-end web server despite what appears to be an intervening edit to the document. This value MUST be NULL if the document has never been saved.

UIVersion: The user interface version number for the document.

CheckinComment: An optional user-supplied description provided when the document is checked in or published. This value can be NULL.

WelcomePageUrl: If the document is a folder, then this specifies an optional page to redirect to when the folder is requested with an HTTP GET operation. The URL is relative to the URL of the folder itself, and MUST be contained within that folder. Attempts to break out of the folder, such as "../../somepage.aspx", are not valid. This value can be NULL.

WelcomePageParameters: Contains optional URL parameters to specify as part of the **WelcomePageUrl** value. These parameters start at either the query string signifier '?' or the bookmark signifier '#'. This value can be NULL.

tp_Flags: The **List Flags** (section 2.2.2.5) value for the list that contains the document.

Acl: The binary serialization of the WSS ACL for the document. The WSS ACL is either explicitly defined for the document, or inherited from the parent object of the document. This value can be NULL if a WSS ACL is not defined for the document.

AnonymousPermMask: A **WSS Rights Mask** (section 2.2.2.14) that indicates the permissions granted to 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 (section 2.2.1.13) 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 (section 2.2.2.6) value specifying the publishing status of the document.

ParentVersion: If the document is a transformed version of another document, then this is the user interface version value from the original document. This value MUST be NULL if the document is not the product of a document transformation.

TransformerId: If the document is a transformed version of another document, this is the GUID of the agent that performed the transformation. This value MUST be NULL if the document is not the product of a document transformation.

ParentLeafName: If the document is a transformed version of another document, this is the leaf name of the original document. The original document MUST be in the same folder as the transformed document. If either is moved, the relationship is broken. This value MUST be NULL if the document is not the product of a document transformation.

ProgId: Specifies a preferred application to open the document. The **ProgID** value is used to distinguish between different applications that save files with a given file extension (for example, different editing applications for .HTML or .XML files). This value MUST be NULL if a **ProgId** was not specified when the document was saved.

DoclibRowId: The document library row identifier for the document. If the document is not contained in a list, this value MUST be NULL.

tp_DefaultWorkflowId: The workflow identifier (section 2.2.1.16) corresponding to the **workflow** to be invoked if the document is in a moderated list and the document is submitted for approval as part of a check in.

ListId: The list identifier of the list that contains the document. If the document is not contained in a list, this value MUST be NULL.

ItemChildCount: The number of list items for the list that contains the document if the document is contained in a list.

FolderChildCount: The number of folders that exist in the list that contains the document if the document is contained in a list.

MetaInfoVersion: A counter incremented any time a change is made to the metainfo for this document and used for internal conflict detection.

{CurVerMetaInfo}: The value MUST be NULL.

ContentVersion: The version number of the document stream being returned.

IsDirty: A bit that specifies this document MUST have implementation-specific processing performed before its stream can be returned. If this document does not require processing, this value MUST be 0.

3.1.5.50 proc_RenameUrl

The **proc_RenameUrl** stored procedure renames a document or folder within a specified site.

```
PROCEDURE proc_RenameUrl(
      @SiteId                       uniqueidentifier,
      @SubWebId                     uniqueidentifier,
      @OldUrl                       nvarchar(260),
      @NewUrl                       nvarchar(260),
      @UserId                       int,
      @ThresholdRowCount            int,
      @NewDoclibRowIdInput          int,
      @MaxNewRowsInput              int,
      @RenameFlags                  int                 = 0,
      @PutFlags                     int                 = 0,
      @ReturnFlags                  int                 = 0,
      @AttachmentOpOldUrl           int                 = 3,
      @AttachmentOpNewUrl           int                 = 3,
      @OldItemId                    int                 = NULL OUTPUT,
      @ParseDocsNow                 tinyint             = NULL OUTPUT,
      @FailedUrl                    nvarchar(260)       = NULL OUTPUT,
      @RequestGuid                  uniqueidentifier    = NULL OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the document or folder.

@SubWebId: The site identifier (section 2.2.1.11) of the site containing the document or folder.

@OldUrl: The store-relative URL of the document or folder to be moved.

@NewUrl: The target store-relative URL of the document or folder after the move.

@UserId: The user identifier (section 2.2.1.13) of the user who is requesting the move.

@ThresholdRowCount: If the *@OldUrl* parameter is a folder and this parameter is greater than 0 and the sum total of documents and folders contained within the *@OldUrl* parameter exceeds this parameter's value, then this stored procedure MUST NOT complete successfully.

@NewDoclibRowIdInput: The next available row number in the new list where this item is being copied.

@MaxNewRowsInput: The number of items that are being renamed as part of this rename operation.

@RenameFlags: A bit field determining document rename options. This can have one or more flags set. The only valid values of the *@RenameFlags* bits are specified in **Rename Flags** (section 2.2.2.8).

@PutFlags: A bit field determining document change options. This can have one or more flags set. The only valid values of the *@PutFlags* bits are specified in **Put Flags** type (section 2.2.2.7).

@ReturnFlags: A bit field determining the type of information requested. This can have one or more flags set. The only valid bit values of *@ReturnFlags* are specified as follows.

Value	Description
0x01	Return information about the moved documents.
0x02	Return information about backward links referencing the moved documents.

@AttachmentOpOldUrl: An **Attachments Flag** (section 2.2.3.1) value for the document URL.

@AttachmentOpNewUrl: An **Attachments Flag** value for the destination URL.

@OldItemId: If this parameter is supplied, it will be set to the row number that was being used by the previous document (the URL specified by the *@OldUrl* parameter). If the *@OldUrl* parameter is a folder, then this parameter MUST be ignored.

@ParseDocsNow: An output parameter set to 1 to indicate that the moved document or documents in the folder MUST be parsed again. Reparsing is necessary when the document properties need to be scanned, for example, when the document is moved across lists, or when the document's file extension is modified. If this output parameter is supplied, then it will be set to 0 indicating that the document or documents in the folder MUST NOT need to be parsed again.

@FailedUrl: An output parameter indicating the URL at which the move operation failed. MUST be set to NULL if the move was successful.

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

Return Values: The **proc_RenameUrl** stored procedure returns an integer return code which MUST be listed in the following table.

Value	Description
0	Successful execution.
2	File not found: Parent of target URL not found.
3	Path not found: Item not found at specified URL, site collection and site.
5	Access denied: The user specified in *@UserId* lacks the necessary privileges.
15 or 51	Cannot move to, or from, a forms folder in a list or document library.
33	Cannot move a folder which contains checked out documents.
34	Cannot rename a folder within a list or move a folder out of a list when the list is not a document library.
36	The rename operation failed to complete successfully because the value specified for the *@ThresholdRowCount* parameter exceeds the sum total of documents and folders contained within the URL specified by the *@OldUrl* parameter.
50	The document is a site, but *@RenameFlags* does not indicate that a site move is requested.
80	Target URL is a document with a **Document Store** type (section 2.2.2.4) of 0 (File), but *@PutFlags* value does not indicate a request to overwrite it.
87	Bad parameter: Unspecified error prevented the requested move.
130	Cannot move to, or from, an image or a thumbnail file in an image library.
138	Cannot move folder across lists.
144	Not an overwrite request and URL is a directory or site, or other invalid combination.
161	Cannot move folder across sites.
190	Cannot move folder containing thicket.
206	Target URL is too long.
214	Cannot move thicket.
1150	Concurrency violation.
1358	Internal database corruption.
1359	Unknown internal error.
8398	There was an error moving a list. At least one list could not be deleted.

This stored procedure MUST return either zero, one or two result sets, depending on the values for input parameters as described for each result set.

3.1.5.50.1 Rename Result Set

The Rename Result Set returns basic information about the old and new URLs for all moved (renamed) Documents. When renaming a container object, affected items in the container are included in the Rename Result Set. The Rename Result Set MUST be returned only when requested (when bit 0x01 is set in *@ReturnFlags*). If the Rename Result Set is returned, it MUST return one row for each document which was renamed during the operation.

```
        {OldUrlDirName}                nvarchar(256),
        {OldUrlLeafName}              nvarchar(128),
        {NewUrlDirName}               nvarchar(256),
        {NewUrlLeafName}              nvarchar(128),
        {Type}                        int;
```

{OldUrlDirName}: The directory name of the document before rename.

{OldUrlLeafName}: The leaf name of the document before rename.

{NewUrlDirName}: The directory name of the document after rename.

{NewUrlLeafName}: The leaf name of the document after rename.

{Type}: The **Document Store** type (section 2.2.2.4) of the renamed document.

3.1.5.50.2 Backward Link Result Set

The Backward Link Result Set returns the URL of each document containing a backward link to the moved (renamed) document(s). The Backward Link Result Set MUST be returned only when requested (bit 0x2 is set in *@ReturnFlags* and bit 0x4 is set in *@RenameFlags*). If the Backward Link Result Set is returned, it MUST return one row for each item containing a backward link to any of the renamed document(s).

```
DocUrl                         nvarchar(260),
```

DocUrl: The store-relative form URL to the document holding the backward link to the renamed document.

3.1.5.51 proc_SecAddPrincipalToRole

The **proc_SecAddPrincipalToRole** stored procedure is invoked to add a **security principal** to a role defined within a site collection.

```
PROCEDURE proc_SecAddPrincipalToRole(
      @SiteId                       uniqueidentifier,
      @WebId                        uniqueidentifier,
      @ScopeId                      uniqueidentifier,
      @RoleId                       int,
      @UserId                       int,
      @AddChangeLog                 bit = 0,
      @ReturnAuditMask              bit = 1,
      @AddToCurrentScopeOnly        bit = 0,
      @RequestGuid                  uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the role and security principal.

@WebId: The site identifier (section 2.2.1.11) of the site containing the role and security principal.

@ScopeId: The scope identifier (section 2.2.1.8) of the security scope containing the role.

@RoleId: Specifies the role identifier of the **role definition** that the security principal is added into. *@RoleId* MUST correspond to a valid role.

@UserId: The user identifier (section 2.2.1.13) for the security principal to be added to the specified role. This value MUST refer to an existing user identifier for the specified site collection.

@AddChangeLog: A bit flag specifying whether to update the **change log**. A value of 1 indicates that the change log MUST be updated The default value of 0 indicates that the change log MUST NOT updated.

@ReturnAuditMask: A bit flag specifying whether to return **Audit Flags** (section 2.2.2.1) data for the specified site. The default value of 1 indicates that **Audit Flags** data MUST be returned. A value of 0 indicates that **Audit Flags** data MUST NOT be returned.

@AddToCurrentScopeOnly: A bit flag specifying whether the principal will be added only to the given security scope. The default value of 0 indicates that security principal MUST be added to all security scopes up to the security scope of web indicated by *@WebId*. A value of 1 indicates that security principal MUST be added only to the security scope indicated by *@ScopeId*.

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

Return Values: The **proc_SecAddPrincipalToRole** stored procedure returns an integer return code which MUST be in the following table.

Value	Description
0	Successful execution.
3	The site specified by *@WebId* is not valid for membership permissions or there are no roles defined for this site.

The **proc_SecAddPrincipalToRole** stored procedure MUST return one result set if the *@ReturnAuditMask* value is set to 1, and MUST NOT return a result set if the value is set to 0.

3.1.5.51.1 Site Audit Mask Result Set

The Site Audit Mask Result Set contains information about the **Audit Flags** (section 2.2.2.1) set for the site. The Site Audit Mask Result Set MUST be returned if the *@ReturnAuditMask* value is set to 1. The Site Audit Mask Result Set is defined in Site Audit Mask Result Set, section 2.2.5.20.

3.1.5.52 proc_SecAddRoleDef

The **proc_SecAddRoleDef** stored procedure creates a new role definition for a specified site within a site collection.

```
PROCEDURE proc_SecAddRoleDef(
      @SiteId                          uniqueidentifier,
      @WebId                           uniqueidentifier,
      @Title                           nvarchar(255),
      @Description                     nvarchar(512),
      @Hidden                          bit,
      @RoleOrder                       int,
      @Type                            tinyint,
      @PermMask                        tPermMask,
      @IdToCreate                      int,
      @RoleDefId                       int = NULL OUTPUT,
      @RequestGuid                     uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection to add the role definition to.

@WebId: The site identifier (section 2.2.1.11) of the site to add the role definition to. The site MUST have its own scope specifying unique permissions.

@Title: The title of the role definition for display in the front-end web server. This MUST be a value which does not match an existing role definition title within the site, and MUST NOT be NULL or an empty string.

@Description: The description of the role definition for display in the front-end web server.

@Hidden: A bit flag specifying whether or not the role definition is to be displayed by the front-end web server. This value MUST NOT be NULL. If this parameter is set to 1, then the front-end web server MUST NOT display the role definition.

@RoleOrder: An integer value specifying the relative position in which this role definition is to be displayed in the front-end web server. Multiple role definitions can have the same *@RoleOrder* value. This value MUST NOT be NULL.

@Type: The **Role Definition** type (section 2.2.3.16) value for the role definition to be added.

@PermMask: A **WSS Rights Mask** (section 2.2.2.14) specifying the rights to grant to the role definition. The **WSS Rights Mask** associated with the role definition MUST be set to this value.

@IdToCreate: The role identifier to assign to the new role definition. If this parameter is NULL, a new value MUST be assigned and returned in *@RoleDefId*. This parameter MUST NOT be a role identifier which already exists within the site collection.

@RoleDefId: The role identifier of the created role definition, returned as an output parameter.

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

Return Values: The **proc_SecAddRoleDef** stored procedure returns an integer return code which MUST be listed in the following table.

Value	Description
0	Successful execution.
80	The *@IdToCreate* or *@Title* already exists or is NULL for a role definition within the site collection.
1816	The limit for the maximum number of role definitions in the specified site has been reached.

Upon successful execution, the **proc_SecAddRoleDef** stored procedure MUST return a single result set.

3.1.5.52.1 Site Audit Mask Result Set

The Site Audit Mask Result Set contains information about the **Audit Flags** (section 2.2.2.1) associated with the site. The Site Audit Mask Result Set MUST be returned on successful completion and MUST contain one row. The Site Audit Mask Result Set is defined in Site Audit Mask Result Set, section 2.2.5.20.

3.1.5.53 proc_SecAddUser

The **proc_SecAddUser** stored procedure is invoked to add a new entry for a **security principal** to the **UserInfo** table (section 2.2.7.10) that contains descriptive properties and security information about security principals, and is stored in the back-end database server.

```
PROCEDURE proc_SecAddUser(
      @SiteId                        uniqueidentifier,
      @SystemId                      varbinary(512),
      @ExternalToken                 varbinary(max),
      @ExternalTokenTime             datetime,
      @IsDomainGroup                 bit,
      @IsActive                      bit,
      @Login                         nvarchar(255),
      @Title                         nvarchar(255),
      @Email                         nvarchar(255),
      @Notes                         nvarchar(1023),
      @MobilePhone                   nvarchar(127),
      @Flags                         int,
      @MembershipWebId               uniqueidentifier = NULL,
      @IncrementUserCount            bit = 0,
      @ImportDeleted                 bit = 0,
      @AddedToTable                  bit OUTPUT,
      @UserIdOut                     int OUTPUT,
      @LoginOut                      nvarchar(255) OUTPUT,
      @TitleOut                      nvarchar(255) OUTPUT,
      @EmailOut                      nvarchar(255) OUTPUT,
      @NotesOut                      nvarchar(1023) OUTPUT,
      @MobilePhoneOut                nvarchar(127)  OUTPUT,
      @DeletedOut                    bit OUTPUT,
      @NeedtoAddToList               bit OUTPUT,
      @RequestGuid                   uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection to associate with the **principal**.

@SystemId: The **SystemID** (section 2.2.1.12) of the principal to be added or updated. If a user exists with the specified **SystemID**, its record will be updated. Otherwise, a new user will be added with the specified **SystemID**.

@ExternalToken: An **External Group Token** (section 2.2.4.2) specifying information on the principal's domain group membership, derived from an **external security provider**.

@ExternalTokenTime: A **datetime** in **UTC** format specifying the most recent time the *@ExternalToken* value was updated.

@IsDomainGroup: A bit flag specifying whether the principal to be added is a domain group. If this is set to 1, the principal being added is a domain group. If this is set to 0, then the principal is a user. This parameter MUST NOT be NULL.

@IsActive: A bit flag specifying if the principal is a user not marked as deleted in the site collection. When set to 1, the principal is a user not marked as deleted in the site collection. If this is set to 0, then the principal is a user marked as deleted in the site collection. This parameter MUST NOT be NULL.

@Login: The login name of the principal to be added. This parameter MUST NOT be NULL.

@Title: The display name of the principal to be added. This parameter MUST NOT be NULL.

@Email: The email address of the principal to be added. This parameter MUST NOT be NULL.

@Notes: A string containing notes about the principal to be added. This parameter MUST NOT be NULL.

@MobilePhone: The mobile phone of the principal to be added.

@Flags: A **UserInfo Flags** (section 2.2.2.11) specifying the principal's options.

@MembershipWebId: The site identifier (section 2.2.1.11) of the site within the same site collection that the principal will be added as a **member** of.

@IncrementUserCount: A bit flag specifying whether to increment the user count of the site collection. When this parameter is set to 1, the user count in the site collection MUST be incremented.

@ImportDeleted: A bit flag specifying whether the principal is to be added as deleted. When this parameter is set to 1, the user information for the principal MUST be marked as deleted. The deleted state is set within the **UserInfo** table, rather than dropping entries from the **UserInfo** table, to preserve list item ownership information.

@AddedToTable: An output parameter indicating that an entry for the principal has been added to the **UserInfo** table. Its value MUST be listed in the following table.

Value	Description
0	The principal already exists in the **UserInfo** table.
1	The user information for the principal has been added to the **UserInfo** table.

@UserIdOut: An output parameter which MUST contain the user identifier (section 2.2.1.13) of the added principal on successful completion.

@LoginOut: An output parameter which MUST contain the login name of the added principal on successful completion.

@TitleOut: An output parameter which MUST contain the display name of the added principal on successful completion. It MUST be NULL otherwise.

@EmailOut: An output parameter which MUST contain the email address of the added principal on successful completion. It MUST be NULL otherwise.

@NotesOut: An output parameter which MUST contain the notes about the added principal on successful completion. It MUST be NULL otherwise.

@MobilePhoneOut: An output parameter which MUST contain the mobile phone of the added principal on successful completion. It MUST be NULL otherwise.

@DeletedOut: An output parameter which can indicate the deleted state of the principal. Its value MUST be listed in the following table.

Value	Description
0	The user information has not been marked as deleted.
1	The user information has been marked as deleted.

@NeedtoAddtoList: An output parameter which can indicate if this user is already in the **UserInfo** list.

Value	Description
0	The user is already in the **UserInfo** list, we do not need to add it again.
1	The user is not in the **UserInfo** list, it needs to be added to UserInfo list later.

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

Return Values: The **proc_SecAddUser** stored procedure returns an integer return code, which MUST be listed in the following table.

Value	Description
0	Successful execution.

Value	Description
80	A principal with the same login name but a different **SystemID** already exists.
212	The site collection is locked against write operations occurring.
1359	An internal error has occurred.
1816	The maximum number of users that can be added to a site has been exceeded and no additional users can be added to the site.

The **proc_SecAddUser** stored procedure MUST NOT return any result sets.

3.1.5.54 proc_SecAddUserToSiteGroup

The **proc_SecAddUserToSiteGroup** stored procedure is invoked to add a user to a site **group** in the site collection. The user is added to the site group and the root site of the site collection, the user's **WSS user Token** (section 2.2.4.10) and status as active user is updated, and the change is logged to the change log and site audit log.

```
PROCEDURE proc_SecAddUserToSiteGroup(
      @SiteId                        uniqueidentifier,
      @GroupId                       int,
      @UserIDToBeAdded               int,
      @UserID                        int,
      @SiteAdmin                     bit,
      @BelongsToGroup                bit,
      @GroupOwnerId                  int,
      @CurrentUserIsOwner            bit,
      @RequestGuid                   uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) for a site collection that contains the site group specified by *@GroupId*.

@GroupId: A site group identifier (section 2.2.1.10) for the site group that the user specified by *@UserIDToBeAdded* refers to. *@GroupId* MUST refer to an existing site group within the site collection specified by *@SiteId*.

@UserIDToBeAdded: A user identifier (section 2.2.1.13) for the user being added to the site group specified by *@GroupId*. This value MUST refer to an existing user identifier for the site collection specified by *@SiteId*.

@UserID: The user identifier for the user who is adding the user specified by *@UserIDToBeAdded* to the site group specified by *@GroupId*. This value MUST refer to an existing user Identifier in the site collection specified by *@SiteId*.

@SiteAdmin: If this parameter is set to 1, the user specified by *@UserID* is a site collection administrator. If this parameter is set to 0, then the user specified by *@UserID* is not a site collection administrator.

@BelongsToGroup: If this parameter is set to 1, then the user specified by *@UserID* is a **member** of the site group. If this parameter is set to 0, then the user specified by *@UserID* is not a member of the site group.

@GroupOwnerId: The user identifier for the owner of the site group specified by *@GroupId*. *@GroupOwnerId* MUST refer to an existing owner, which can be either a user or another site group.

@CurrentUserIsOwner: If this parameter is set to 1, then the user specified by *@GroupOwnerId* is the owner of the site group specified by *@GroupId*. If this parameter is set to 0, the user is not the owner.

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

Return Values: The **proc_SecAddUserToSiteGroup** stored procedure returns an integer return code, which MUST be listed in the following table.

Value	Description
0	Successful execution or the user already belongs to the site group.
5	The user specified by *@UserID* does not have rights to modify the membership of the site group specified by *@GroupID*.

The **proc_SecAddUserToSiteGroup** stored procedure MUST NOT return any result sets.

3.1.5.55 proc_SecAddWebMembership

The **proc_SecAddWebMembership** stored procedure is invoked to associate an existing user within a site collection to a site within the same site collection.

```
PROCEDURE proc_SecAddWebMembership(
      @SiteId                        uniqueidentifier,
      @WebId                         uniqueidentifier,
      @UserId                        int,
      @RequestGuid                   uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the requested site specified by *@WebId* and requested user identifier (section 2.2.1.13) specified by *@UserId*.

@WebId: The site identifier (section 2.2.1.11) of the site to add a new user to. This value MUST refer to an existing site identifier within the site collection specified by *@SiteId*.

@UserId: The user identifier for the user to be added to membership of the site. This value MUST refer to an existing user identifier within the site collection specified by *@SiteId*.

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

Return Values: The **proc_SecAddWebMembership** stored procedure returns an integer return code, which MUST be listed in the following table.

Value	Description
0	Successful execution.
3	The site specified by *@WebId* was not found, or the ancestor site with unique permissions that defines the scope for the site specified by *@WebId* was not found.

The **proc_SecAddWebMembership** stored procedure MUST NOT return any result sets.

3.1.5.56 proc_SecChangeToInheritedList

The **proc_SecChangeToInheritedList** stored procedure is invoked to change the scope of the specified list to the specified site and remove any **role assignments** and permission settings specific to the list.

The permissions of any list items contained in the specified list which have unique permissions MUST NOT be changed.

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

@WebId: The site identifier (section 2.2.1.11) of the site containing the list specified by *@ListId*.

@ListId: The list identifier (section 2.2.1.5) of the list.

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

Return Values: The **proc_SecChangeToInheritedList** stored procedure returns an integer return code which MUST be 0, and MUST return one result set on successful completion. If the stored procedure encounters a failure, then no result set is returned.

3.1.5.56.1 Site Audit Mask Result Set

The Site Audit Mask Result Set contains information about the Audit Flags (section 2.2.2.1) associated with the specified Site. The Site Audit Mask Result Set MUST be returned on successful completion, and MUST return a single row. The Site Audit Mask Result Set is defined in Site Audit Mask Result Set, section 2.2.5.20.

3.1.5.57 proc_SecChangeToInheritedWeb

The **proc_SecChangeToInheritedWeb** stored procedure changes a site from having its own unique permissions to instead use the permissions inherited from its nearest ancestor with unique permissions. When a site is changed to have inherited permissions, all role definitions and permissions for each list and document library are set to inherit from the security scope that the site uses.

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

@SiteId: The site collection identifier (section 2.2.1.9) for the site collection containing the site to be changed specified by *@WebId*.

@WebId: The site identifier (section 2.2.1.11) for the site to change to use inherited permissions.

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

Return Values: The **proc_SecChangeToInheritedWeb** stored procedure returns an integer return code that MUST be in the following table.

Value	Description
0	Successful execution.
3	The site was not found, the site does not have unique permissions, or an ancestor site with unique permissions was not found.

The **proc_SecChangeToInheritedWeb** stored procedure MUST return two result sets on successful execution. If the stored procedure encounters a failure, then no result sets are returned.

3.1.5.57.1 Inherited Site Result Set

The Inherited Site Result Set returns the scope identifier (section 2.2.1.8) and site identifier (section 2.2.1.11) that the site specified by *@WebId* now inherits its permissions from. The Inherited Site Result Set MUST be returned on successful completion and MUST contain one row.

```
          ScopeId                        uniqueidentifier,
          {ParentWebPermAncestor}        uniqueidentifier;
```

ScopeId: The scope identifier for the scope that the changed site specified by *@WebId* now inherits from.

{ParentWebPermAncestor}: The site identifier for an ancestor site that the changed site specified by *@WebId* now derives permissions from.

3.1.5.57.2 Site Audit Mask Result Set

The Site Audit Mask Result Set contains information about the **Audit Flags** (section 2.2.2.1) associated with the site specified by *@WebId*. The Site Audit Mask Result Set MUST be returned on successful completion and MUST contain a single row.

The Site Audit Mask Result Set is defined in Site Audit Mask Result Set, section 2.2.5.20.

3.1.5.58 proc_SecChangeToUniqueScope

The **proc_SecChangeToUniqueScope** stored procedure sets a **securable object** such as a site, list, or document library to use its own unique security scope, instead of inheriting its security scope from the first ancestor with uniquely permissions.

```
    PROCEDURE proc_SecChangeToUniqueScope(
          @SiteId                        uniqueidentifier,
          @WebId                         uniqueidentifier,
          @OldScopeId                    uniqueidentifier,
          @CopyFromScopeId               uniqueidentifier,
          @Url                           nvarchar(260),
          @DocId                         uniqueidentifier,
          @bIsWeb                        bit,
          @UserId                        int,
          @CopyAnonymousMask             bit,
          @CopyRoleAssignments           bit,
          @ClearSubScopes                bit,
          @bBreakBySiteOwner             bit,
          @ReturnAuditMask               bit,
          @MaxScopeInList                int,
          @NewScopeId                    uniqueidentifier = NULL OUTPUT,
          @RequestGuid                   uniqueidentifier = NULL OUTPUT
    );
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the securable object specified by *@Url* or *@DocId* to be set to use a unique security scope.

@WebId: The site identifier (section 2.2.1.11) of the site that is or contains the securable object.

@OldScopeId: The scope identifier (section 2.2.1.8) for the original security scope of the securable object specified by *@Url* or *@DocId*.

@CopyFromScopeId: The scope identifier for a security scope to copy the administrator role, anonymous user permissions, and role assignments from for use as the new security scope. This parameter MUST NOT be NULL.

@Url: The store-relative form URL for the securable object. The securable object MUST be specified by the *@Url* or the *@DocId* parameter. The *@Url* parameter MUST be NULL to specify the securable object with the *@DocId* parameter.

@DocId: The **document identifier** (section 2.2.1.2) of the securable object. The *@DocId* parameter MUST be ignored and can be NULL if *@Url* specifies the securable object.

@bIsWeb: A bit flag specifying whether the securable object is a site. If this parameter is set to 1, then the securable object is a site, and all subsites which inherit their security scope from the site specified by *@SiteId* MUST have their inheritances changed to the new security scope. This parameter MUST be set to 0 if *@Url* does not point to a site, and this parameter MUST be set to 1 if *@Url* points to a site.

@UserId: Specifies the **principal** to be added to the new security scope in the administrator role, unless overridden by the *@bBreakBySiteOwner* parameter or the *@CopyFromScopeId* parameter.

@CopyAnonymousMask: A bit flag specifying whether to copy anonymous user permissions from the *@CopyFromScopeId* parameter into the new security scope. If this parameter is set to 1, then the permissions for anonymous user access MUST be copied from the *@CopyFromScopeId* parameter into the new security scope.

@CopyRoleAssignments: A bit flag specifying whether to copy the role assignments from the *@CopyFromScopeId* parameter into the new security scope. If this parameter is set to 1, then the role assignments MUST be copied from the *@CopyFromScopeId* parameter into the new security scope. If both *@bBreakBySiteOwner* and *@CopyRoleAssignments* are set to 1, then the setting of *@CopyRoleAssignments* will take precedence. The role assignments are copied from the security scope provided in *@CopyFromScopeId*.

@ClearSubScopes: A bit flag specifying whether to change every site, list and document library under the URL to the new security scope or to change only inheriting ones to the new security scope. If this parameter is set to 1, then every site, list and document library under the URL MUST be changed to the new security scope, including no inheriting ones.

@bBreakBySiteOwner: A bit flag specifying whether to use the site owner in the administrator role instead of the principal specified by *@UserId*. If this parameter is set to 1, then the site owner MUST be added to the administrator role in the new security scope definition. If both *@bBreakBySiteOwner* and *@CopyRoleAssignments* are set, then the setting of *@CopyRoleAssignments* will take precedence. The role assignments are copied from the security scope provided in *@CopyFromScopeId*.

@ReturnAuditMask: A bit flag specifying whether to return a Site Audit Mask Result Set. If this parameter is set to 1, then a Site Audit Mask Result Set MUST be returned on successful completion.

@MaxScopeInList: An integer value specifying the maximum number of unique security scopes under a list. If a list item's security scope is changed and the list including this list item already has *@MaxScopeInList* unique security scopes, then the stored procedure MUST return an error using the integer return code 1340.

@NewScopeId: The scope identifier generated for the new security scope, returned as an output parameter.

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

Return Values: The **proc_SecChangeToUniqueScope** stored procedure returns an integer return code which MUST be listed in the following table.

Value	Description
0	Successful execution.
3	The specified securable object was not found at the specified location, the *@OldScopeId* does not

Value	Description
	match the securable object's scope identifier, or the securable object has unique permissions.
1340	A list item's security scope is changed and the list including this list item already has *@MaxScopeInList* unique security scopes.

If *@ReturnAuditMask* has a value of 1, then the **proc_SecChangeToUniqueScope** stored procedure MUST return a single result set on successful completion; otherwise, zero result sets MUST be returned.

3.1.5.58.1 Site Audit Mask Result Set

The Site Audit Mask Result Set contains information about the **Audit Flags** (section 2.2.2.1) associated with the site. If *@ReturnAuditMask* has a value of 1, the Site Audit Mask Result Set MUST be returned on successful completion, and MUST contain only one row. The Site Audit Mask Result Set is defined in Site Audit Mask Result Set, (section 2.2.5.20).

3.1.5.59 proc_SecCheckDeletedAccounts

The **proc_SecCheckDeletedAccounts** stored procedure is invoked to check if a login name exists in the site collection.

```
PROCEDURE proc_SecCheckDeletedAccounts(
      @SiteId                        uniqueidentifier,
      @Login                         nvarchar(255),
      @RequestGuid                   uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) for the site collection.

@Login: The login name of the **principal** as specified by the **security provider** in use. This parameter MUST NOT be NULL.

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

Return Values: The **proc_SecCheckDeletedAccounts** stored procedure returns an integer return code, which MUST be 0, and it MUST return a single result set.

3.1.5.59.1 Login Result Set

If a valid login name is found in the site collection, the Login result set MUST return one row containing the login name for each time *@Login* is found for the site collection specified by *@SiteId* within the **UserInfo** table (section 2.2.7.10); otherwise zero rows are returned.

```
      tp_Login                       nvarchar(255);
```

tp_Login: The login name for the user.

3.1.5.60 proc_SecCloneRoleDefinitions

The **proc_SecCloneRoleDefinitions** stored procedure creates a copy of the current role definition for a site. After successful execution, the site will have its own copy of the role definition and unique role assignments, and the site and all subsites that inherit permissions will use the new role definition.

```
PROCEDURE proc_SecCloneRoleDefinitions(
        @SiteId                         uniqueidentifier,
        @WebId                          uniqueidentifier,
        @CopyRoleAssignments            bit,
        @UserId                         int,
        @NewScopeId                     uniqueidentifier OUTPUT,
        @RequestGuid                    uniqueidentifier = NULL OUTPUT
  );
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the site.

@WebId: The site identifier (section 2.2.1.11) of the site which will have its current role definition copied.

@CopyRoleAssignments: Specifies whether to keep the current role assignments. If this parameter is set to 1, the current role assignments will kept. If this parameter is set to 0, then the user specified by *@UserId* will be added to the Administrator Role, and everyone else will be removed from all roles. *@CopyRoleAssignments* MUST NOT be NULL.

@UserId: The user identifier (section 2.2.1.13) of the current user. *@UserId* is assigned to the administrator role when *@CopyRoleAssignments* is set to 0. This value MUST refer to an existing user identifier for the specified site collection.

@NewScopeId: An output parameter which contains a scope identifier (section 2.2.1.8) for the security scope of the site. If the site already has unique role assignments before this call, then a new security scope MUST NOT be generated, and the output parameter MUST be the original scope identifier of the site. If the site does not have unique role assignments before this call, then a new security scope MUST be generated for this site, and the scope identifier of the new security scope MUST be returned in the output parameter.

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

Return Values: The **proc_SecCloneRoleDefinitions** stored procedure returns an integer return code, which MUST be listed in the following table.

Value	Description
0	Successful execution.
3	The site was not found. Either the site specified by *@SiteId/@WebId* does not exist, or the site already has its own role definition.
80	The site has a role definition with the same title as one of the role definitions that will be cloned.

The **proc_SecCloneRoleDefinitions** stored procedure MUST return either one or two result sets on successful execution, both of which used the same result set definition.

3.1.5.60.1 Site Audit Mask Result Set

The Site Audit Mask Result Set contains the information about the **Audit Flags** (section 2.2.2.1) associated with the specified site. On successful execution, the Site Audit Mask Result Set MUST be returned only once if *@CopyRoleAssignments* is set to 1. If *@CopyRoleAssignments* is set to 0, then the Site Audit Mask Result Set MUST be returned twice on successful execution.

The Site Audit Mask Result Set MUST contain a single row. The Site Audit Mask Result Set is defined Site Audit Mask Result Set, section 2.2.5.20.

3.1.5.61 proc_SecCreateSiteGroup

The **proc_SecCreateSiteGroup** stored procedure is invoked to add a new site **group** to a site collection.

```
PROCEDURE proc_SecCreateSiteGroup(
      @SiteId                      uniqueidentifier,
      @Title                       nvarchar(255),
      @Description                 nvarchar(512),
      @OwnerID                     int,
      @OwnerIsUser                 bit,
      @FirstMemberId               int,
      @UseExisting                 bit,
      @SelfOwner                   bit,
      @GroupID                     int                   OUTPUT,
      @RequestGuid                 uniqueidentifier = NULL     OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the new site group to be created.

@Title: The title of the new site group. This parameter MUST NOT be NULL.

@Description: A description of the new site group.

@OwnerID: A user identifier (section 2.2.1.13) or site group identifier (section 2.2.1.10) for the new site group's owner.

@OwnerIsUser: A bit flag specifying whether the site group owner specified by *@OwnerID* is a user or a site group.

- When *@OwnerIsUser* is set to 1, the new site group owner is a user in the site collection.
- When *@OwnerIsUser* is set to 0, the owner is a site group.

@FirstMemberId: A user identifier for the first **member** of the new site group. This value MUST correspond to an existing user id or be NULL. If this value is NULL then the site group MUST be created with no members.

@UseExisting: A bit flag specifying whether to return the site group identifier of an existing site group with a title matching *@Title* or to create a new site group.

- When *@UseExisting* is set to 1, the procedure MUST return an existing site group with the same title as the *@Title* input parameter, and if such a site group is not found, then it MUST create a new site group.
- When *@UseExisting* is set to 0, the procedure MUST return a failure code value of 80 if an existing site group with a matching title is found, and if such a site group is not found, then it MUST create a new site group.

@SelfOwner: A bit flag specifying whether or not the site group is its own owner. When *@SelfOwner* is set to 1, the site group MUST be set as its own owner, and the values of *@OwnerID* and *@OwnerIsUser* MUST be ignored.

@GroupID: An output parameter holding the integer site group identifier for the newly created or existing site group. If the procedure fails, then the value of this parameter SHOULD be ignored.

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

Return Values: The **proc_SecCreateSiteGroup** stored procedure returns an integer return code, which MUST be included in the following table.

Value	Description
0	Successful execution.
80	Failed to create the site group because a site group with an identical title exists in the site collection.

The **proc_SecCreateSiteGroup** stored procedure MUST return no result sets.

3.1.5.62 proc_SecDecCurrentUsersCount

The **proc_SecDecCurrentUsersCount** stored procedure is invoked to reduce by one the total number of users in the specified site collection when configured to use Active Directory Creation Mode. **proc_SecDecCurrentUsersCount** does not delete or update any user information.

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

@SiteID: The site collection identifier (section 2.2.1.9) of the site collection.

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

Return Values: The **proc_SecDecCurrentUsersCount** stored procedure returns an integer return code, which MUST be 0.

The **proc_SecDecCurrentUsersCount** stored procedure MUST return no result sets.

3.1.5.63 proc_SecGetAccountStatus

The **proc_SecGetAccountStatus** stored procedure provides status information for a site collection's users, which are not marked as deleted, and match the specified login name or email address.

```
PROCEDURE proc_SecGetAccountStatus(
      @SiteId                        uniqueidentifier,
      @Login                         nvarchar(255),
      @Email                         nvarchar(255),
      @RequestGuid                   uniqueidentifier = NULL     OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the requested Users.

@Login: The login name of a user to be matched.

@Email: The email address of a user to be matched. If this parameter is an empty string, it is ignored.

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

Return Values: The **proc_SecGetAccountStatus** stored procedure MUST return an integer return code of 0.

The **proc_SecGetAccountStatus** stored procedure MUST return one result set as follows.

3.1.5.63.1 Account Status Result Set

The Account Status Result Set returns account information for the non-deleted Users matching the specified login name or email address. This result set MUST be returned and MUST contain one row for each user which is not a domain group. The Account Status Result Set MUST contain no rows if no matching Users are found.

```
          tp_Login                       nvarchar(255),
          tp_Email                       nvarchar(255),
          tp_SystemID                    varbinary(512);
```

tp_Login: The login name of the matching user.

tp_Email: The email address of the matching user.

tp_SystemID: The **SystemID** (section 2.2.1.12) of the matching user.

3.1.5.64 proc_SecGetAclFromScope

The **proc_SecGetAclFromScope** stored procedure is invoked to obtain the Access Control List and the anonymous user permissions for a given scope.

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

@SiteId: The site collection identifier (section 2.2.1.9) for the site collection containing the scope.

@ScopeId: The scope identifier (section 2.2.1.8) for the scope for which the Access Control List and anonymous user permissions are requested.

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

Return Values: The **proc_SecGetAclFromScope** stored procedure returns an integer return code which MUST be 0.

The **proc_SecGetAclFromScope** stored procedure MUST return a single result set as defined as follows.

3.1.5.64.1 ACL and Permission Result Set

The ACL and Permission Result Set contains permission information for the specified scope. If either of the required input parameters is not valid, then the ACL and Permission Result Set MUST contain zero rows; otherwise, one row MUST be returned. The ACL and Permission Result Set is defined in ACL and Permission Result Set, section 2.2.5.1.

3.1.5.65 proc_SecGetAllAclsForSite

The **proc_SecGetAllAclsForSite** stored procedure is invoked to list all scope identifiers (section 2.2.1.8) and their associated ACL and anonymous permission masks in a site collection.

```
    PROCEDURE proc_SecGetAllAclsForSite(
          @SiteId                        uniqueidentifier,
          @MaxCount                      int,
          @RowCount                      int                   out,
          @RequestGuid                   uniqueidentifier = NULL    OUTPUT
    );
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection for which the information is requested.

@MaxCount: The maximum size, in rows, allowed in the result set. If the number of unique scopes within the site collection exceeds *@MaxCount*, the Access Control List Result Set will not be returned.

@RowCount: Output parameter indicating the number of unique scopes that exist in the site collection.

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

Return Values: The **proc_SecGetAllAclsForSite** stored procedure MUST return an integer return code of 0.

If the total count of unique Access Control Lists in the site collection does not exceed the value of the input parameter *@MaxCount*, then the **proc_SecGetAllAclsForSite** stored procedure MUST return the Access Control List Result Set. Otherwise, 0 result sets MUST be returned.

3.1.5.65.1 Access Control List Result Set

The Access Control List Result Set returns one row for each unique scope and associated Access Control List defined in the site collection.

```
      ScopeId                          uniqueidentifier,
      Acl                              varbinary(max),
      AnonymousPermMask                bigint;
```

ScopeId: The scope identifier (section 2.2.1.8) of the scope.

Acl: The Access Control List (ACL) associated with the scope.

AnonymousPermMask: The **WSS Rights Mask** (section 2.2.2.14) for anonymous users for this scope.

3.1.5.66 proc_SecGetAllGroupsAndMembershipInfo

The **proc_SecGetAllGroupsAndMembershipInfo** stored procedure is invoked to return information about all site **groups** and site group members within a site collection.

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

@SiteID: The site collection identifier (section 2.2.1.9) of the site collection to find all site groups in.

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

Return Code Values: The **proc_SecGetAllGroupsAndMembershipInfo** stored procedure returns an integer which MUST be 0.

The **proc_SecGetAllGroupsAndMembershipInfo** stored procedure MUST return two result sets in the following order.

3.1.5.66.1 Groups Result Set

The Groups Result Set returns all site **groups** within the site collection, one row per site group, sorted by site group identifier (section 2.2.1.10), in ascending order. The Groups Result Set MUST be empty if there is no site group on the site collection.

```
ID                              int,
Title                           nvarchar(255),
Description                     nvarchar(512),
Owner                           int,
OwnerIsUser                     bit;
```

ID: The site group identifier of the site group.

Title: The title of the site group.

Description: The description of the site group.

Owner: The user identifier (section 2.2.1.13) or site group identifier of the owner of the site group.

OwnerIsUser: A bit value indicating whether the owner of the group is a user or a site group. If the owner is a user, this parameter's value MUST be 1, otherwise, its value MUST be 0.

3.1.5.66.2 Group Membership Result Set

The Group Membership Result Set returns information about site **group** members who are not marked as deleted. A **member** can be a user or a domain group. Every row represents one site group member. The first column is the **GroupId**, the site group identifier (section 2.2.1.10), followed by columns about the site group member. The Group Membership Result Set is sorted by **GroupId**, in ascending order. If a member belongs to more than one site group, then the member's information will appear in multiple rows, with different **GroupIds**. The Group Membership Result Set MUST be empty if none of the site groups have any members.

```
GroupId                         int,
tp_Id                           int,
tp_SystemID                     varbinary(512),
tp_Title                        nvarchar(255),
tp_Login                        nvarchar(255),
tp_Email                        nvarchar(255),
tp_Notes                        nvarchar(1023),
tp_SiteAdmin                    bit,
tp_DomainGroup                  bit,
tp_Flags                        int;
```

GroupId: The site group identifier of the site group which contains this member.

tp_Id: The user identifier (section 2.2.1.13) of the member who belongs to the site group specified by the **GroupId** in the first column.

tp_SystemID: The **SystemID** (section 2.2.1.12) of the member.

tp_Title: The display name of the member.

tp_Login: The login name of the member.

tp_Email: The email address of the member. This parameter can be empty, but it MUST NOT be NULL.

tp_Notes: A string which contains extra information about the member. It can be empty, but it MUST NOT be NULL.

tp_SiteAdmin: A bit value indicating whether the member is a site collection administrator. If the member is a site collection administrator, this parameter's value MUST be 1, otherwise, it MUST be 0.

tp_DomainGroup: A bit value indicating whether the member is a domain group. If the site group member is a domain group, this parameter's value MUST be 1. Otherwise, its value MUST be 0.

tp_Flags: A 4-byte integer bit mask determining the user's options. See section 2.2.2.11.

3.1.5.67 proc_SecGetApplicationPrincipalAndUserToken

The **proc_SecGetApplicationPrincipalAndUserToken** stored procedure is invoked to return information about a **principal** based on the principal's login name and user identifier (section 2.2.1.13).

```
PROCEDURE proc_SecGetApplicationPrincipalAndUserToken (
      @SiteId                        uniqueidentifier,
      @AppLogin                      nvarchar(255),
      @UserId                        int,
      @RequestGuid                   uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the principal.

@AppLogin: The principal's login name.

@UserId: The user identifier of the principal.

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

Return values: The **proc_SecGetApplicationPrincipalAndUserToken** stored procedure returns an integer return code, which MUST be 0.

If *@UserId* is NULL, then the **proc_SecGetApplicationPrincipalAndUserToken** stored procedure MUST return one result set. If *@UserId* is not NULL, the **proc_SecGetApplicationPrincipalAndUserToken** stored procedure MUST return two result sets in the order specified.

3.1.5.67.1 Application Principal Result Set

The Application Principal Result Set returns information on a **principal** specified by *@AppLogin*. The **proc_SecGetApplicationPrincipalAndUserToken** stored procedure MUST return the Application Principal Result Set.

If *@AppLogin* does not match a principal, the Application Principal Result Set MUST be returned with zero rows. Otherwise, the Application Principal Result Set MUST be returned with a single row of data for the principal.

```
      tp_ID                          int,
      tp_IsActive                    bit,
      tp_Title                       nvarchar(255),
      tp_Email                       nvarchar(255),
      tp_Token                       varbinary(max),
      tp_Flags                       int;
```

tp_ID: The user identifier of the principal.

tp_IsActive: Set to "1" if the current user is an active user in the site collection.

tp_Title: The principal's display name.

tp_Email: The principal's email address.

tp_Token: A **WSS user Token** (section 2.2.4.10) value specifying the site **group** membership of the principal.

tp_Flags: Contains the **UserInfo Flags** (section 2.2.2.11) value associated with this principal.

3.1.5.67.2 External Token Result Set

The External Token Result Set returns information on a **principal** specified by *@UserId*. If *@UserId* is NULL, then the **proc_SecGetApplicationPrincipalAndUserToken** stored procedure MUST NOT return the External Token Result Set. If *@UserId* does not match a principal, then the External Token Result Set MUST be returned with zero rows. Otherwise, the External Token Result Set MUST be returned with a single row of data for the principal.

```
          tp_ExternalToken               varbinary(max);
```

tp_ExternalToken: An **External Group Token** (section 2.2.4.2) value encoding information on external **group** membership derived from an external authentication role provider.

3.1.5.68 proc_SecGetCompleteWebRoleMemberList

The **proc_SecGetCompleteWebRoleMemberList** stored procedure is invoked to list all role assignments for all the **principals** with permissions on a specified site.

```
    PROCEDURE proc_SecGetCompleteWebRoleMemberList(
          @SiteId                        uniqueidentifier,
          @WebId                         uniqueidentifier,
          @LatestSecurityVersion         bigint              OUTPUT,
          @RequestGuid                   uniqueidentifier = NULL     OUTPUT
    );
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the site whose permission information is requested.

@WebId: The site identifier (section 2.2.1.11) of the site whose permission information is requested.

@LatestSecurityVersion: The current security version value for the specified site collection.

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

Return Values: The **proc_SecGetCompleteWebRoleMemberList** stored procedure returns an integer return code which MUST be 0.

The **proc_SecGetCompleteWebRoleMemberList** stored procedure MUST return one result set.

3.1.5.68.1 Role Member Result Set

The Role Member Result Set returns one row for each **principal's** role assignment for the specified Site. If the *@SiteId* or *@WebId* parameters are not valid, then zero rows MUST be returned.

```
          RoleId                         int,
          tp_Id                          int,
          tp_SystemID                    varbinary(512),
          tp_DomainGroup                 bit;
```

RoleId: The role identifier of the role for the role assignment.

tp_Id: The site **group** identifier (section 2.2.1.10) or user identifier (section 2.2.1.13) of the principal.

tp_SystemID: The **SystemID** (section 2.2.1.12) of the principal.

tp_DomainGroup: Set to 1 if the principal is a domain group; otherwise, 0.

3.1.5.69 proc_SecGetCurrentUsersCount

The **proc_SecGetCurrentUsersCount** stored procedure returns a result set containing a count of users in the specified site collection.

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

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection.

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

Return Values: The **proc_SecGetCurrentUsersCount** stored procedure MUST return an integer return code of 0.

The **proc_SecGetCurrentUsersCount** stored procedure MUST return a single result set for the specified site collection as follows.

3.1.5.69.1 User Count Result Set

The user Count Result Set returns the number of users registered with this site collection when configured in Active Directory Creation mode. When not in Active Directory creation mode, **UsersCount** will be 1. The user Count Result Set MUST return one row for a given valid *@SiteId*, otherwise if *@SiteId* is invalid, zero rows MUST be returned.

```
      UsersCount                      int,
      UserQuota                       int,
      {StorageQuotaError}             int;
```

UsersCount: Contains the integer value for the number of users registered with the specified site collection when configured in Active Directory creation mode. In any other configuration, **UsersCount** MUST return 1.

UserQuota: Contains the limit for the number of Users allowed in the specified site collection. A value of 0 specifies no limit on the number of allowed users.

{StorageQuotaError}: An error number generated when a site collection is over quota or locked. Its value MUST be listed in the following table.

Value	Description
0	Default value, no error.
212	The site collection is locked.
1816	Quota has been exceeded on this site collection.

3.1.5.70 proc_SecGetDomainGroupMapData

The **proc_SecGetDomainGroupMapData** stored procedure is invoked to retrieve the domain group map cache information for a site collection.

```
PROCEDURE proc_SecGetDomainGroupMapData(
      @SiteId                        uniqueidentifier,
      @DGCacheVersion                bigint,
      @RequestGuid                   uniqueidentifier = NULL     OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9).

@DGCacheVersion: The version number of the Domain Group Map Cache in the front-end web server.

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

Return Values: The **proc_SecGetDomainGroupMapData** stored procedure returns an integer return code which MUST be 0.

The **proc_SecGetDomainGroupMapData** stored procedure MUST return 1 or 2 of the following result sets.

3.1.5.70.1 Domain Group Cache Versions Result Set

The Domain Group Cache Versions Result Set MUST be returned and MUST contain one row of version number data. If the specified *@DGCacheVersion* value is -2, then all columns returned MUST have the value -2, indicating that the value MUST NOT be used for comparison.

The Domain Group Cache Versions Result Set is defined in Domain Group Cache Versions Result Set section 2.2.5.4.

3.1.5.70.2 Domain Group Cache BEDS Update Result Set

The Domain Group Cache BEDS Update Result Set MUST be returned if *@DGCacheVersion* does not equal "-2" and the value of **RealVersion** is greater than the value of **CachedVersion** in the results of the Domain Group Cache Versions Result Set (section 2.2.5.4). Otherwise, the Domain Group Cache BEDS Update Result Set MUST NOT be returned.

If the Domain Group Cache BEDS Update Result Set is returned, then there MUST be one row in the Domain Group Cache BEDS Update Result Set for each domain group, which is a **member** of a site **group** in the site collection, ordered by the identifier of the domain groups. A row is also returned for every other domain group that does not have a **GroupId** tied to it, including domain groups with NULL **GroupIds**.

The Domain Group Cache BEDS Update Result Set is defined in Domain Group Cache BEDS Update Result Set, section 2.2.5.3.

3.1.5.70.3 Domain Group Cache WFE Update Result Set

The Domain Group Cache WFE Update Result Set MUST be returned if *@DGCacheVersion* does not equal "-2" and the value of **RealVersion** is less than or equal to the value of **CachedVersion** in the results of the Domain Group Cache Versions Result Set (section 2.2.5.4). Otherwise, the Domain Group Cache WFE Update Result Set MUST NOT be returned.

If the Domain Group Cache WFE Update Result Set is returned, then it MUST contain one row.

The Domain Group Cache WFE Update Result Set is defined in Domain Group Cache WFE Update Result Set, section 2.2.5.5.

3.1.5.71 proc_SecGetGroupById

The **proc_SecGetGroupById** stored procedure is invoked to check whether the specified **group** (either a site group or a domain group) exists in the specified site collection.

```
PROCEDURE proc_SecGetGroupById(
      @SiteId                        uniqueidentifier,
      @GroupId                       int,
      @Count                         int                  OUTPUT,
      @RequestGuid                   uniqueidentifier = NULL      OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection to search for the group.

@GroupId: The identifier (a site group identifier (section 2.2.1.10) for site groups, or a user identifier (section 2.2.1.13) for domain groups) of the specified group.

@Count: Specifies whether the specified group exists in the site collection as either a site group or a domain group. If the specified group exists, *@Count* MUST be 1; otherwise, *@Count* MUST be 0.

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

Return Code Values: The **proc_SecGetGroupById** stored procedure returns an integer, which MUST be 0.

The **proc_SecGetGroupById** stored procedure MUST NOT return any result sets.

3.1.5.72 proc_SecGetGroupOwner

The **proc_SecGetGroupOwner** stored procedure is invoked to retrieve the user identifier (section 2.2.1.13) or site **group** identifier (section 2.2.1.10) for the owner of a site group. User identifier and group identifier do not collide within the same site collection.

```
PROCEDURE proc_SecGetGroupOwner(
      @SiteId                        uniqueidentifier,
      @GroupId                       int,
      @OwnerId                       int OUTPUT,
      @RequestGuid                   uniqueidentifier = NULL     OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) for the site collection containing the site group.

@GroupId: The site group identifier.

@OwnerId: An output parameter containing the user identifier or site group identifier of the site group owner, or "-1" if the specified site group does not exist.

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

Return Values: The **proc_SecGetGroupOwner** stored procedure returns an integer return code which MUST be listed in the following table.

Value	Description
0	Successful execution.

Value	Description
1319	The site group does not exist.

The **proc_SecGetGroupOwner** stored procedure MUST NOT return a result set.

3.1.5.73 proc_SecGetGroupSecurityScopes

The **proc_SecGetGroupSecurityScopes** stored procedure is invoked to retrieve a site **group's** role assignments information on all security scopes within a given site collection.

```
PROCEDURE proc_SecGetGroupSecurityScopes(
      @SiteId                        uniqueidentifier,
      @PrincipalId                   int,
      @RequestGuid                   uniqueidentifier = NULL    OUTPUT,
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the site group.

@PrincipalId: The site group identifier (section 2.2.1.10).

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

Return Values: The **proc_SecGetGroupSecurityScopes** stored procedure MUST return an integer return code of 0.

The **proc_SecGetGroupSecurityScopes** stored procedure MUST return the following result set.

3.1.5.73.1 Security Scopes Result Set

The Security Scopes Result Set returns scope information for the specified **security principal**. The Security Scopes Result Set will be returned with zero or more rows.

```
      ScopeUrl                       nvarchar(260),
      Title                          nvarchar(255);
```

ScopeUrl: The URL to the root of the Security Scope.

Title: The name of the role to which the site **group** is assigned on the Security Scope specified by **{ScopeUrl}**.

3.1.5.74 proc_SecGetIndividualUrlSecurityCheckEventReceivers

The **proc_SecGetIndividualUrlSecurityCheckEventReceivers** stored procedure is invoked to request security information and event receivers information about a document at a specified location. This stored procedure can lock the table of lists if the *@bLockLists* is set to 1. If the document does not exist, **proc_SecGetIndividualUrlSecurityCheckEventReceivers** provides security information about the specified location.

```
PROCEDURE proc_SecGetIndividualUrlSecurityCheckEventReceivers(
      @SiteId                        uniqueidentifier,
      @WebId                         uniqueidentifier,
      @FullUrl                       nvarchar(260),
      @DirName                       nvarchar(256),
      @LeafName                      nvarchar(128),
      @UserId                        int,
      @AttachmentsFlag               tinyint,
      @bGetAttachmentWritePerm        bit,
      @bGetListMetaData               bit                     = 0,
      @bGetListScopes                 bit                     = 0,
      @bLockLists                     bit                     = 0,
      @Level                          tinyint                 = NULL,
      @HasEventReceiver               bit                     OUTPUT,
      @MinLevel                       tinyint                 OUTPUT,
      @RequestGuid                    uniqueidentifier        OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the document location.

@WebId: The site identifier (section 2.2.1.11) of the site containing the document location.

@FullUrl: The store-relative form URL of the document location, used to find a containing list or document library if the document does not exist.

@DirName: The directory name of the document location.

@LeafName: The leaf name of the document location.

@UserId: The user identifier (section 2.2.1.13) of the current user, used to check for access privileges.

@AttachmentsFlag: An **Attachments Flag** (section 2.2.3.1) value specifying whether the document location is, or is contained within, an attachments folder.

@bGetAttachmentWritePerm: A bit flag specifying whether or not to return information about the write permissions of the current user to save an attachment at the specified document location. If this parameter is set to 1, and the *@AttachmentsFlag* parameter is not 0 or NULL, then the stored procedure MUST return the information about the current user's permissions as part of the Individual URL Security Result Set (section 3.1.5.74.1). This parameter MUST be ignored if *@AttachmentsFlag* is 0 or NULL.

@bGetListMetaData: A bit flag specifying whether or not metadata for the list or document library containing the specified document is requested.

@bGetListScopes: A bit flag specifying whether or not security scope information is requested for the list or document library containing the document location.

@bLockLists: A bit flag specifying whether or not to prevent updates to the table of lists when obtaining the list identifier (section 2.2.1.5) of the list or document library containing the document location. If this parameter is set to 1, the table MUST be locked while obtaining the list identifier.

@Level: If not set to NULL, then this parameter specifies the **Publishing Level** type (section 2.2.2.6) value to return in the Individual URL Security Result Set. If this parameter is NULL, the publishing level of the current version of the document for the current user MUST be used.

@HasEventReceiver: An output parameter that MUST be set to 1 if at least one event receiver is registered for the specified document; otherwise it is set to 0.

@MinLevel: A **Publishing Level** type value that indicates the lowest value of publishing levels present for the document, returned as an output parameter.

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

Return Values: The **proc_SecGetIndividualUrlSecurityCheckEventReceivers** stored procedure returns an integer return code which MUST be 0.

The **proc_SecGetIndividualUrlSecurityCheckEventReceivers** stored procedure MUST return between 1 and 5 result sets according to the input parameters. See result set descriptions for more details. The result sets are returned in the following listed order.

3.1.5.74.1 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 URL is within a list or document library, security information is returned for the specified document location.

The Individual URL Security Result Set MUST be returned if the specified document location is contained within a list or document library. Otherwise, the NULL Individual URL Security Result Set (section 3.1.5.74.2) 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 Individual URL Security Result Set, section 2.2.5.10.

3.1.5.74.2 NULL Individual URL Security Result Set

The NULL Individual URL Security Result Set indicates that the document location is not contained within a list or document library. The NULL Individual URL Security Result Set MUST only be returned if the specified document location is not contained within a List or document library.

The NULL Individual URL Security Result Set is defined in NULL Individual URL Security Result Set, section 2.2.5.14.

3.1.5.74.3 List Metadata Result Set

The List Metadata Result Set returns the metadata for the list or document library containing the specified document location. The List Metadata Result Set MUST only be returned if the *@bGetListMetadata* parameter is set to 1 and the document location is contained within a list or document library, otherwise, the List Metadata Result Set is not returned. If returned, then the List Metadata Result Set MUST contain a single row.

The List Metadata Result Set is defined in List Metadata Result Set, section 2.2.5.12.

3.1.5.74.4 Event Receivers Result Set

The Event Receivers Result Set contains information about event receivers defined for the containing list or document library. This result set MUST only be returned if the *@bGetListMetaData* parameter is set to 1 and the document location is contained within a list or document library, otherwise, the Event Receivers Result Set is not returned. If returned, then the Event Receivers Result Set MUST contain one row for each event receiver registered for the list or document library.

The Event Receivers Result Set is defined in Event Receivers Result Set, section 2.2.5.9.

3.1.5.74.5 List Scopes Result Set

The List Scopes Result Set returns the security information for the security scope associated with the document location. The List Scopes Result Set MUST only be returned if the *@bGetListScopes* parameter is set to 1 and the document location is within a list or document library, otherwise the List Scopes Result Set is not returned. If returned, then the List Scopes Result Set MUST contain one row.

```
ScopeId                        uniqueidentifier,
Acl                            varbinary(max),
AnonymousPermMask              bigint;
```

ScopeId: The scope identifier (section 2.2.1.8) for the scope associated with the document location.

Acl: The ACL in **WSS ACL Format** (section 2.2.4.6) of the scope associated with the document location.

AnonymousPermMask: The **WSS Rights Mask** (section 2.2.2.14) that applies to an anonymous user, or a user with no assigned rights, in the scope associated with the document location.

3.1.5.75 proc_SecGetItemsWithUniquePermissions

The **proc_SecGetItemsWithUniquePermissions** stored procedure is invoked to return information about list items with unique permissions.

```
PROCEDURE proc_SecGetItemsWithUniquePermissions (
      @SiteId                         uniqueidentifier,
      @WebId                          uniqueidentifier,
      @ListId                         uniqueidentifier,
      @TitleMode                      int,
      @MaxItemToReturn                int,
      @FolderOnly                     bit,
      @RequestGuid                    uniqueidentifier   = NULL OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the list items.

@WebId: The site identifier (section 2.2.1.11) of the site containing the list items.

@ListId: The list identifier (section 2.2.1.5) of the list containing the list items.

@TitleMode: *@TitleMode* determines how title information is returned. If the list is a document library, then *@TitleMode* MUST be 0. If the list has a **List Base** type (section 2.2.3.11) of issues list, or a **List Server Template** (section 2.2.3.12) of links list, then *@TitleMode* MUST be 1. Otherwise, *@TitleMode* MUST be 2.

@MaxItemToReturn: The maximum number of rows to be returned in the result set.

@FolderOnly: If this is not 0, then the result set MUST contain only information about folders. Otherwise, information about both folders and non-folders will be returned.

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

Return values: The **proc_SecGetItemsWithUniquePermissions** stored procedure returns an integer return code, which MUST be 0.

This **proc_SecGetItemsWithUniquePermissions** stored procedure MUST return one result set.

3.1.5.75.1 Items with Unique Permissions Result Set

The Items with Unique Permissions Result Set returns one row for each list item with unique permissions.

```
      ScopeUrl                        nvarchar(260),
      DoclibRowId                     int,
      {Title}                         nvarchar(255),
      IsFolder                        bit;
```

ScopeUrl: The URL to the root of the security scope.

DoclibRowId: The row identifier for the document within the list.

{Title}: If *@TitleMode* is 0, then **{Title}** MUST be the leaf name of the document. If *@TitleMode* is 1, then **{Title}** MUST be the list item identifier (section 2.2.1.6). If *@TitleMode* is 2, then **{Title}** MUST contain the title of this list item.

IsFolder: If this list item is a folder, **IsFolder** MUST be 1. Otherwise, IsFolder MUST be 0.

3.1.5.76 proc_SecGetPrincipalByEmail

The **proc_SecGetPrincipalByEmail** stored procedure is invoked to return user information about a user associated with a specified email address.

```
PROCEDURE proc_SecGetPrincipalByEmail(
      @SiteId                        uniqueidentifier,
      @Email                         nvarchar(255),
      @RequestGuid                   uniqueidentifier= NULL OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) for the site collection containing the user.

@Email: The user's email address. If this parameter is NULL, the email address MUST be the empty string.

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

Return Values: The **proc_SecGetPrincipalByEmail** stored procedure returns an integer return code, which MUST be 0.

The **proc_SecGetPrincipalByEmail** stored procedure MUST return a single result set.

3.1.5.76.1 Principal User Information Result Set

The Principal user Information Result Set returns information about the user associated with the specified email address. The Principal user Information Result Set MUST contain zero rows if no users have the specified email address or if the associated user has been marked as deleted. The Principal user Information Result Set is defined in Principal user Information Result Set, section 2.2.5.17.

3.1.5.77 proc_SecGetPrincipalById

The **proc_SecGetPrincipalById** stored procedure is invoked to return information about a **principal** or collection of principals based on a specified site **group** identifier (section 2.2.1.10) or user identifier (section 2.2.1.13).

```
PROCEDURE proc_SecGetPrincipalById(
      @SiteId                        uniqueidentifier,
      @PrincipalId                   int,
      @GetSTSToken                   bit                = 0,
      @GetExternalToken              bit                = 0,
      @GetExpandedSTSGroup           bit                = 0,
      @RequestGuid                   uniqueidentifier   = NULL OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the principal.

@PrincipalId: The site group identifier (section 2.2.1.10) of the site group or user identifier (section 2.2.1.13) of the principal to return information for.

@GetSTSToken: If this parameter is set to 1, then it indicates the **{tp_token}** value MUST be returned in both instances of the Principal user Information Result Set.

@GetExternalToken: If this parameter is set to 1, then it indicates the **{tp_ExternalTokenLastUpdated}** and **{tp_ExternalToken}** values MUST be returned in both instances of the Principal user Information Result Set.

@GetExpandedSTSGroup: If this parameter is set to 1, then it indicates a second Principal user Information Result Set MUST be returned when the first instance of the Principal user Information Result Set has zero rows. The second result set contains information about group membership for the site group represented by *@PrincipalId*. The rows returned in the second result set are all the members of that site group.

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

Return Values: The **proc_SecGetPrincipalById** stored procedure returns an integer return code, which MUST be 0.

The **proc_SecGetPrincipalById** stored procedure MUST return one result set which can be returned either once or twice as described in the following.

3.1.5.77.1 User Information Result Set

The user Information Result Set returns information on a specified **principal** or the members of a site **group**.

If *@PrincipalId* matches a principal, the user Information Result Set MUST be returned once with a single row of data for the principal.

Otherwise, the user Information Result Set MUST be returned with zero rows, and if *@GetExpandedSTSGroup* is 1, another instance of the user Information Result Set MUST be returned, with one row of data for each principal within the site group.

```
tp_ID                          int,
tp_SystemID                    varbinary(512),
tp_Title                       nvarchar(255),
tp_Login                       nvarchar(255),
tp_Email                       nvarchar(255),
tp_Notes                       nvarchar(1023),
tp_SiteAdmin                   bit,
tp_DomainGroup                 bit,
tp_Flags                       int,
{tp_ExternalTokenLastUpdated}  datetime,
{tp_ExternalToken}             varbinary(max),
{tp_Token}                     varbinary(max);
```

tp_ID: The user identifier (section 2.2.1.13) of the principal.

tp_SystemID: The principal's **SystemID** (section 2.2.1.12).

tp_Title: The principal's display name.

tp_Login: The principal's login name.

tp_Email: The principal's email address.

tp_Notes: A longer descriptive text associated with the principal.

tp_SiteAdmin: Set to 1 if the principal is a site collection administrator; otherwise, 0.

tp_DomainGroup: Set to 1 if the principal is a domain group; otherwise, 0.

tp_Flags: Contains the **UserInfo Flags** (section 2.2.2.11) value describing this principal.

{tp_ExternalTokenLastUpdated}: A timestamp in **UTC** format specifying the time when the **External Group Token** (section 2.2.4.2) was last updated. This parameter MUST be NULL if *@GetExternalToken* is 0.

{tp_ExternalToken}: An **External Group Token** value specifying the domain group membership of the principal. This parameter MUST be NULL if *@GetExternalToken* is 0.

{tp_Token}: A **WSS user Token** (section 2.2.4.10) value specifying the site group membership of the principal. This parameter MUST be NULL if *@GetSTSToken* is 0.

3.1.5.78 proc_SecGetPrincipalByIdEx

The **proc_SecGetPrincipalByIdEx** stored procedure is invoked to return information about a **principal** based on a specified user identifier (section 2.2.1.13) or a collection of principals based on a specified site **group** identifier (section 2.2.1.10). The **proc_SecGetPrincipalByIdEx** stored procedure is similar to the **proc_SecGetPrincipalById** stored procedure, except that the **proc_SecGetPrincipalByIdEx** stored procedure also returns the principal's mobile phone information.

```
PROCEDURE proc_SecGetPrincipalByIdEx (
      @SiteId                        uniqueidentifier,
      @PrincipalId                   int,
      @GetSTSToken                   bit                  = 0,
      @GetExternalToken              bit                  = 0,
      @GetExpandedSTSGroup           bit                  = 0,
      @RequestGuid                   uniqueidentifier     = NULL OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the principal(s).

@PrincipalId: The site group identifier of the site group or user identifier (section 2.2.1.13) of the principal(s) to return information for.

@GetSTSToken: If this parameter is set to 1, it indicates the **{tp_token}** value MUST be returned in both instances of the Principal user Information Result Set.

@GetExternalToken: If this parameter is set to 1, it indicates the **{tp_ExternalTokenLastUpdated}** and **{tp_ExternalToken}** values MUST be returned in both instances of the Principal user Information Result Set.

@GetExpandedSTSGroup: If this parameter is set to 1, it indicates a second Principal user Information Result Set MUST be returned when the first instance of the Principal user Information Result Set has zero rows. The second result set contains information about group membership for the site group represented by *@PrincipalId*. The rows returned in the second result set are all the members of that site group.

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

Return Values: The **proc_SecGetPrincipalById** stored procedure returns an integer return code, which MUST be 0.

The **proc_SecGetPrincipalByIdEx** stored procedure MUST return one result set which can be returned either once or twice as described in the following.

3.1.5.78.1 Extended Principal User Information Result Set

The Extended Principal user Information Result Set returns information on a specified **principal** or the principals that are members of a site **group**.

If *@PrincipalId* matches a principal, then the Extended Principal user Information Result Set MUST be returned once with a single row of data for the principal.

Otherwise, the Extended Principal user Information Result Set MUST be returned with zero rows, and if *@GetExpandedSTSGroup* is 1, another instance of the Extended Principal user Information Result Set MUST be returned, with one row of data for each principal within the site group.

```
          tp_ID                         int,
          tp_SystemID                   varbinary(512),
          tp_Title                      nvarchar(255),
          tp_Login                      nvarchar(255),
          tp_Email                      nvarchar(255),
          tp_Notes                      nvarchar(1023),
          tp_SiteAdmin                  bit,
          tp_DomainGroup                bit,
          tp_Flags                      int,
          {tp_ExternalTokenLastUpdated} datetime,
          {tp_ExternalToken}            varbinary(max),
          {tp_Token}                    varbinary(max),
          tp_Mobile                     nvarchar(127);
```

tp_ID: The user identifier (section 2.2.1.13) of the principal.

tp_SystemID: The principal's **SystemID** (section 2.2.1.12).

tp_Title: The principal's display name.

tp_Login: The principal's login name.

tp_Email: The principal's email address.

tp_Notes: A longer descriptive text associated with the principal.

tp_SiteAdmin: Set to 1 if the principal is a site collection administrator; otherwise, 0.

tp_DomainGroup: Set to 1 if the principal is a domain group; otherwise, 0.

tp_Flags: Contains the **UserInfo Flags** (section 2.2.2.11) value describing this principal.

{tp_ExternalTokenLastUpdated}: A **timestamp** in **UTC** format specifying the time when the **External Group Token** (section 2.2.4.2) was last updated. This parameter MUST be NULL if *@GetExternalToken* is 0.

{tp_ExternalToken}: An **External Group Token** value specifying the domain group membership of the principal. This parameter MUST be NULL if *@GetExternalToken* is 0.

{tp_Token}: A **WSS user Token** (section 2.2.4.10) value specifying the site group membership of the principal. This parameter MUST be NULL if *@GetSTSToken* is 0.

tp_Mobile: The principal's mobile phone number.

3.1.5.79 proc_SecGetPrincipalByLogin

The **proc_SecGetPrincipalByLogin** stored procedure is invoked to return security and attribute information for a **principal** (a user or domain group) identified by a specified login name.

```
    PROCEDURE proc_SecGetPrincipalByLogin(
          @SiteId                       uniqueidentifier,
          @Login                        nvarchar(255),
          @GetSTSToken                  bit                = 0,
          @GetExternalToken             bit                = 0,
        @RequestGuid                    uniqueidentifier    = NULL OUTPUT
    );
```

@SiteId: The site collection identifier (section 2.2.1.9) for the site collection associated with the principal whose information is requested.

@Login: The login name of the principal. This parameter MUST NOT be NULL.

@GetSTSToken: A bit flag specifying whether to include site **group** membership information for the principal. If this parameter is set to 1, then a **WSS user Token** (section 2.2.4.10) containing the principal's site group membership information MUST be returned in the user Information Result Set.

@GetExternalToken: A bit flag specifying whether to include domain group membership and related timestamp information for the principal. If this parameter is set to 1, then an **External Group Token** (section 2.2.4.2) containing the principal's domain group membership information and a timestamp indicating the principal's most recent update time MUST be returned in the user Information Result Set.

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

Return Values: The **proc_SecGetPrincipalByLogin** stored procedure returns an integer return code, which MUST be 0.

The **proc_SecGetPrincipalByLogin** stored procedure MUST return a single result set as follows.

3.1.5.79.1 User Information Result Set

The user Information Result Set returns security and attribute information about a specified **principal**. The user Information Result Set MUST be returned, and MUST contain one row if the *@Login* parameter matches an existing principal who is not marked as deleted in the **UserInfo** table (section 2.2.7.10); otherwise it MUST contain no rows.

```
        tp_ID                           int,
        tp_SystemID                     varbinary(512),
        tp_Title                        nvarchar(255),
        tp_Login                        nvarchar(255),
        tp_Email                        nvarchar(255),
        tp_Notes                        nvarchar(1023),
        tp_SiteAdmin                    bit,
        tp_DomainGroup                  bit,
        tp_Flags                        int,
        {tp_ExternalTokenLastUpdated}   datetime,
        {tp_ExternalToken}              varbinary(max),
        {tp_Token}                      varbinary(max);
```

tp_ID: The user identifier (section 2.2.1.13) for the specified principal.

tp_SystemID: The **SystemID** (section 2.2.1.12) for the principal.

tp_Title: The display name of the principal.

tp_Login: The principal's login name.

tp_Email: The principal's email address.

tp_Notes: A descriptive text string associated with the principal.

tp_SiteAdmin: A bit set to 1 if the principal has administrator rights in the site collection; otherwise, 0.

tp_DomainGroup: A bit set to 1 if the principal is a domain group; otherwise 0, indicating the principal is a user.

tp_Flags: Contains the **UserInfo Flags** (section 2.2.2.11) value describing this principal.

{tp_ExternalTokenLastUpdated}: A **datetime** in **UTC** format specifying the time when the **External Group Token** (section 2.2.4.2) was last updated. This value MUST be NULL if the *@GetExternalToken* parameter is not set to 1.

{tp_ExternalToken}: An **External Group Token** value specifying the domain group membership of the principal. This value MUST be NULL if the *@GetExternalToken* parameter is not set to 1.

{tp_Token}: A **WSS user Token** (section 2.2.4.10) value specifying the site **group** membership of the principal. This value MUST be NULL if the *@GetSTSToken* parameter is not set to 1.

3.1.5.80 proc_SecGetPrincipalByLogin20

The **proc_SecGetPrincipalByLogin20** stored procedure is invoked to return **security principal** information based on up to 20 separate login names.

```
PROCEDURE proc_SecGetPrincipalByLogin20(
      @SiteId                          uniqueidentifier,
      @PrincipalId01                   nvarchar(255),
      @PrincipalId02                   nvarchar(255),
      @PrincipalId03                   nvarchar(255),
      @PrincipalId04                   nvarchar(255),
      @PrincipalId05                   nvarchar(255),
      @PrincipalId06                   nvarchar(255),
      @PrincipalId07                   nvarchar(255),
      @PrincipalId08                   nvarchar(255),
      @PrincipalId09                   nvarchar(255),
      @PrincipalId10                   nvarchar(255),
      @PrincipalId11                   nvarchar(255),
      @PrincipalId12                   nvarchar(255),
      @PrincipalId13                   nvarchar(255),
      @PrincipalId14                   nvarchar(255),
      @PrincipalId15                   nvarchar(255),
      @PrincipalId16                   nvarchar(255),
      @PrincipalId17                   nvarchar(255),
      @PrincipalId18                   nvarchar(255),
      @PrincipalId19                   nvarchar(255),
      @PrincipalId20                   nvarchar(255),
      @RequestGuid                     uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) for the site collection that contains the security principals.

@PrincipalId##: The login names for users to be returned.

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

Return Values: The **proc_SecGetPrincipalByLogin20** stored procedure returns an integer return code which MUST be 0.

The **proc_SecGetPrincipalByLogin20** stored procedure MUST return one result set of security principal information for each non-NULL *@PrincipalId##*.

3.1.5.80.1 User Information Result Set

The user Information Result Set returns user information on a specified **security principal**. The user Information Result Set MUST return one row if the *@PrincipalId##* matches a security principal's ID, otherwise it MUST contain no rows. The user Information Result Set is defined in user Information Result Set, section 3.1.5.79.1.

3.1.5.81 proc_SecGetPrincipalDisplayInformation20

The **proc_SecGetPrincipalDisplayInformation20** stored procedure is invoked to return **security principal** or site **group** information for up to 20 **principal** identifiers.

```
PROCEDURE proc_SecGetPrincipalDisplayInformation20(
      @SiteId                        uniqueidentifier,
      @WebId                         uniqueidentifier,
      @PrincipalId01                 int,
      @PrincipalId02                 int,
      @PrincipalId03                 int,
      @PrincipalId04                 int,
      @PrincipalId05                 int,
      @PrincipalId06                 int,
      @PrincipalId07                 int,
      @PrincipalId08                 int,
      @PrincipalId09                 int,
      @PrincipalId10                 int,
      @PrincipalId11                 int,
      @PrincipalId12                 int,
      @PrincipalId13                 int,
      @PrincipalId14                 int,
      @PrincipalId15                 int,
      @PrincipalId16                 int,
      @PrincipalId17                 int,
      @PrincipalId18                 int,
      @PrincipalId19                 int,
      @PrincipalId20                 int,
      @RequestGuid                   uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) for the site collection containing the security principals and the Security Groups to be listed.

@WebId: A site identifier (section 2.2.1.11) for a site. This parameter is ignored.

@PrincipalId##: The identifier for a security principal or a site group to be returned. Result sets are returned for each non-NULL *@PrincipalId##* parameter.

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

Return Values: The **proc_SecGetPrincipalDisplayInformation20** stored procedure returns an integer return code which MUST be 0.

The **proc_SecGetPrincipalDisplayInformation20** stored procedure MUST return one or two result sets for each non-NULL *@PrincipalId##*.

3.1.5.81.1 Site Group Principal Display Information Result Set

The Site Group Principal Display Information Result Set returns information on a specified *@PrincipalId##*. The *@PrincipalId##* MUST match the ID of either a site **group** or a **principal**. If *@PrincipalId##* matches the ID of a site group, then the Site Group Principal Display Information Result Set MUST return one row. If the *@PrincipalId##* matches the ID of a principal, then the Site Group Principal Display Information Result Set MUST have no rows.

```
      IsUser                         bit,
      IsSiteGroup                    bit,
      UserID                         int,
      UserSID                         varbinary(512),
      UserName                        nvarchar(255),
      UserLogin                       nvarchar(255),
      UserEmail                       nvarchar(255),
      UserNotes                       nvarchar(1023),
      UserSiteAdmin                   bit,
      UserDomainGroup                 bit,
      GroupID                         int,
      GroupName                       nvarchar(255),
      GroupDescription                nvarchar(512),
      GroupOwnerID                    int,
      GroupOwnerIsUser                bit,
      GroupType                       tinyint;
```

IsUser: MUST be 0.

IsSiteGroup: MUST be 1.

UserID: MUST be NULL.

UserSID: MUST be NULL.

UserName: MUST be NULL.

UserLogin: MUST be NULL.

UserEmail: MUST be NULL.

UserNotes: MUST be NULL.

UserSiteAdmin: MUST be NULL.

UserDomainGroup: MUST be NULL.

GroupID: The site group identifier (section 2.2.1.10). MUST match *@PrincipalId##*.

GroupName: The display name of the site group, specified by *@GroupID*.

GroupDescription: The description of the site group specified by *@GroupID*.

GroupOwnerID: The identifier of the user who owns the site group, specified by *@GroupID*.

GroupOwnerIsUser: A bit set to 1 if the user who owns of the site group is a principal; otherwise it MUST be 0.

GroupType: MUST be 0.

3.1.5.81.2 Security Principal Display Information Result Set

If *@PrincipalId##* matches the ID of a site **group**, then the Security Principal Display Information Result Set MUST NOT be returned. If the *@PrincipalId##* matches the ID of a **security principal**, then the Security Principal Display Information Result Set MUST have one row for the matching security principal.

```
        IsUser                          bit,
        IsSiteGroup                     bit,
        UserID                          int,
        UserSID                         varbinary(512),
        UserName                        nvarchar(255),
        UserLogin                       nvarchar(255),
        UserEmail                       nvarchar(255),
        UserNotes                       nvarchar(1023),
        UserSiteAdmin                  bit,
        UserDomainGroup                bit,
        UserFlags                      int,
        GroupID                        int,
        GroupName                      nvarchar(255),
        GroupDescription               nvarchar(512),
        GroupOwnerID                   int,
        GroupOwnerIsUser               bit,
        GroupType                      tinyint;
```

IsUser: MUST be 1.

IsSiteGroup: MUST be 0.

UserID: The user identifier (section 2.2.1.13) of the **principal**. MUST match *@PrincipalId##*.

UserSID: The **SystemID** (section 2.2.1.12) of the principal.

UserName: The display name of the principal.

UserLogin: The login name of the principal.

UserEmail: The email address of the principal.

UserNotes: Description text associated with the principal.

UserSiteAdmin: Set to 1 to indicate that the principal is a site collection administrator, otherwise 0.

UserDomainGroup: Set to 1 to indicate that the principal is a domain group, otherwise 0.

UserFlags: Contains the **UserInfo Flags** (section 2.2.2.11) value describing this principal.

GroupID: MUST be NULL.

GroupName: MUST be NULL.

GroupDescription: MUST be NULL.

GroupOwnerID: MUST be NULL.

GroupOwnerIsUser: MUST be NULL.

GroupType: MUST be NULL.

3.1.5.82 proc_SecGetRoleAssignments

The **proc_SecGetRoleAssignments** stored procedure is invoked to request the **WSS ACE** (section 2.2.4.5) for a specified security scope in a site collection.

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

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the requested security scope.

@ScopeId: The scope identifier (section 2.2.1.8) of the security scope containing the requested **WSS ACE**s.

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

Return Values: The **proc_SecGetRoleAssignments** stored procedure returns an integer return code, which MUST be 0.

The **proc_SecGetRoleAssignments** stored procedure MUST return one result set. It will return zero or more rows.

3.1.5.82.1 WSSACE Result Set

The WSSACE Result Set MUST return one row for each role assignment matching the given security scope within the site collection. Each row represents a **WSS ACE** (section 2.2.4.5).

```
          PrincipalId                    int,
          PermMask                       bigint;
```

PrincipalId: A 4 byte signed integer specifying the user identifier (section 2.2.1.13) of the **principal** for this **WSS ACE**.

PermMask: A **WSS Rights Mask** (section 2.2.2.14) that contains the list of rights that SHOULD be granted to the principal.

3.1.5.83 proc_SecGetRoleBindingsForAllPrincipals

The **proc_SecGetRoleBindingsForAllPrincipals** stored procedure is invoked to list the role assignments for all **principals** in a security scope.

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

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the security scope.

@ScopeId: The scope identifier (section 2.2.1.8) of the security scope containing the role assignments.

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

Return Values: The **proc_SecGetRoleBindingsForAllPrincipals** stored procedure MUST return an integer return code of 0.

The **proc_SecGetRoleBindingsForAllPrincipals** stored procedure MUST return one result set as follows.

3.1.5.83.1 Role Assignment Result Set

The Role Assignment Result Set MUST return one row for each undeleted **principal** in the scope.

```
          RoleId                         int,
          PrincipalId                    int;
```

RoleId: The role identifier of a role associated with the role assignment.

PrincipalId: The user identifier (section 2.2.1.13) of the principal.

3.1.5.84 proc_SecGetRoleDefs

proc_SecGetRoleDefs is invoked to retrieve role definition information for items in the specified site collection and scope.

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

@SiteId: The site collection identifier (section 2.2.1.9) for the desired site collection. This parameter MUST correspond to a valid site collection.

@ScopeId: The scope identifier (section 2.2.1.8) of the desired scope.

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

Return Values: The **proc_SecGetRoleDefs** stored procedure MUST return an integer return code of 0, and MUST return a Role Definition Result Set (section 3.1.5.84.1).

3.1.5.84.1 Role Definition Result Set

The Role Definition Result Set contains a list of role definitions. The Role Definition Result Set MUST contain one row for each role defined in the specified site collection at the specified scope.

```
      RoleId                        int,
      Title                         nvarchar(255),
      Description                   nvarchar(512),
      Hidden                        bit,
      RoleOrder                     int,
      Type                          tinyint,
      WebId                         uniqueidentifier,
      {PermMaskUpper}               int,
      {PermMaskLower}               int,
      WebGroupId                    int;
```

RoleId: The role identifier for the role.

Title: The display name of the role.

Description: A user-defined description of the role.

Hidden: Implementation-specific bit flag used to inform the front-end web server whether or not to display the role definition.

RoleOrder: Specifies the order in which roles MUST be displayed in the front-end web server. Roles MUST be sorted first in ascending RoleOrder, then by descending Type.

Type: The **Role Definition** type (section 2.2.3.16) of the role.

WebId: The site identifier (section 2.2.1.11) for the site within the site collection to which the role is assigned.

{PermMaskUpper}: An integer containing the high-order 32 bits of a **WSS Rights Mask** (section 2.2.2.14) defining the permissions for the role.

{PermMaskLower}: An integer containing the low-order 32 bits of a **WSS Rights Mask** defining the permissions for the role.

WebGroupId: If the role was upgraded from a site **group**, then **WebGroupId** MUST be the identifier of the site group from which it was upgraded, otherwise, it MUST be -1.

3.1.5.85 proc_SecGetSecurityInfo

proc_SecGetSecurityInfo retrieves security permissions information about a document. The document can be specified using its **document identifier** (section 2.2.1.2) or its URL.

```
PROCEDURE proc_SecGetSecurityInfo(
      @SiteId                         uniqueidentifier,
      @WebId                          uniqueidentifier,
      @Url                            nvarchar(260),
      @ThresholdCount                 int,
      @DocId                          uniqueidentifier,
      @RequestGuid                    uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the document.

@WebId: The site identifier (section 2.2.1.11) of the site containing the document. This parameter MUST be ignored.

@Url: The store-relative form URL for the document. The *@Url* parameter MUST be NULL if you wish to specify the document with the *@DocID* parameter.

@ThresholdCount: If *@Url* is a folder and the list does not have more list items than *@ThresholdCount*, then **BigListFolder** in the result set MUST be 0.

@DocId: The document identifier of the document. The *@DocId* parameter MUST be ignored if *@Url* is not NULL.

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

Return Values: The **proc_SecGetSecurityInfo** stored procedure returns an integer return code which MUST be listed in the following table.

Value	Description
0	Successful execution.
3	The Document was not found.

The **proc_SecGetSecurityInfo** stored procedure MUST return a single Security Information Result Set (section 3.1.5.85.1) upon successful execution, or none if the document is not found.

3.1.5.85.1 Security Information Result Set

The Security Information Result Set returns information about the security permissions on the document. The Security Information Result Set MUST be returned only if the document exists, and MUST return one row of information.

```
      ScopeId                         uniqueidentifier,
      ScopeUrl                        nvarchar(260),
      {IsScope}                       bit,
      Acl                             varbinary(max),
      AnonymousPermMask               bigint,
        BigListFolder                 bit;
```

ScopeId: Contains the scope identifier (section 2.2.1.8) of the scope that contains the document.

ScopeUrl: The store-relative form URL of the scope that contains the document.

{IsScope}: A flag indicating that the document is the securable object at the root of the scope. This value MUST be set to 1 if ScopeUrl is the store-relative form URL of the document; otherwise, the value MUST be 0.

Acl: The Access Control List of the scope that contains the document.

AnonymousPermMask: The **WSS Rights Mask** (section 2.2.2.14) in effect for anonymous users in the scope.

BigListFolder: A flag indicating whether or not the list is considered too big for non-administrators. If *@Url* is not a folder, then **BigListFolder** MUST be 0. If the list is not throttled, **BigListFolder** MUST be 0. If the list does not contain more list items than *@ThresholdCount*, then **BigListFolder** MUST be 0. Otherwise, **BigListFolder** MUST be 1.

3.1.5.86 proc_SecGetSiteAdmins

The **proc_SecGetSiteAdmins** stored procedure is invoked to return a list of all site collection administrators not marked as deleted.

```
    PROCEDURE proc_SecGetSiteAdmins(
          @SiteId                       uniqueidentifier,
          @RequestGuid                  uniqueidentifier              OUTPUT
    );
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the site collection administrators.

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

Return Values: The **proc_SecGetSiteAdmins** stored procedure returns an integer return code, which MUST be 0 and it MUST return a single result set

3.1.5.86.1 Site Administrators Result Set

The Site Administrators Result Set returns a list of all site collection administrators not marked as deleted. The Site Administrators Result Set MUST be returned. The Site Administrators Result Set MUST return one row per site collection administrator.

```
        tp_Id                         int,
        tp_SystemID                   varbinary(512),
        tp_Title                      nvarchar(255),
        tp_Login                      nvarchar(255),
        tp_Email                      nvarchar(255),
        tp_Notes                      nvarchar(1023),
        tp_SiteAdmin                  bit,
        tp_DomainGroup                bit,
        tp_Flags                      int;
```

tp_Id: The user identifier (section 2.2.1.13) of the **security principal** who is a site collection administrator.

tp_SystemID: The **SystemID** (section 2.2.1.12) of the security principal.

tp_Title: The name of the security principal, used for display on the front-end web server.

tp_Login: The login name of the security principal, for example, EXAMPLE\username.

tp_Email: The email address of the security principal, for example, "username@mail.example.com".

tp_Notes: Notes associated with the site collection administrator.

tp_SiteAdmin: This value is set to 1 if the security principal is a site collection administrator. This value is set to 0 if the security principal is NOT a site collection administrator.

tp_DomainGroup: This value is set to 1 if the security principal is a domain group. This value is set to 0 if the security principal is NOT a domain group.

tp_Flags: A 4-byte integer bit mask determining the security principal's options as specified in **UserInfo Flags** (section 2.2.2.11).

3.1.5.87 proc_SecGetSiteGroupById

The **proc_SecGetSiteGroupById** stored procedure gets information about a site **group** given its ID.

```
PROCEDURE proc_SecGetSiteGroupById(
      @SiteId                        uniqueidentifier,
      @GroupId                       int,
      @RequestGuid                   uniqueidentifier
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the site group.

@GroupId: The identifier of the site group.

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

Return Values: The **proc_SecGetSiteGroupById** stored procedure MUST return an integer return code of 0.

The **proc_SecGetSiteGroupById** stored procedure MUST return a single result set.

3.1.5.87.1 Site Group Result Set

The Site Group Result Set returns information about a site **group**. The Site Group Result Set MUST contain 1 row of site group information if the **Sec_SiteGroupsView.SiteWebID** equals *@SiteId* and **Sec_SiteGroupsView.ID** equals *@GroupID*; Otherwise, 0 rows MUST be returned.

The Site Group Result Set is defined using T-SQL syntax, in **Sec_SiteGroupsView** (section 2.2.7.6).

3.1.5.88 proc_SecGetSiteGroupByTitle

The **proc_SecGetSiteGroupByTitle** stored procedure is invoked to get site **group** information for the site group with the specified user-friendly name.

```
PROCEDURE proc_SecGetSiteGroupByTitle(
      @SiteId                        uniqueidentifier,
      @Title                         nvarchar(255),
      @RequestGuid                   uniqueidentifier
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the site group.

@Title: The user-friendly name of the site group.

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

Return Values: The **proc_SecGetSiteGroupByTitle** stored procedure MUST return an integer return code of 0 and it MUST return a single result set.

3.1.5.88.1 Site Group Information Result Set

The Site Group Information Result Set returns the available site **group** information for the specified site group. The Site Group Information Result Set MUST contain one row of site group information if the *@SiteID* parameter value and the *@Title* parameter value match an existing site group; otherwise, 0 rows MUST be returned.

The Site Group Information Result Set schema is defined in **Sec_SiteGroupsView** (section 2.2.7.6).

3.1.5.89 proc_SecGetSiteGroupByTitle20

The **proc_SecGetSiteGroupByTitle20** stored procedure provides information about site **groups** in bulk, as specified by a set of up to 20 site group titles.

```
PROCEDURE proc_SecGetSiteGroupByTitle20(
      @SiteId                        uniqueidentifier,
      @PrincipalId01                 nvarchar(255),
      @PrincipalId02                 nvarchar(255),
      @PrincipalId03                 nvarchar(255),
      @PrincipalId04                 nvarchar(255),
      @PrincipalId05                 nvarchar(255),
      @PrincipalId06                 nvarchar(255),
      @PrincipalId07                 nvarchar(255),
      @PrincipalId08                 nvarchar(255),
      @PrincipalId09                 nvarchar(255),
      @PrincipalId10                 nvarchar(255),
      @PrincipalId11                 nvarchar(255),
      @PrincipalId12                 nvarchar(255),
      @PrincipalId13                 nvarchar(255),
      @PrincipalId14                 nvarchar(255),
      @PrincipalId15                 nvarchar(255),
      @PrincipalId16                 nvarchar(255),
      @PrincipalId17                 nvarchar(255),
      @PrincipalId18                 nvarchar(255),
      @PrincipalId19                 nvarchar(255),
      @PrincipalId20                 nvarchar(255),
      @RequestGuid                   uniqueidentifier
);
```

@SiteId: The site collection identifier (section 2.2.1.9) for the site collection containing the specified site groups.

@PrincipalId##: Twenty Site Group Title fields (01 - 20).

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

Return Values: The **proc_SecGetSiteGroupByTitle20** stored procedure MUST return an integer return code of 0.

The **proc_SecGetSiteGroupByTitle20** stored procedure MUST return one result set for each NON-NULL *@PrincipalId##*.

3.1.5.89.1 Site Group Information Result Set

The Site Group Information Result Set returns once per each NON-NULL *@PrincipalId##*, and the Site Group Information Result Set MUST contain one row of site **group** information if the *@SiteID* parameter value and the *@PrincipalId##* parameter value match an existing site group; otherwise, zero rows MUST be returned.

The Site Group Information Result Set is defined using T-SQL syntax, as defined in **Sec_SiteGroupsView** (section 2.2.7.6).

3.1.5.90 proc_SecGetUserAccountDirectoryPath

The **proc_SecGetUserAccountDirectoryPath** stored procedure is invoked to return the **user account directory path** of a specified site collection.

This stored procedure is called when a user has specific user restrictions and the user account directory path is required along other user information.

```
PROCEDURE proc_SecGetUserAccountDirectoryPath(
      @SiteId                            uniqueidentifier,
      @RequestGuid                       uniqueidentifier          OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection.

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

Return Code Values: The **proc_SecGetUserAccountDirectoryPath** stored procedure returns an integer return code, which MUST be 0.

The **proc_SecGetUserAccountDirectoryPath** stored procedure MUST return one result set.

3.1.5.90.1 User Account Directory Path Result Set

The user Account Directory Path Result Set returns the user Account Directory Path for the site collection. The user Account Directory Path Result Set MUST contain one row when the specified site collection exists, and it MUST contain zero rows when the specified site collection does not exist.

```
UserAccountDirectoryPath          nvarchar(512);
```

UserAccountDirectoryPath: Contains the user Account Directory Path for the site collection.

3.1.5.91 proc_SecGetUserPermissionOnGroup

The **proc_SecGetUserPermissionOnGroup** stored procedure is invoked to determine what permissions the requesting user has within a specified site **group**.

```
PROCEDURE proc_SecGetUserPermissionOnGroup(
      @SiteId                          uniqueidentifier,
      @GroupId                         int,
      @UserId                          int,
      @BelongsToGroup                  bit,
      @CanViewMembership               bit                       OUTPUT,
      @CanEditMembership               bit                       OUTPUT,
      @GroupOwnerId                    int                       OUTPUT,
      @IsExplicitlyInMembership        bit                       OUTPUT,
      @RequestGuid                     uniqueidentifier          OUTPUT
```

```
    );
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the site group specified by *@GroupId*.

@GroupId: The site group identifier (section 2.2.1.10) of the site group whose permissions are to be returned.

@UserId: The user identifier (section 2.2.1.13) of the current user making the request for permissions.

@BelongsToGroup: When this parameter is set to 1, it indicates that the user specified by *@UserId* belongs to the site group specified by *@GroupId*. This is used to determine access to site group information, as site groups can restrict membership information to only members of the site group.

@CanViewMembership: When this parameter is set to 1, it indicates that the user specified by *@UserId* is permitted to view site group membership.

@CanEditMembership: When this parameter is set to 1, it indicates that the user specified by *@UserId* is permitted to edit site group membership.

@GroupOwnerId: The user identifier of the user who owns the site group specified by *@GroupId*.

@IsExplicitlyInMembership: When this parameter is set to 1, it indicates that the user specified by *@UserId* is directly a **member** of the site group specified by *@GroupId* and not conferred membership through an intermediate site group membership.

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

Return Values: The **proc_SecGetUserPermissionOnGroup** stored procedure returns an integer return code, which MUST be listed in the following table.

Value	Description
0	Successful execution.
1319	The site group was not found for the provided *@GroupId* and *@SiteId*.

The **proc_SecGetUserPermissionOnGroup** stored procedure MUST return zero result sets.

3.1.5.92 proc_SecGetWebsAndListWithUniquePermissions

The **proc_SecGetWebsAndListsWithUniquePermissions** is invoked to return a list of sites and lists that have unique permissions.

```
    PROCEDURE proc_SecGetWebsAndListsWithUniquePermissions(
              @SiteId                    uniqueidentifier,
              @WebId                     uniqueidentifier,
              @WebUrl                    nvarchar(260),
              @RequestGuid               uniqueidentifier          OUTPUT
    );
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection.

@WebId: The site identifier (section 2.2.1.11) of the site to query.

@WebUrl: The URL of the of the site collection.

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

Return Values: The **proc_SecGetWebsAndListsWithUniquePermissions** stored procedure returns an integer return code which MUST be 0. The **proc_SecGetWebsAndListsWithUniquePermissions** MUST return 4 result sets.

3.1.5.92.1 Site Information Result Set

The Site Information Result Set MUST return one row if the site has unique permissions. The Site Information Result Set MUST return 0 rows if the site has no unique permissions.

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

FullUrl: The URL of the site specified by *@WebId*.

Id: The site identifier (section 2.2.1.11) of the site specified by *@WebId*.

Title: The title associated with the site specified by *@WebId*.

3.1.5.92.2 Sub Site Information Result Set

The Sub Site Information Result Set returns information about each subsite of the site specified by *@WebId*. The result set MUST return 0 or more rows. There MUST be one row per subsite that has unique permissions.

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

FullUrl: The URL of the subsite.

Id: The site identifier (section 2.2.1.11) of the subsite.

Title: The title associated with the subsite.

3.1.5.92.3 Site List Information Result Set

The Site List Information Result Set returns information about lists in the site specified by *@WebId*. The number of rows returned MUST be the same as the number of lists which either have unique permissions or contain documents that have unique permissions.

```
FullUrl                      nvarchar(256),
Id                           uniqueidentifier,
Title                        nvarchar(255),
ListRootFolderUrl            nvarchar(260),
tp_Id                        uniqueidentifier,
tp_Title                     nvarchar(25),
HasUniquePerm                bit;
```

FullUrl: The URL of the site specified by *@WebId*.

Id: The site identifier (section 2.2.1.11) of the site specified by *@WebId*.

Title: The title associated with the site specified by *@WebId*.

ListRootFolderUrl: Root folder of the list which has unique permissions in the site specified by *@WebId*.

tp_Id: The list identifier (section 2.2.1.5) of the list.

tp_Title: The list title.

HasUniquePerm: This parameter is set to 0 if the list permissions is as the same as the site. This parameter is set to 1 if the list permissions is not as the same as the site.

3.1.5.92.4 Sub Site List Information Result Set

The Sub Site List Information Result Set returns information about lists in the subsite that is in the site specified by the *@WebId*.

There MUST be a row for each list in the subsite of the site specified by *@WebId* which either have unique permissions or contain documents that have unique permissions.

```
FullUrl                      nvarchar(256),
Id                           uniqueidentifier,
Title                        nvarchar(255),
ListRootFolderUrl            nvarchar(260),
tp_Id                        uniqueidentifier,
tp_Title                     nvarchar(25),
HasUniquePerm                bit;
```

FullUrl: The URL of the site specified by the *@WebId*.

Id: The site identifier (section 2.2.1.11) of the site specified by the *@WebId*.

Title: The title associated with the site.

ListRootFolderUrl: Root folder of the list which has unique permissions in the site.

tp_Id: The list identifier (section 2.2.1.5) of the list.

tp_Title: The list title.

HasUniquePerm: This parameter is set to 0 if the list permissions is as the same as the site. This parameter is set to 1 if the list permissions is not as the same as the site.

3.1.5.93 proc_SecListAllSiteMembers

The **proc_SecListAllSiteMembers** stored procedure provides information about all non-deleted **security principals** in the specified site collection.

```
PROCEDURE proc_SecListAllSiteMembers (
      @SiteID                     uniqueidentifier,
      @RequestGuid                uniqueidentifier          OUTPUT
);
```

@SiteID: The site collection identifier (section 2.2.1.9) of the site collection containing the requested security principal information.

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

Return Values: The **proc_SecListAllSiteMembers** stored procedure returns an integer return code, which MUST be 0 and it MUST return a single result set.

3.1.5.93.1 User Information Result Set

The user Information Result Set returns information about the non-deleted users directly associated with the site collection specified by *@SiteID*. Zero or more rows MUST be returned. There MUST be one row per non-deleted user in the site collection specified by *@SiteId*.

```
          tp_Id                         int,
          tp_SystemID                   varbinary(512),
          tp_Title                      nvarchar(255),
          tp_Login                      nvarchar(255),
          tp_Email                      nvarchar(255),
          tp_Notes                      nvarchar(1023),
          tp_SiteAdmin                  bit,
          tp_DomainGroup                bit,
          tp_Flags                      int;
```

tp_Id: The user identifier (section 2.2.1.13) that uniquely identifies the security principal with the site collection.

tp_SystemID: The **SystemID** (section 2.2.1.12) of the security principal.

tp_Title: The display name of the security principal.

tp_Login: The login name of the security principal.

tp_Email: The email address of the security principal, for example, "username@mail.example.com".

tp_Notes: Notes associated with the security principal.

tp_SiteAdmin: This value is set to 1 if the security principal is a site collection administrator. This value is set to 0 if the security principal is NOT a site collection administrator.

tp_DomainGroup: This value is set to 1 if the security principal is a domain group. This value is set to 0 if the security principal is NOT a domain group.

tp_Flags: A 4-byte integer bit mask specifying the security principal's options as specified in **UserInfo Flags** (section 2.2.2.11).

3.1.5.94 proc_SecListAllWebMembers

The **proc_SecListAllWebMembers** stored procedure is invoked to list all non-deleted user and domain group accounts registered with a specified site.

```
    PROCEDURE proc_SecListAllWebMembers(
          @SiteId                       uniqueidentifier,
          @WebId                        uniqueidentifier,
          @RequestGuid                  uniqueidentifier          OUTPUT
    );
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the site specified by *@WebId*.

@WebId: The site identifier (section 2.2.1.11) of the site to query for members.

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

Return Values: The **proc_SecListAllWebMembers** stored procedure returns an integer return code which MUST be 0 and it MUST return a single result set.

3.1.5.94.1 Site Membership Result Set

The Site Membership Result Set returns a list of all non-deleted users or domain groups registered with the specified site. The Site Membership Result Set MUST return zero rows if the site specified by *@WebId* has no members, otherwise it MUST contain one row for each **member** if *@WebId* matches an existing site within the site collection specified by *@SiteId*.

```
        tp_Id                          int,
        tp_SystemID                    varbinary(512),
        tp_Title                       nvarchar(255),
        tp_Login                       nvarchar(255),
        tp_Email                       nvarchar(255),
        tp_Notes                       nvarchar(1023),
        tp_SiteAdmin                   bit,
        tp_DomainGroup                 bit,
        tp_Flags                       int;
```

tp_ID: The user identifier (section 2.2.1.13) of the **security principal**.

tp_SystemID: The **SystemID** (section 2.2.1.12) of the security principal.

tp_Title: The display name of the security principal.

tp_Login: The login name of the security principal.

tp_Email: The email address of the security principal.

tp_Notes: A descriptive text associated with the security principal.

tp_SiteAdmin: Contains a bit flag set to 1 if the security principal is a site collection administrator. This value is set to 0 if the user is NOT a site collection administrator.

tp_DomainGroup: Contains a bit flag set to 1 if the security principal is a domain group. This value is set to 0 if the security principal is NOT for a domain group.

tp_Flags: A 4-byte integer bit mask determining the security principal's options as specified in **UserInfo Flags** (section 2.2.2.11).

3.1.5.95 proc_SecListGroupsInRole

The **proc_SecListGroupsInRole** stored procedure is invoked to list all site **groups** that have the specified role.

```
    PROCEDURE proc_SecListGroupsInRole(
          @SiteId                      uniqueidentifier,
          @ScopeId                     uniqueidentifier,
          @RoleId                      int,
          @RequestGuid                 uniqueidentifier
    );
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection that contains the site groups that have the role.

@ScopeId: The scope identifier (section 2.2.1.8) of the scope containing the role.

@RoleId: Specifies the role identifier for the role.

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

Return Values: The **proc_SecListGroupsInRole** stored procedure returns an integer return code which MUST be 0.

The **proc_SecListGroupsInRole** stored procedure MUST return 1 result set.

3.1.5.95.1 Site Group Information Result Set

The Site Group Information Result Set returns the collection of site **groups** that have the specified role. The Site Group Information Result Set MUST be empty if either the site collection, role, or scope are not found, or if no site groups have the specified role in the specified scope. Otherwise, the Site Group Information Result Set MUST have one row per site group.

The Site Group Information Result Set schema is defined in **Sec_SiteGroupsView** (section 2.2.7.6).

3.1.5.96 proc_SecListScopeGroups

The **proc_SecListScopeGroups** stored procedure is invoked to list all site **groups** that have role assignments in a specified scope.

```
PROCEDURE proc_SecListScopeGroups(
      @SiteId                       uniqueidentifier,
      @ScopeId                      uniqueidentifier,
      @RequestGuid                  uniqueidentifier
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection which contains the scope whose site groups with role assignments are being listed.

@ScopeId: The scope identifier (section 2.2.1.8) of the specified scope.

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

Return Values: The **proc_SecListScopeGroups** stored procedure returns an integer return code which MUST be 0.

The **proc_SecListScopeGroups** stored procedure MUST return a single result set as defined in the following section.

3.1.5.96.1 Site Groups Result Set

The Site Groups Result Set contains all site **group** information for site groups with a role assignment in the specified scope. The Site Groups Result Set MUST be empty if either the site collection or scope are not found, or if no site groups have role assignments in the specified scope. Otherwise, one row MUST be returned for each site group in the site collection which has a role assignment in the specified scope.

The Site Groups Result Set is the same as the view defined in **Sec_SiteGroupsView** (section 2.2.7.6).

3.1.5.97 proc_SecListScopeUsers

The **proc_SecListScopeUsers** stored procedure is invoked to list information about users and domain groups with a direct role assignment association with the specified scope, rather than by membership in a site **group** with a role assignment association with the scope.

```
PROCEDURE proc_SecListScopeUsers(
      @SiteId                       uniqueidentifier,
      @ScopeId                      uniqueidentifier,
      @RequestGuid                      uniqueidentifier
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the scope.

@ScopeId: The scope identifier (section 2.2.1.8) of the scope for which information about users with role assignments is requested.

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

Return Values: The **proc_SecListScopeUsers** stored procedure MUST return an integer return code of 0, and it MUST return a single result set with 0 or more rows.

3.1.5.97.1 User Information Result Set

The user Information Result Set returns information about the users and domain groups directly associated with the specified scope through a role assignment. The user Information Result Set MUST contain 1 row for each non-deleted user or domain group directly associated with the specified scope through a role assignment. The user Information Result Set MUST contain 0 rows if no users or domain groups are directly associated with the specified scope, or if the parameters *@SiteId* or *@ScopeId* do not match existing site collection or scope identifiers (section 2.2.1.8), respectively.

The user Information Result Set is defined in Principal user Information Result Set, section 2.2.5.17.

3.1.5.98 proc_SecListSiteGroupMembership

The **proc_SecListSiteGroupMembership** stored procedure lists members in a specified site **group**. To view membership, the site group MUST be set to allow everyone to view membership, or the current user MUST belong to the site group, be a site auditor, or be the site owner.

```
  PROCEDURE proc_SecListSiteGroupMembership (
        @SiteId                           uniqueidentifier,
        @GroupId                          int,
        @CurrentUserId                    int,
        @SiteAuditor                      bit,
        @BelongsToGroup                   bit,
        @GroupOwnerId                     int,
        @CurrentUserIsOwner               bit,
        @RequestGuid                      uniqueidentifier
  );
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the site group.

@GroupId: The identifier of the site group.

@CurrentUserId: The identifier of the current user. This is used to determine privileges.

@SiteAuditor: A bit set to 1 if the current user is a site auditor. This is used to determine privileges.

@BelongsToGroup: A bit set to 1 if the current user is a **member** of the site group. This is used to determine privileges.

@GroupOwnerId: The identifier of the owner of the site group. This is used to determine privileges.

@CurrentUserIsOwner: A bit set to 1 if the current user is the owner of the site group. This is used to determine privileges.

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

Return Values: The **proc_SecListSiteGroupMembership** stored procedure returns an integer return code which MUST be listed in the following table.

Value	Description
0	Successful execution.
5	The user does not have privileges to view site group membership.

The **proc_SecListSiteGroupMembership** stored procedure MUST return a single result set.

3.1.5.98.1 User Information Result Set

The user Information Result Set MUST either be empty or include 1 row of information for each **member** of the site **group**. If the user specified by *@CurrentUserId* does not have privileges to view membership, then the user Information Result Set will be empty. To view membership, the site group MUST be set to allow everyone to view membership, or the current user MUST belong to the site group, be a site auditor, or be the site owner.

The user Information Result Set is defined in Principal user Information Result Set, section 2.2.5.17.

3.1.5.99 proc_SecListSiteGroups

The **proc_SecListSiteGroups** stored procedure is invoked to list site **group** information for a specified site collection.

```
PROCEDURE proc_SecListSiteGroups(
      @SiteId                        uniqueidentifier,
      @RequestGuid                   uniqueidentifier
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the site groups to list.

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

Return Values: The **proc_SecListSiteGroups** stored procedure returns an integer return code, which MUST be 0, and MUST return one result set.

3.1.5.99.1 Site Group Information Result Set

The site **group** information result set contains all site group information for the site collection. The site group information result set MUST be empty if the site collection is not found. Otherwise, 1 row MUST be returned for each site group in the specified site collection.

The site group information result set is defined using T-SQL syntax, in Sec_SiteGroupsView (section 2.2.7.6).

3.1.5.100 proc_SecListSiteGroupsContainingUser

The **proc_SecListSiteGroupsContainingUser** stored procedure lists the site **groups** in which the user is a **member**.

```
PROCEDURE proc_SecListSiteGroupsContainingUser(
      @SiteId                        uniqueidentifier,
      @UserId                        int,
      @RequestGuid                   uniqueidentifier
```

```
    );
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the site groups to be listed. This parameter MUST correspond to a valid site collection, and MUST NOT be NULL.

@UserId: The user identifier (section 2.2.1.13) of the current user. This is used to determine site groups containing this user.

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

Return Values: The **proc_SecListSiteGroupsContainingUser** stored procedure returns an integer return code which MUST be 0. The **proc_SecListSiteGroupsContainingUser** stored procedure MUST return one result set.

3.1.5.100.1 Site Group Information Result Set

The Site Group Information Result Set contains information on site **groups** where the user is a **member**. The Site Group Information Result Set MUST be returned, and MUST contain 1 row for each site group found for the user. The Site Group Information Result Set schema is defined in Sec_SiteGroupsView (section 2.2.7.6).

3.1.5.101 proc_SecListSiteGroupsWhichUserOwns

The **proc_SecListSiteGroupsWhichUserOwns** stored procedure returns site **group** information for the site groups owned by the specified **principal**.

```
    PROCEDURE proc_SecListSiteGroupsWhichUserOwns(
          @SiteId                        uniqueidentifier,
          @UserId                        int,
          @RequestGuid                   uniqueidentifier
    );
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the principal and the site groups to be listed.

@UserId: The user identifier (section 2.2.1.13) of the principal, used to find the site groups it owns.

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

Return Values: The **proc_SecListSiteGroupsWhichUserOwns** stored procedure returns an integer return code which MUST be 0, and MUST return one result set.

3.1.5.101.1 Site Group Information Result Set

The Site Group Information Result Set returns site **group** information for all site groups that are owned either directly by the user or external group specified by *@UserId*, or for site groups which are owned indirectly, when the user or external group is a **member** of a site group which is the returned site group's owner.

The Site Group Information Result Set MUST contain one row for each site group owned. The T-SQL syntax for the Site Group Information Result Set is defined in Sec_SiteGroupsView (section 2.2.7.6).

3.1.5.102 proc_SecListUsersInRole

The **proc_SecListUsersInRole** stored procedure lists the non-deleted **principals** assigned to a specified role in the specified scope.

```
PROCEDURE proc_SecListUsersInRole(
      @SiteId                         uniqueidentifier,
      @ScopeId                        uniqueidentifier,
      @RoleId                         int,
      @RequestGuid                    uniqueidentifier
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the scope, role, and principals to be listed.

@ScopeId: The scope identifier (section 2.2.1.8) of the scope within which to match principals assigned to the role.

@RoleId: Specifies the role identifier of the role definition assigned to the principals to be listed.

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

Return Values: The **proc_SecListUsersInRole** stored procedure returns an integer return code which MUST be 0, and MUST return one result set.

3.1.5.102.1 User Role Membership Result Set

The user Role Membership Result Set returns information about the non-deleted **principals** assigned to the specified role in the specified scope and site collection. The user Role Membership Result Set MUST be empty if no principals assigned to the role are found; otherwise, it MUST include one row of information for each matching principal.

```
      tp_Id                           int,
      tp_SystemID                     varbinary(512),
      tp_Title                        nvarchar(255),
      tp_Login                        nvarchar(255),
      tp_Email                        nvarchar(255),
      tp_Notes                        nvarchar(1023),
      tp_SiteAdmin                    bit,
      tp_DomainGroup                  bit,
      tp_Flags                        int;
```

tp_Id: The user identifier (section 2.2.1.13) of the principal.

tp_SystemID: The **SystemID** (section 2.2.1.12) of the principal.

tp_Title: The display name of the principal.

tp_Login: The login name of the principal.

tp_Email: The email address of the principal.

tp_Notes: Contains notes about the principal.

tp_SiteAdmin: If the parameter is set to 1, then the principal is a site collection administrator. If the parameter is set to 0, then the principal is not a site collection administrator.

tp_DomainGroup: If the parameter is set to 1, the principal is a domain group. If the parameter is set to 0, the principal is not a domain group.

tp_Flags: The **UserInfo Flags** (section 2.2.2.11) value of the principal.

3.1.5.103 proc_SecMigrateUser

The **proc_SecMigrateUser** stored procedure updates the **SystemID** (section 2.2.1.12) and login name for a **principal** in the **UserInfo** (section 2.2.7.10) and **AllUserData** (section 2.2.7.3) tables. If an existing principal is found with the new login name and **SystemID**, then **proc_SecMigrateUser** MUST update that principal with a new unique **SystemID** and mark that principal as deleted in the **UserInfo** table (section 2.2.7.10) before updating the specified principal. If an existing principal is found with the new login name and **SystemID**, then **proc_SecMigrateUser** MUST also mark that principal as deleted in the **AllUserData** table's user list.

```
PROCEDURE proc_SecMigrateUser(
      @SiteId                        uniqueidentifier,
      @OldLogin                      nvarchar(255),
      @NewLogin                      nvarchar(255),
      @NewSystemId                   varbinary(512),
      @IsWindowsAuth                 bit,
      @PeopleListId                  uniqueidentifier,
      @LoginColumnName               nvarchar(64),
      @DeletedColumnName             nvarchar(64),
      @OldSystemId                   varbinary(512) = NULL,
      @RequestGuid                   uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the principal to be updated.

@OldLogin: The current value of the principal's login name.

@NewLogin: The new value to set as the principal's login name. This parameter MUST NOT be NULL.

@NewSystemId: The new **SystemID** to set for the principal. This parameter MUST NOT be NULL.

@IsWindowsAuth: A bit specifying whether the principal is authenticated with Windows-integrated authentication. If this parameter is set to 1 or NULL, then the principal is authenticated using Windows-integrated authentication.

@PeopleListId: The list identifier (section 2.2.1.5) of the user list.

@LoginColumnName: The name of the column in the **AllUserData** table that stores the user list's login names.

@DeletedColumnName: The name of the column in the **AllUserData** table that marks whether a principal in the user list has been deleted.

@OldSystemId: The current value of the principal's **SystemID**. This parameter can be NULL. If the *@OldSystemId* parameter is not NULL, then the principal specified by *@OldLogin* MUST also have the **SystemID** specified by *@OldSystemId*.

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

Return Values: The **proc_SecMigrateUser** stored procedure returns an integer return code which MUST be listed in the following table.

Value	Description
0	Successful execution.
1003	An existing principal was found with the new login name and **SystemID**, but **proc_SecMigrateUser** failed to update it with a new **SystemID** and mark it as deleted. Failed to find or update the specified principal.

The **proc_SecMigrateUser** stored procedure MUST return no result sets.

3.1.5.104 proc_SecReCalculateWebFGP

The **proc_SecReCalculateWebFGP** stored procedure is invoked to update the bit at 0x00000400 of the **Site Property Flags** (section 2.2.2.10) to indicate whether the site has at least one uniquely secured object within it. This bit can be used by a search tool to determine whether it SHOULD avoid indexing pages in this site so as not to reveal secure information.

```
PROCEDURE proc_SecReCalculateWebFGP(
      @SiteId                       uniqueidentifier,
      @WebId                        uniqueidentifier,
      @RequestGuid                  uniqueidentifier
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the site to be updated. This parameter MUST correspond to a valid site collection, and MUST NOT be NULL.

@WebId: The site identifier (section 2.2.1.11) of the site containing the flag to be updated.

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

Return Values: The **proc_SecReCalculateWebFGP** stored procedure returns an integer return code, which MUST be 0, and it MUST return zero result sets.

3.1.5.105 proc_SecRefreshToken

The **proc_SecRefreshToken** stored procedure is invoked to update the **UserInfo** table (section 2.2.7.10) with information about a specified user's membership in external **groups**.

```
PROCEDURE proc_SecRefreshToken(
      @SiteId                       uniqueidentifier,
      @UserId                       int,
      @ExternalToken                varbinary(max),
      @ExternalTokenTime            datetime,
      @RequestGuid                  uniqueidentifier
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the user to update in the **UserInfo** table.

@UserId: The user identifier (section 2.2.1.13) of the user.

@ExternalToken: An **External Group Token** (section 2.2.4.2) containing external group membership information for the user. This value can be NULL, specifying that the user is not a **member** of any external groups. **proc_SecRefreshToken** MUST update the **UserInfo** table row matching the user identifier with this value in the **tp_ExternalToken** field.

@ExternalTokenTime: A **datetime** in **UTC** specifying when the **External Group Token** in the *@ExternalToken* parameter was last updated. This parameter MUST be NULL if the *@ExternalToken* parameter is NULL. **proc_SecRefreshToken** MUST update the **UserInfo** table row matching the user identifier with this value in the **tp_ExternalTokenLastUpdated** field.

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

Return Values: The **proc_SecRefreshToken** stored procedure returns an integer return code , which MUST be 0, and it MUST NOT return a result set.

3.1.5.106 proc_SecRemoveGroup

The **proc_SecRemoveGroup** stored procedure is invoked to remove a site **group** from a site collection.

```
PROCEDURE proc_SecRemoveGroup(
      @SiteId                        uniqueidentifier,
      @GroupId                       int,
      @UserID                        int,
      @SiteAdmin                     bit,
      @GroupOwnerId                  int,
      @CurrentUserIsOwner            bit,
      @ReturnDLAlias                 bit,
      @CheckIfInUse                  bit,
      @RequestGuid                   uniqueidentifier
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the site group to be removed.

@GroupId: The site group identifier (section 2.2.1.10) for the site group to be removed.

@UserID: The user identifier (section 2.2.1.13) for the current user. The ownership of any site group previously owned by the deleted site group MUST be re-assigned to this user identifier.

@SiteAdmin: A bit flag specifying whether the current user is a site collection administrator. If this flag is set to 1, then the site group MUST be removed. If this flag is set to 0, then the site group MUST NOT be removed unless *@CurrentUserIsOwner* specifies it.

@GroupOwnerId: The user identifier or site group identifier of the site group owner. If *@CurrentUserIsOwner* is set to 1, then this value MUST be set to the user identifier of the current user, or a site group which includes the current user. If *@CurrentUserIsOwner* is set to 0, then this value MUST be ignored.

@CurrentUserIsOwner: A bit flag specifying that the current user is an owner of the site group. If *@CurrentUserIsOwner* is set to 1 and *@GroupOwnerId* contains the user identifier of the owner of the site group, or the site group identifier of a site group, which includes the current user, then the site group MUST be removed. If *@CurrentUserIsOwner* is set to 0, then the site group MUST NOT be removed unless *@SiteAdmin* specifies it.

@ReturnDLAlias: A bit flag specifying whether to return the DLAlias Result Set. If this flag is set to 1, the result set MUST be returned. If this flag is set to 0, then the result MUST NOT be returned.

@CheckIfInUse: A bit flag specifying whether to check if the site group is in use before removing it. A site group is in use if it is assigned to any role in the site collection. If the site group is not is use, it is removed. If the site group is in use, then it MUST NOT be removed, but the stored procedure MUST return an empty DLAlias Result Set if *@ReturnDLAlias* is 1 and a Return Value of 0.

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

Return Values: The **proc_SecRemoveGroup** stored procedure returns an integer return code, which MUST be listed in the following table.

Value	Description
0	Successful execution.
5	The current user has insufficient permissions to remove the site group.

The **proc_SecRemoveGroup** stored procedure returns either zero or one result set. If the site group is successfully removed and @ReturnDLAlias is set to 1, then **proc_SecRemoveGroup** MUST return a result set. If the current user does not have sufficient permissions to remove the site group or @ReturnDLAlias is set to 0, then **proc_SecRemoveGroup** MUST NOT return a result set.

3.1.5.106.1 DLAlias Result Set

If the site **group** existed and was removed successfully and if @ReturnDLAlias is set to 1, the DLAlias Result Set MUST return the email distribution address of the site group in a single row. The DLAlias Result Set MUST be empty if the email distribution address was blank.

```
      DLAlias                        varchar(128);
```

DLAlias: The email distribution address for the site group.

3.1.5.107 proc_SecMarkGroupForWebDelete

The **proc_SecMarkGroupForWebDelete** stored procedure is invoked to mark a site **group** for later removal from a site collection.

```
PROCEDURE proc_SecMarkGroupForWebDelete(
      @SiteId                        uniqueidentifier,
      @GroupId                       int,
      @UserID                        int,
      @SiteAdmin                     bit,
      @GroupOwnerId                  int,
      @CurrentUserIsOwner            bit,
      @DeletionWebId                 uniqueidentifier,
      @RequestGuid                   uniqueidentifier
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the site group to be marked for removal.

@GroupId: The site group identifier (section 2.2.1.10) for the site group to be marked for removal.

@UserID: The user identifier (section 2.2.1.13) for the current user.

@SiteAdmin: A bit flag specifying whether the current user is a site collection administrator. If this flag is set to 1, then the site group MUST be marked for removal. If this flag is set to 0, then the site group MUST NOT be marked for removal unless *@CurrentUserIsOwner* specifies it.

@GroupOwnerId: The user identifier or site group identifier of the site group owner. If *@CurrentUserIsOwner* is set to 1, then this value MUST be set to the user identifier of the current user, or a site group that includes the current user. If *@CurrentUserIsOwner* is set to 0, then this value MUST be ignored.

@CurrentUserIsOwner: A bit flag specifying that the current user is an owner of the site group. If *@CurrentUserIsOwner* is set to 1 and *@GroupOwnerId* contains the user identifier of the owner of the site group, or the site group identifier of a site group that includes the current user, then the site group MUST be marked for removal. If *@CurrentUserIsOwner* is set to 0, then the site group MUST NOT be marked for removal unless *@SiteAdmin* specifies it.

@DeletionWebId: The site identifier (section 2.2.1.11) of the site associated with the group.

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

Return Values: The **proc_SecMarkGroupForWebDelete** stored procedure returns an integer return code, which MUST be listed in the following table.

Value	Description
0	Successful execution.
5	The current user has insufficient permissions to mark the site group for removal.

The **proc_SecMarkGroupForWebDelete** stored procedure returns either zero or one result set. If the site group is successfully marked for removal, then **proc_SecMarkGroupForWebDelete** MUST return a result set. If the current user does not have sufficient permissions to mark the site group for removal, then **proc_SecMarkGroupForWebDelete** MUST NOT return a result set.

3.1.5.107.1 DLAlias Result Set

If the site **group** existed and was marked for removal successfully, the DLAlias Result Set MUST return the email distribution address of the site group in a single row. The DLAlias Result Set MUST be empty if the email distribution address was blank.

```
        DLAlias                  varchar(128);
```

DLAlias: The email distribution address for the site group.

3.1.5.108 proc_SecRemovePrincipalFromScope

The proc_SecRemovePrincipalFromScope stored procedure removes a **principal** from a **security role** associated with a site in a specified security scope. If no role is specified, or if the role specified is the "Guest" Role, then the principal MUST be removed from all roles in the scope, and it MUST also be removed from all roles in other specified scopes or subsite contained within the Site.

```
PROCEDURE proc_SecRemovePrincipalFromScope (
      @SiteId                        uniqueidentifier,
      @WebId                         uniqueidentifier,
      @WebScopeId                    uniqueidentifier,
      @ScopeId                       uniqueidentifier,
      @RemoveFromCurrentScopeOnly    bit,
      @PrincipalId                   int,
      @RoleId                        int = NULL,
      @RequestGuid                   uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the principal and scope.

@WebId: The site identifier (section 2.2.1.11) of the site containing the principal and scope.

@WebScopeId: The scope identifier (section 2.2.1.8) of the scope which includes the site. If the site has unique permissions, then this MUST be the scope of the site. If *@RoleId* is not NULL and not the role identifier of the Guest Role (1073741825), then *@WebScopeId* MUST be ignored. Otherwise, *@WebScopeId* determines additional scopes or sites from which the principal MUST be removed, as follows.

Value	Description
@WebScopeId =	Remove the principal from the specified role or roles within the site and within all

Value	Description
@ScopeId	of the site's subsites which share the same scope.
@WebScopeId ≠ @ScopeId	Remove the principal from the specified role or roles in all scopes contained within the site.

@ScopeId: The scope identifier of the scope in which to remove the principal from the role. See *@WebScopeId* for behavior based on comparison of the *@ScopeId* and *@WebScopeId* parameter values.

@RemoveFromCurrentScopeOnly: The bit flag indicating from which scope the principal will be removed. If this flag is set to 1, then **proc_SecRemovePrincipalFromScope** MUST remove the principal from the current scope only.

@PrincipalId: The user identifier (section 2.2.1.13) of the principal to remove.

@RoleId: The role identifier of the role to remove the principal from, within the specified scopes. The value in *@RoleId* MUST either correspond to a valid role or be NULL. If this value is NULL, or if *@RoleID* is the role identifier of the Guest Role (1073741825), then **proc_SecRemovePrincipalFromScope** MUST remove the principal from all roles in the specified scopes.

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

Return Values: The **proc_SecRemovePrincipalFromScope** stored procedure returns an integer return code, which MUST be listed in the following table.

Value	Description
0	Successful execution.
3	The scope identifier points to a scope whose path cannot be found.

The **proc_SecRemovePrincipalFromScope** stored procedure MUST return a single result set as follows.

3.1.5.108.1 Site Audit Mask Result Set

The Site Audit Mask Result Set contains the information about the **Audit Flags** (section 2.2.2.1) associated with the specified site. The Site Audit Mask Result Set MUST be returned and MUST contain a single row. The Site Audit Mask Result Set is defined in Site Audit Mask Result Set, section 2.2.5.20.

3.1.5.109 proc_SecRemoveRoleDef

The **proc_SecRemoveRoleDef** stored procedure removes a role definition, and any role assignments which use that role definition, from a specified site.

```
PROCEDURE proc_SecRemoveRoleDef(
      @SiteId                        uniqueidentifier,
      @WebId                         uniqueidentifier,
      @RoleId                        int,
      @RequestGuid                   uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the role definition to be removed.

@WebId: The site identifier (section 2.2.1.11) of the site associated with the role definition to be removed.

@RoleId: The identifier of the role definition to be removed.

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

Return Values: The **proc_SecRemoveRoleDef** stored procedure MUST return an integer return code of 0. The **proc_SecRemoveRoleDef** stored procedure MUST return the following result set.

3.1.5.109.1 Site Audit Mask Result Set

The Site Audit Mask Result Set contains information about the **Audit Flags** (section 2.2.2.1) associated with the site. The Site Audit Mask Result Set MUST be returned and MUST contain one row. The Site Audit Mask Result Set is defined in Site Audit Mask Result Set (section 2.2.5.20).

3.1.5.110 proc_SecRemoveUserFromScopeByLogin

The **proc_SecRemoveUserFromScopeByLogin** stored procedure is invoked to remove a user, specified by login name, from a role in a specified security scope. If no role is specified, or if the role specified is the "Guest" Role, then the user will be removed from all roles in the specified scope and will also be removed from other scopes associated with the specified site.

```
PROCEDURE proc_SecRemoveUserFromScopeByLogin(
      @SiteId                       uniqueidentifier,
      @WebId                        uniqueidentifier,
      @WebScopeId                   uniqueidentifier,
      @ScopeId                      uniqueidentifier,
      @RemoveFromCurrentScopeOnly   bit,
      @LoginName                    nvarchar(255),
      @RoleId                       int                 = NULL,
      @RequestGuid                  uniqueidentifier    = NULL OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the user and scope.

@WebId: The site identifier (section 2.2.1.11) of the site using the role.

@WebScopeId: The scope identifier (section 2.2.1.8) of the scope which includes the site specified by *@WebId*. If *@RoleId* is not NULL and is not the role identifier of the "Guest" Role (1073741825), then *@WebScopeId* MUST be ignored. Otherwise, *@WebScopeId* determines which additional scopes the user MUST be removed from, as follows:

- If *@WebScopeId* is equal to *@ScopeId*, remove the specified user from all scopes associated with all sites sharing the permissions of the site specified by *@WebId*.
- If *@WebScopeId* is not equal to *@ScopeId*, remove the specified user from all scopes associated with the site specified by *@WebId*.

@ScopeId: The scope identifier of the scope in the site collection from which to remove the user. See *@WebScopeId* for conditions of equality between *@ScopeId* and *@WebScopeId*.

@RemoveFromCurrentScopeOnly: The bit flag indicating from which scope the user SHOULD be removed. If this flag is set to 1, then **proc_SecRemoveUserFromScopeByLogin** MUST remove the user from the current scope only.

@LoginName: The login name of the user.

@RoleId: The role identifier to remove from the specified scopes. *@RoleId* MUST be NULL or correspond to a valid role. If *@RoleId* is NULL, or is the role identifier of the Guest role (1073741825), then **proc_SecRemoveUserFromScopeByLogin** MUST remove the user corresponding to *@LoginName* from all roles.

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

Return Values: The **proc_SecRemoveUserFromScopeByLogin** stored procedure returns an integer return code, which MUST be in the following table.

Value	Description
0	Successful execution.
1317	UserId not found for site collection or login name.

The **proc_SecRemoveUserFromScopeByLogin** stored procedure MUST return a single result set as follows.

3.1.5.110.1 Site Audit Mask Result Set

The Site Audit Mask Result Set returns audit information about the site specified by *@WebId*. The Site Audit Mask Result Set MUST be returned, and MUST contain only one row, as defined in Site Audit Mask Result Set, section 2.2.5.20.

3.1.5.111 proc_SecRemoveUserFromSite

The **proc_SecRemoveUserFromSite** stored procedure is invoked to remove a user from a site collection. If the user owns site **groups**, then **proc_SecRemoveUserFromSite** will assign site group ownership to the specified new group owner.

```
PROCEDURE proc_SecRemoveUserFromSite(
      @SiteId                        uniqueidentifier,
      @NewSiteGroupOwnerId           int,
      @UserId                        int,
      @RequestGuid                   uniqueidentifier   = NULL OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) for the site collection containing the user to be removed.

@NewSiteGroupOwnerId: An identifier for the user who will take over site group ownership from the user being removed. To succeed, this value MUST be -1 or a valid user identifier (section 2.2.1.13) in the specified site collection, and **NewSiteGroupOwnerId** MUST NOT have the same value as *@UserId*. If the value is -1, then the site collection owner will be the new site group owner.

@UserId: The user identifier for the user who is being removed. If *@UserId* is the site collection owner or secondary **contact**, the change will not succeed. To succeed, this value MUST be an existing user identifier for the specified site collection.

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

Return Values: The **proc_SecRemoveUserFromSite** stored procedure returns an integer return code which MUST be included in the following table.

Value	Description
0	0 is returned whenever the conditions for the alternate return values of 16 and 4335 are not met.

Value	Description
16	*@NewSiteGroupOwnerId* equals *@UserId*. No changes were made.
4335	*@UserId* is the site collection owner or secondary contact. No changes were made.

The **proc_SecRemoveUserFromSite** stored procedure MUST NOT return any result sets.

3.1.5.112 proc_SecRemoveUserFromSiteGroup

The **proc_SecRemoveUserFromSiteGroup** stored procedure is invoked to remove a user from a site **group** in a site collection.

```
PROCEDURE proc_SecRemoveUserFromSiteGroup(
      @SiteId                      uniqueidentifier,
      @GroupId                     int,
      @UserIDToBeDeleted           int,
      @UserID                      int,
      @SiteAdmin                   bit,
      @BelongsToGroup              bit,
      @GroupOwnerId                int,
      @CurrentUserIsOwner          bit,
      @RequestGuid                 uniqueidentifier   = NULL OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection that will be queried for the requested site group.

@GroupId: The site group identifier (section 2.2.1.10) for the site group that the user specified by *@UserIDToBeDeleted* is to be removed from.

@UserIDToBeDeleted: The user identifier (section 2.2.1.13) for the user to remove from the specified site group. If *@UserId* and *@UserIdToBeDeleted* refer to the same user, and the site group permits users to remove themselves from site group membership, then the user specified by *@UserIdToBeDeleted* MUST be removed from the group.

@UserID: The user identifier for the current user who is removing the user to be deleted from the specified site group. If *@UserId* and *@UserIdToBeDeleted* refer to the same user, and the Site Group permits site group members to edit membership, then the user specified by *@UserIdToBeDeleted* is removed from the site group.

@SiteAdmin: A bit flag specifying whether the user specified by *@UserID* is a site collection administrator of the site collection specified by *@SiteId*. If *@SiteAdmin* is set to "1", then **proc_SecRemoveUserFromSiteGroup** MUST remove the user specified by *@UserIDToBeDeleted* from the site group.

@BelongsToGroup: A bit flag specifying whether the user specified by *@UserID* is a **member** of the site group specified by *@GroupId*. If *@BelongsToGroup* is set to "1" and the site group permits site group members to edit membership, and the current user specified in *@UserID* is a member of the site group, then **proc_SecRemoveUserFromSiteGroup** MUST remove the user specified by *@UserIDToBeDeleted* from the site group.

@GroupOwnerId: The user identifier or site group identifier of the site group owner specified by *@GroupId*. *@GroupOwnerId* MUST be set to the user identifier of the current user, or, if *@CurrentUserIsOwner* is set to "1," to the site group identifier of a site group which includes the current user. If *@CurrentUserIsOwner* is set to "0", then *@GroupOwnerId* MUST be ignored.

@CurrentUserIsOwner: A bit flag specifying that the user specified by *@GroupOwnerId* is an owner of the site group specified by *@GroupId*. If *@CurrentUserIsOwner* is set to "1" and *@GroupOwnerId* contains either the user identifier of the owner of the site group, or the site group identifier of a site group which includes the current user, then **proc_SecRemoveUserFromSiteGroup** MUST remove the user from the site group.

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

Return Values: The **proc_SecRemoveUserFromSiteGroup** stored procedure returns an integer return code, which MUST be one of the values in the following table.

Value	Description
0	Successful execution.
5	The current user has insufficient authority to remove the user from the site group.

The **proc_SecRemoveUserFromSiteGroup** stored procedure MUST NOT return any result sets.

3.1.5.113 proc_SecRemoveUserFromSiteGroupByLogin

The **proc_SecRemoveUserFromSiteGroupByLogin** stored procedure is invoked to remove a user identified by login name from a specified site **group**.

```
PROCEDURE proc_SecRemoveUserFromSiteGroupByLogin(
      @SiteId                        uniqueidentifier,
      @GroupId                       int,
      @LoginName                     nvarchar(255),
      @UserID                        int,
      @SiteAdmin                     bit,
      @BelongsToGroup                bit,
      @GroupOwnerId                  int,
      @CurrentUserIsOwner            bit,
      @RequestGuid                   uniqueidentifier   = NULL OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the site group and the user to be removed.

@GroupId: The identifier of the site group containing the user to be removed.

@LoginName: The login name of the user to be removed from the site group.

@UserID: The user identifier (section 2.2.1.13) of the current user.

@SiteAdmin: A bit flag specifying whether the user specified by *@UserID* is a site collection administrator of the site collection specified by *@SiteId*. If *@SiteAdmin* is set to "1", and *@LoginName* specifies an existing user, then **proc_SecRemoveUserFromSiteGroupByLogin** MUST remove the user specified by *@LoginName* from the site group.

@BelongsToGroup: A bit flag specifying whether the user specified by *@UserID* is a **member** of the site group specified by *@GroupId*. If *@BelongsToGroup* is set to "1" and the site group permits users to manage site group memberships, and the current user specified in *@UserID* is a member of the site group, and *@LoginName* specifies an existing user, then **proc_SecRemoveUserFromSiteGroupByLogin** MUST remove the user specified by *@LoginName* from the site group.

@GroupOwnerId: The user identifier or site group identifier (section 2.2.1.10) of the site group owner specified by *@GroupId*. *@GroupOwnerId* MUST be set to the user identifier of the current user, or, if *@CurrentUserIsOwner* is set to "1," to the site group identifier of a site group which includes the current user. If *@CurrentUserIsOwner* is set to "0", *@GroupOwnerId* MUST be ignored.

@CurrentUserIsOwner: A bit flag specifying that the user specified by *@GroupOwnerId* is an owner of the site group specified by *@GroupId*. If *@CurrentUserIsOwner* is set to "1" and *@GroupOwnerId* contains either the user identifier of the owner of the site group, or the site group identifier of a site group which includes the current user, and *@LoginName* specifies an existing user, then **proc_SecRemoveUserFromSiteGroupByLogin** MUST remove the user specified by *@LoginName* from the site group.

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

Return Values: The **proc_SecRemoveUserFromSiteGroupByLogin** stored procedure returns an integer return code, which MUST be included in the following table.

Value	Description
0	Successful execution.
5	The current user has insufficient permissions to remove the specified user from the site group.
1317	UserId not found for site collection or login name.

The **proc_SecRemoveUserFromSiteGroupByLogin** stored procedure MUST NOT return a result set.

3.1.5.114 proc_SecResetItemPerm

The **proc_SecResetItemPerm** stored procedure is invoked to cause a list item to revert its set of permissions so that it inherits permissions from its parent rather than has its own. The previous set of unique permissions for the list item are discarded.

```
PROCEDURE proc_SecResetItemPerm(
      @SiteId                       uniqueidentifier,
      @WebId                        uniqueidentifier,
      @OldScopeId                   uniqueidentifier,
      @Url                          nvarchar(260),
      @DocId                        uniqueidentifier,
      @NewScopeId                   uniqueidentifier = NULL OUTPUT,
      @RequestGuid                  uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the list item.

@WebId: The site identifier (section 2.2.1.11) of the site containing the item.

@OldScopeId: The scope identifier (section 2.2.1.8) of the security scope of the item.

@Url: The store relative-form URL of the item.

@DocId: The **document identifier** (section 2.2.1.2) of the document.

@NewScopeId: This is an output parameter, which provides the scope identifier of the security scope of the item. This MUST be the scope identifier of the item's parent container upon successful execution; otherwise, it remains unchanged.

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

Return Values: The **proc_SecResetItemPerm** stored procedure returns an integer return code which MUST be in the following table.

Value	Description
0	Successful execution.
3	The item specified is not configured to have unique permissions. This value MUST also be returned when the list item is not found at the specified location, or the *@OldScopeID* does not match the specified list item's security scope.

Upon successful execution, the **proc_SecResetItemPerm** stored procedure MUST return one result set as follows.

3.1.5.114.1 Site Audit Mask Result Set

The Site Audit Mask Result Set contains the information about the **Audit Flags** (section 2.2.2.1) associated with the specified Site. It MUST contain a single row. The Site Audit Mask Result Set is defined using T-SQL syntax, as defined in Site Audit Mask Result Set, section 2.2.5.20.

3.1.5.115 proc_SecResetWebToDefaultRoleDefinition

The **proc_SecResetWebToDefaultRoleDefinition** stored procedure is invoked to replace existing roles, role assignments, and permissions for a specified site and its subsites with a default set of role definitions, as specified by input parameters. If the specified site inherits its security permissions, then a new security scope will be made to hold the new security settings. **proc_SecResetWebToDefaultRoleDefinition** will fail if the specified site does not exist or if the specified site has a unique role definition.

```
PROCEDURE proc_SecResetWebToDefaultRoleDefinition(
      @WebId                          uniqueidentifier,
      @AuthorID                       int,
      @AdminsName                     nvarchar(255),
      @AdminsDescription              nvarchar(512),
      @AdminsPermMask                 bigint,
      @AuthorsName                    nvarchar(255),
      @AuthorsDescription             nvarchar(512),
      @AuthorsPermMask                bigint,
      @ContributorsName               nvarchar(255),
      @ContributorsDescription        nvarchar(512),
      @ContributorsPermMask           bigint,
      @BrowsersName                   nvarchar(255),
      @BrowsersDescription            nvarchar(512),
      @BrowsersPermMask               bigint,
      @GuestsName                     nvarchar(255),
      @GuestsDescription              nvarchar(512),
      @GuestsPermMask                 bigint,
      @AnonymousPermMask              bigint,
      @ScopeId                        uniqueidentifier OUTPUT,
      @RequestGuid                    uniqueidentifier   = NULL OUTPUT
);
```

@WebId: The site identifier (section 2.2.1.11) of the site.

@AuthorID: The user identifier (section 2.2.1.13) of the **principal** to be assigned to the Administrator Role. This MUST be the user identifier of an existing principal in the site collection.

@AdminsName: The display name for the Administrator Role. This value MUST NOT be NULL.

@AdminsDescription: The description for the Administrator Role.

@AdminsPermMask: The **WSS Rights Mask** (section 2.2.2.14) for the Administrator Role. This value MUST NOT be NULL.

@AuthorsName: The display name for the Web Designer Role. This value MUST NOT be NULL.

@AuthorsDescription: The description for the Web Designer Role.

@AuthorsPermMask: The **WSS Rights Mask** for the Web Designer Role. This value MUST NOT be NULL.

@ContributorsName: The display name for the Contributor Role. This value MUST NOT be NULL.

@ContributorsDescription: The description for the Contributor Role.

@ContributorsPermMask: The **WSS Rights Mask** for the Contributor Role. This value MUST NOT be NULL.

@BrowsersName: The display name for the Reader Role. This value MUST NOT be NULL.

@BrowsersDescription: The description for the Reader Role.

@BrowsersPermMask: The **WSS Rights Mask** for the Reader Role. This value MUST NOT be NULL.

@GuestsName: The display name for the Guest Role. This value MUST NOT be NULL.

@GuestsDescription: The description for the Guest Role.

@GuestsPermMask: The **WSS Rights Mask** for the Guest Role. This value MUST NOT be NULL.

@AnonymousPermMask: A **WSS Rights Mask** that indicates the rights granted to a user that is anonymous, or has no specific rights.

@ScopeId: The scope identifier (section 2.2.1.8) of the security scope of the updated permissions, returned as an output parameter.

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

Return Values: The **proc_SecResetWebToDefaultRoleDefinition** stored procedure returns an integer return code which MUST be in the following table.

Value	Description
0	Successful execution.
3	Site does not exist or site has a unique role definition associated with it.

If execution is not successful, then the **proc_SecResetWebToDefaultRoleDefinition** stored procedure MUST return no result sets. Otherwise, **proc_SecResetWebToDefaultRoleDefinition** MUST return the Site Audit Mask Result Set six times.

3.1.5.115.1 Site Audit Mask Result Set

The Site Audit Mask Result Set is returned six times upon successful execution. The first is returned after the Administrator Role has been added, the second after the **principal** specified by *@AuthorId* has been newly assigned to the administrator role and security scope, and then once per additional role that is added.

Each Site Audit Mask Result Set contains the **Audit Flags** (section 2.2.2.1) of the site and MUST hold a single row, and is defined in Site Audit Mask Result Set, section 2.2.5.20.

3.1.5.116 proc_SecResolvePrincipal

The **proc_SecResolvePrincipal** stored procedure retrieves information about a **principal** or site **group** in the specified site collection, given the principal's login name, email address, or display name.

```
PROCEDURE proc_SecResolvePrincipal(
      @SiteId                         uniqueidentifier,
      @Input                          nvarchar(255),
      @InputIsEmailOnly               bit,
      @AccountName                    nvarchar(255),
      @DLAlias                        nvarchar(128),
      @DLAliasServerAddress           nvarchar(255),
      @SearchScope                    int,
      @RequestGuid                    uniqueidentifier   = NULL OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection.

@Input: The principal's display name, email address, or login name. This parameter is used to find the principal.

@InputIsEmailOnly: Specifies how input parameters are used to find the principal. If this parameter is 0, then matching in the following **order:**

1. *@Input* to site group display name;

2. *@AccountName* or *@Input* (when *@AccountName* is NULL) as login name;

3. *@Input* as mobile phone number;

4. *@Input* or *@DLAlias* as email address;

5. *@Input* to principal's display name.

For any value of *@InputIsEmailOnly* that is not 0, skip steps 1, 2 and 3.

@AccountName: The principal's login name. If NULL, and *@Input* did not match on display name, then the value of *@Input* will be taken as login name.

@DLAlias: The principal's email alias to search for.

@DLAliasServerAddress: The server portion of an email address. If the principal is a site group whose email address is available, then this is used to construct the full email address in the **{Email}** column in the Principal Information Result Set (section 3.1.5.116.1). If this parameter is NULL and the principal is a site group, **{Email}** MUST return NULL.

@SearchScope: A bit mask signifying the scope in which to search for matching principals. This can have one or more flags set. The following bits within *@SearchScope* can be set to 1.

Value	Description
0x01	User.
0x04	Domain group.
0x08	Site group.

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

Return Values: The **proc_SecResolvePrincipal** stored procedure MUST return an integer return code of 0.

If a match for the principal is found, then the **proc_SecResolvePrincipal** stored procedure MUST return the Principal Information Result Set. Otherwise, it MUST NOT return a result set.

3.1.5.116.1 Principal Information Result Set

The Principal Information Result Set MUST return a single row containing information about the **principal** which uniquely matches the input parameters.

```
          {MatchType}                    int,
          {PrincipalType}                int,
          {Id}                           int,
          {Login}                        nvarchar(255),
          {Email}                        nvarchar(255),
          {DisplayName}                  nvarchar(255);
```

{MatchType}: Designates the type of match found between input variables and a principal in the site **group**. Values MUST be listed in the following table.

Value	Description
1	Matched by login name.
2	Matched by email address.
4	Matched by display name.
5	Matched by mobile phone number.

{PrincipalType}: Designates the type of the principal. Value MUST be listed in the following table.

Value	Description
1	Domain user.
4	Domain Group.
8	Site group.

{Id}: The identifier for the principal. If no match was found then the value is -1.

{Login}: The login name of the principal. This parameter is NULL if no match was found.

{Email}: The email address, for example, someone@example.com, of the principal. This parameter is NULL if no match was found, if no email address is available, or if the principal is a site group but *@DLAliasServerAddress* was NULL.

{DisplayName}: The display name of the principal. The value is NULL if no match was found.

3.1.5.117 proc_SecSetSiteGroupProperties

The **proc_SecSetSiteGroupProperties** stored procedure is invoked to update properties of a site **group**.

```
    PROCEDURE proc_SecSetSiteGroupProperties(
          @SiteId                        uniqueidentifier,
          @GroupId                       int,
          @Title                         nvarchar(255),
          @Description                   nvarchar(512),
          @OwnerID                       int,
          @OwnerIsUser                   bit,
          @UserID                        int,
          @SiteAdmin                     bit,
          @GroupOwnerId                  int,
          @CurrentUserIsOwner            bit,
          @DLAlias                       nvarchar(255),
          @DLErrorMessage                nvarchar(255),
          @DLFlags                       int,
          @DLJobId                       int,
          @DLArchives                    varchar(4000),
          @RequestEmail                  nvarchar(255),
          @Flags                         int,
          @RequestGuid                   uniqueidentifier   = NULL OUTPUT
    );
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the site group to update.

@GroupId: The identifier for the site group to update.

@Title: The new user-friendly display name for the site group. If this parameter is NULL, then the update MUST fail and the return code MUST be 80. However, if the user does not have sufficient permissions, the return code MUST be 5.

@Description: The new description for the site group. This parameter can be NULL.

@OwnerID: The identifier of a user, domain group, or site group to be set as the owner of the site group. This parameter MUST contain the identifier of an existing user, domain group, or site group.

@OwnerIsUser: A bit which specifies whether the value in *@OwnerId* is a user or domain group rather than a site group. A value of 1 specifies that the value in *@OwnerId* is a user or domain group. A value of "0" specifies that the value in *@OwnerId* is a site group identifier (section 2.2.1.10).

@UserID: The user identifier (section 2.2.1.13) of the current user. This parameter MUST be ignored by **proc_SecSetSiteGroupProperties**.

@SiteAdmin: A bit flag specifying whether the current user has site collection administrator permission level. A value of 1 specifies that the current user has site collection administrator permission level. This parameter is used to check user's permission to change the site group permissions and it can be NULL.

@GroupOwnerId: The user identifier or site group identifier to compare with the existing owner of the site group. Site group management permissions MUST be granted if the value in *@GroupOwnerId* matches the identifier of the existing owner of the site group and the value of *@CurrentUserIsOwner* is "1". This parameter is used to check user's permission to change the site group permissions and it can be NULL.

@CurrentUserIsOwner: A bit flag specifying whether the value in *@GroupOwnerId* is to be compared with the owner of the site group. Site group management permissions MUST be granted if the value in *@GroupOwnerId* matches the identifier of the existing owner of the site group and the value of *@CurrentUserIsOwner* is "1". This parameter can be NULL. If this parameter is NULL, the update MUST fail and the return code MUST be 80. However, if the user does not have sufficient permissions, the return code MUST be 5.

@DLAlias: The new email address for the **distribution list** associated with this site group. This parameter can be NULL.

@DLErrorMessage: The most recent error message returned by an asynchronous email distribution list operation, if any. This parameter can be NULL.

@DLFlags: An integer containing new status flags for the distribution list associated with this group, as defined for the **DLFlags** column in Sec_SiteGroupsView (section 2.2.7.6). This parameter can be NULL.

@DLJobId: Contains the job ID of a new currently pending asynchronous distribution list operation. A value of 0 or NULL specifies there is no pending operation. This parameter can be NULL.

@DLArchives: An array of bytes containing a new list of pairs of site identifiers (section 2.2.1.11) and list identifiers (section 2.2.1.5) defining additional lists, which are the recipients of email sent to the distribution list via the email address in *@DLAlias*. Each pair MUST be stored as a string, with commas separating the list's parent site **document identifier** (section 2.2.1.2) and the list identifier, and with semicolons following each pair. This parameter can be NULL (this will overwrite the current value).

@RequestEmail: The new email address used to send a request to join or depart this site group. This parameter can be NULL (this will overwrite the current value).

@Flags: Contains the new membership permissions bit flags for the site group. The bit flags are defined in the description of the **DLFlags** column in Sec_SiteGroupsView (section 2.2.7.6).

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

Return Values: The **proc_SecSetSiteGroupProperties** stored procedure MUST return an integer return code from the following table.

Value	Description
0	Successful execution.
5	The current user does not have sufficient permissions to manage the site group, or the site collection is set to read-only.
80	An update error occurred or a parameter was invalid.

The **proc_SecSetSiteGroupProperties** stored procedure MUST NOT return a result set.

3.1.5.118 proc_SecSetUserAccountDirectoryPath

The **proc_SecSetUserAccountDirectoryPath** stored procedure is invoked to set or clear the **user account directory path** for a specified site collection.

```
PROCEDURE proc_SecGetUserAccountDirectoryPath(
      @SiteId                        uniqueidentifier,
      @Restriction                   nvarchar(512),
      @RequestGuid                   uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection.

@Restriction: Specifies the user account directory path for the specified site collection. If *@Restriction* is not NULL the **proc_SecSetUserAccountDirectoryPath** stored procedure MUST set the user account directory path for the site collection, and also set the 0x00080000 bit of the site **Collection Flags** (section 2.2.2.9) to 1. If *@Restriction* is NULL, then the **proc_SecSetUserAccountDirectoryPath** stored procedure MUST set the user account directory path for the site collection to NULL, and also set the 0x00080000 bit of the site **Collection Flags** to 0.

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

Return Values: The **proc_SecSetUserAccountDirectoryPath** stored procedure returns an integer which MUST be 0.

The **proc_SecGetUserAccountDirectoryPath** stored procedure MUST NOT return a result set.

3.1.5.119 proc_SecSetWebRequestAccess

The **proc_SecSetWebRequestAccess** stored procedure is invoked to set the request access email address for a specified site. Site access requests to the particular site are routed to the specified request access email address.

```
PROCEDURE proc_SecSetWebRequestAccess(
      @WebId                         uniqueidentifier,
      @RequestAccessEmail            nvarchar(255),
      @RequestGuid                   uniqueidentifier = NULL OUTPUT
);
```

@WebId: The site identifier (section 2.2.1.11) of the site to set the request access email address.

@RequestAccessEmail: The new request access email address.

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

Return Values: The **proc_SecSetWebRequestAccess** stored procedure MUST return an integer return code of 0.

The **proc_SecSetWebRequestAccess** stored procedure MUST NOT return a result set.

3.1.5.120 proc_SecUpdateAnonymousPermMask

The **proc_SecUpdateAnonymousPermMask** stored procedure is invoked to modify the permissions for the anonymous user. The site MUST have a unique scope identified by *@ScopeId* rather than an inherited scope. If the scope specified by *@SiteId*, *@WebId*, and *@ScopeId* does not exist, then the change MUST NOT occur.

```
PROCEDURE proc_SecUpdateAnonymousPermMask(
      @SiteId                        uniqueidentifier,
      @WebId                         uniqueidentifier,
      @ScopeId                       uniqueidentifier,
      @AnonymousPermMask             bigint,
      @RequestGuid                   uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the scope.

@WebId: The site identifier (section 2.2.1.11) of the site associated with the scope.

@ScopeId: The scope identifier (section 2.2.1.8) of the scope containing the permission set.

@AnonymousPermMask: A **WSS Rights Mask** (section 2.2.2.14). The scope's permissions for anonymous users MUST be updated to this value.

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

Return Values: The **proc_SecUpdateAnonymousPermMask** stored procedure returns an integer return code which MUST be 0.

The **proc_SecUpdateAnonymousPermMask** stored procedure MUST NOT return a result set.

3.1.5.121 proc_SecUpdateDomainGroupMapData

The **proc_SecUpdateDomainGroupMapData** stored procedure is invoked to update the domain group map cache of a specified site collection.

```
    PROCEDURE proc_SecUpdateDomainGroupMapData(
          @SiteId                        uniqueidentifier,
          @CacheVersion                  bigint,
          @DomainGroupMapCache           varbinary(max),
          @RequestGuid                   uniqueidentifier = NULL OUTPUT
    );
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection to update. This parameter MUST correspond to a valid site collection, and MUST NOT be NULL.

@CacheVersion: A signed 64-bit (8-byte) integer containing the cache version number. This value is derived from the value of the **SecurityVersion** field contained in the site collection Access Control List (ACL). See WSS ACL (section 2.2.4.6). This parameter MUST NOT be NULL.

@DomainGroupMapCache: The domain group map value is defined in WSS External Group Map Cache Format (section 2.2.4.7).

Value	Bytes	Description
CachedVersion	8	The current cached security version. This is the same value as *@CacheVersion*.
DGCount	4	Count of Domain Groups.
REPEAT		Repeat the following fields "DGCount" times.
DGSigSize	4	Size of the domain group signature, in bytes.
DGSig	DGSigSize	Value of the domain group signature.
PIDCount	4	Count of PIDs (principal IDs) in this domain group.
PIDs	4*PIDCount	The value of each PID.

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

Return Values: The **proc_SecUpdateDomainGroupMapData** stored procedure MUST return an integer return code of 0.

The **proc_SecUpdateDomainGroupMapData** stored procedure MUST NOT return a result set.

3.1.5.122 proc_SecUpdateRoleDef

The **proc_SecUpdateRoleDef** stored procedure is invoked to update a role definition with a new title, description, display order, **Role Definition** type (section 2.2.3.16), and **WSS Rights Mask** (section 2.2.2.14).

```
    PROCEDURE proc_SecUpdateRoleDef(
          @SiteId                        uniqueidentifier,
          @WebId                         uniqueidentifier,
          @RoleId                        int,
          @Title                         nvarchar(255),
          @Description                   nvarchar(512),
          @Order                         int,
          @Type                          tinyint,
          @PermMask                      bigint,
          @RequestGuid                   uniqueidentifier = NULL OUTPUT
    );
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the role definition to be updated.

@WebId: The site identifier (section 2.2.1.11) of the site containing the role definition to be updated.

@RoleId: The role identifier of the role definition to be updated.

@Title: The title of the role definition for display in the front-end web server. This MUST be a value that does not match an existing role definition title within the site, and MUST NOT be NULL.

@Description: The description of the role definition for display in the front-end web server.

@Order: An integer value specifying the relative position in which this role definition is displayed in the front-end web server. Multiple role definitions can have the same *@Order* value. This value MUST NOT be NULL.

@Type: The **Role Definition** type (section 2.2.3.16) value for the updated role.

@PermMask: A **WSS Rights Mask** that specifies the rights to grant to the role definition. If this field is not NULL, then the **WSS Rights Mask** associated with the role definition MUST be set to this value and any permissions associated with this role definition in the site and site collection MUST be modified. If this field is NULL, then the existing **WSS Rights Mask** associated with the role definition MUST NOT be changed.

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

Return Values: The **proc_SecUpdateRoleDef** stored procedure returns an integer return code which MUST be in the following table.

Value	Description
0	Successful execution.
80	The role definition could not be updated.

Upon successful execution, the **proc_SecUpdateRoleDef** stored procedure MUST return a single result set as follows.

3.1.5.122.1 Site Audit Mask Result Set

The Site Audit Mask Result Set contains information about the **Audit Flags** (section 2.2.2.1) associated with the site. If execution is successful, then the Site Audit Mask Result Set MUST be returned, and MUST contain one row.

The Site Audit Mask Result Set is defined in Site Audit Mask Result Set, section 2.2.5.20.

3.1.5.123 proc_SecUpdateUser

The **proc_SecUpdateUser** stored procedure is invoked to update the display name, email address, notes, or administrative privileges listed in the user information associated with a **principal**.

```
    PROCEDURE proc_SecUpdateUser(
        @SiteId                        uniqueidentifier,
        @CurrentUserId                 int,
        @UserIdToUpdate                int,
        @Title                         nvarchar(255),
        @Email                         nvarchar(255),
        @Notes                         nvarchar(1023),
        @SiteAdmin                     bit,
        @Flags                         int,
        @RequestGuid                    uniqueidentifier = NULL OUTPUT
  );
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing both the current user and the principal to be updated.

@CurrentUserId: The user identifier (section 2.2.1.13) of the current user making the update request.

@UserIdToUpdate: The user identifier of the principal whose user information is to be updated.

@Title: The display name to be set in the principal's user information. If this parameter is NULL, then the user information update MUST fail.

@Email: The email address to be set in the principal's user information. If this parameter is NULL, then the user information update MUST fail.

@Notes: The notes text to be set in the principal's user information. If this parameter is NULL, then the user information update MUST fail.

@SiteAdmin: A bit specifying whether or not to set the principal as a site collection administrator.

- When this parameter is set to "1", if the current user is a site collection administrator, and the principal is not a domain group, then the principal MUST be set as a site collection administrator.
- When the *@SiteAdmin* parameter is set to "0", **proc_SecUpdateUser** MUST clear the site collection administrator flag in the user information for the principal.
- If the parameter is NULL, then **proc_SecUpdateUser** MUST make no changes to the site collection administrator flag.

@Flags: A flag specifying the **UserInfo Flags** (section 2.2.2.11) to be set in the principal's user information.

- If the current user is NOT a site collection administrator, then the value MUST be NULL.
- If the parameter is NULL, then **proc_SecUpdateUser** MUST make no changes to the site collection administrator flag.

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

Return Values: The **proc_SecUpdateUser** stored procedure returns an integer return code which MUST be in the following table.

Value	Description
0	Successful execution.
1399	Cannot be a site collection administrator.
4335	Cannot successfully remove the site collection administrator privilege from the principal, as no other site collection administrator was found.

The **proc_SecUpdateUser** stored procedure MUST return no result sets.

3.1.5.124 proc_SecUpdateUserActiveStatus

The **proc_SecUpdateUserActiveStatus** stored procedure marks a user as an active user in a site collection.

```
PROCEDURE proc_SecUpdateUserActiveStatus(
      @SiteId                        uniqueidentifier,
      @UserId                        int,
      @RequestGuid                    uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the user.

@UserId: The user identifier (section 2.2.1.13) for the user to be marked as active user. If the user is not found or if the user is the system account, then **proc_SecUpdateUserActiveStatus** MUST take no action.

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

Return Values: The **proc_SecUpdateUserActiveStatus** stored procedure returns an integer return code which MUST be 0.

The **proc_SecUpdateUserActiveStatus** stored procedure MUST return no result sets.

3.1.5.125 proc_TakeOverCheckOut

The **proc_TakeOverCheckOut** stored procedure is invoked to override checkout for a document that is checked out and has no checked-in draft or published version.

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

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the document.

@ListId: The list identifier (section 2.2.1.5) of the list containing the document.

@DirName: The directory name of the document.

@LeafName: The leaf name of the document.

@UserId: The user identifier (section 2.2.1.13) for the current user. This value MUST refer to an existing user identifier for the specified site collection.

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

Return Values: The **proc_TakeOverCheckOut** stored procedure returns an integer return code, which MUST be listed in the following table.

Value	Description
0	Successful execution, or the user specified by *@UserId* does not exist, or specified document is already checked out to the specified user.

Value	Description
2	File Not Found: The specified document was not found; or the document is not checked out; or the document is checked out, but another current version of the document exists.

The **proc_TakeOverCheckOut** stored procedure MUST NOT return any result sets.

3.1.5.126 proc_TransferStream

The **proc_TransferStream** stored procedure overwrites the document content corresponding to *@NewDocId* with the document content corresponding to *@TempDocId*. It MUST NOT write document content for *@NewDocId* if no document corresponding to *@NewDocId* existed before the stored procedure was invoked. The stored procedure MUST delete the document content corresponding to *@TempDocId* regardless of whether a document corresponding to *@NewDocId* existed. The document content corresponding to *@TempDocId* MUST have been established by **proc_WriteChunkToAllDocStreams** (section 3.1.5.135), **proc_WriteExternalTokenToAllDocStreams** (section 3.1.5.136) or **proc_WriteRBSTokenToAllDocStreams** (section 3.1.5.137) prior to invoking this stored procedure.

```
PROCEDURE dbo.proc_TransferStream(
    @SiteId uniqueidentifier,
    @NewDocId uniqueidentifier,
    @NewLevel tinyint,
    @TempDocId uniqueidentifier,
    @DocSize int,
    @IsExternal bit);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site containing the document.

@NewDocId: The **document identifier** (section 2.2.1.2) of the document to be overwritten.

@NewLevel: The publishing level of the document to be overwritten.

@TempDocId: The document identifier corresponding to the source document content.

@DocSize: The size in bytes of the document content corresponding to *@TempDocId*.

@IsExternalBit: If *@TempDocId* was established by **proc_WriteExternalTokenToAllDocStreams**, this MUST be 1; otherwise, it MUST be 0.

Return Values: The **proc_TransferStream** stored procedure returns 0.

The **proc_TransferStream** stored procedure MUST NOT return a result set.

3.1.5.127 proc_UncheckoutDocument

The **proc_UncheckoutDocument** stored procedure is invoked to release a checked out document. The current document is set to the most recent previous published or draft version, or to the version currently checked out, depending on the document state and **proc_UncheckoutDocument**'s parameters.

```
PROCEDURE proc_UncheckoutDocument (
      @SiteId                       uniqueidentifier,
      @WebId                        uniqueidentifier,
      @DirName                      nvarchar(256),
      @LeafName                     nvarchar(128),
      @UserId                       int,
      @ForceUncheckout              bit,
      @RequestGuid                  uniqueidentifier = NULL OUTPUT
```

```
    );
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the document to have its **short-term lock** released or its check out reverted.

@WebId: The site identifier (section 2.2.1.11) of the site containing the document.

@DirName: The document's directory name.

@LeafName: The document's leaf name.

@UserId: The user identifier (section 2.2.1.13) of the current user.

@ForceUncheckout: Specifies whether to force the reversion of a check out of the document held by another user. If this parameter is set to "1", then the checkout of the document MUST be released, and the most recent previous draft or published version MUST be reverted to the current version, even if the current user is not the user the document is checked out to. The owner of the checkout can also force the reversion of a checkout. If *@ForceUncheckout* is not set to "1", then the check-out of the document will not be released, and the most recent previous draft or published version will not revert to the current version if the current user is not the user the document is checked out to.

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

Return Values: The **proc_UncheckoutDocument** stored procedure returns an integer return code which MUST be listed in the following table.

Value	Description
0	Successful execution.
3	The document was not found at the specified location.
33	The document is long-term checked out by another user.
158	The document is not checked out, or does not have a short-term lock, or the current user attempted to release a short-term lock on a document that is checked out, or to revert the check out of a document that has a short-term lock.
167	The document is long-term checked out and the current user attempted to release a short-term lock.
173	Another user has checked out the document or has a short-term lock on the document, and *@Force* is not set to "1".
2401	A checked-out document was specified and no published or draft version of the document exists to revert to.
6002	The current user attempted to release a shared-lock.
6009	The current user attempted to release a lock, but the file is short-term locked by someone else.

The **proc_UncheckoutDocument** stored procedure MUST return 2 or 3 result sets in the order specified in the following sections.

3.1.5.127.1 Link Info Result Set

The Link Info Result Set returns information about all forward links and backward links associated with the document. The Link Info Result Set MUST be returned and MUST contain one row for each forward link and backward link associated with the specified document after the **short-term lock** has been released or the checkout has been reverted.

The Link Info Result Set is defined in Link Information Result Set, section 2.2.5.11.

3.1.5.127.2 Document Metadata Result Set

The Document Metadata Result Set returns the metadata for the specified document after the **short-term lock** has been released or the checkout has been reverted. The Document Metadata Result Set MUST be returned and MUST contain one row for the specified document.

The Document Metadata Result Set is defined in Document Metadata Result Set, section 2.2.5.6.

3.1.5.127.3 NULL Result Set

The NULL Result Set returns no data. The NULL Result Set MUST only be returned if the Document Metadata Result Set is not empty and *@WebId* is NULL. The NULL Result Set MUST return zero rows in a schema containing a single NULL column.

3.1.5.127.4 Event Receivers Result Set

The Event Receivers Result Set contains information about the list item event receivers defined for this document.

The Event Receivers Result Set MUST only be returned if the Document Metadata Result Set is not empty and *@WebId* is not NULL. There MUST be one row for each event receiver with an **Event Host** type (section 2.2.3.5) of 3 (list item) registered for the specified document after the **short-term lock** has been released or the checkout has been reverted.

The Event Receivers Result Set is defined in Event Receivers Result Set, section 2.2.5.9.

3.1.5.128 proc_UpdateDiskUsed

The **proc_UpdateDiskUsed** stored procedure is called to update the disk-used size, in bytes, of a **site collection**.

```
PROCEDURE proc_UpdateDiskUsed (
@SiteId uniqueidentifier,
@bUpdateTimeStampForce bit = 0
);
```

@SiteId: The **site collection identifier** of the site collection whose disk-used size is to be updated.

@bUpdateTimeStampForce: A bit flag specifying whether to update the time stamp of the **LastContentChange** of the site collection. If the value is 1, the **LastContentChange** of the site collection MUST be updated with the current time. If the value is 0, the **LastContentChange** MUST be unmodified.

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

The **proc_UpdateDiskUsed** stored procedure MUST NOT return any result sets.

3.1.5.129 proc_UpdateDocBuildDependencySet

The **proc_UpdateDocBuildDependencySet** stored procedure is invoked to store an updated version of a build dependency set for a specified document.

```
PROCEDURE[proc_UpdateDocBuildDependencySet](
      @DocSiteId                      uniqueidentifier,
      @DocDirName                     nvarchar(256),
      @DocLeafName                    varchar(128),
      @Level                          tinyint,
      @BuildDependencySet             varbinary(max),
      @RequestGuid                    uniqueidentifier = NULL OUTPUT

);
```

@DocSiteID: The site collection identifier (section 2.2.1.9) of the site collection containing the document.

@DocDirName: The directory name of the document.

@DocLeafName: The leaf name of the document.

@Level: The **Publishing Level** type (section 2.2.2.6) value of the document containing the dependency set to update.

@BuildDependencySet: The build dependency set for the document. This parameter MUST NOT be NULL. Use **proc_DeleteDocBuildDependencySet** to set the document dependencies to NULL.

@RequestGuid: The optional request identifier for the current request

Return Values: The **proc_UpdateDocBuildDependencySet** stored procedure MUST return an integer return code of 0.

The **proc_UpdateDocBuildDependencySet** stored procedure MUST NOT return a result set.

3.1.5.130 proc_UpdateDocument

The **proc_UpdateDocument** stored procedure is invoked to update the metadata and **stream** of a document.

```
    PROCEDURE proc_UpdateDocument(
          @DocSiteId                      uniqueidentifier,
          @DocWebId                       uniqueidentifier,
          @DocDirName                     nvarchar(256),
          @DocLeafName                    nvarchar(128),
          @SendingContent                 bit,
          @DocMetaInfo                    varbinary(max),
          @DocSize                        int,
          @DocMetaInfoSize                int,
          @DocFileFormatMetaInfo          varbinary(max),
          @DocFileFormatMetaInfoSize      int,
          @DocDirty                       bit,
          @DocVersion                     int,
          @DocMetaInfoVersion             int,
          @DocFlags                       int,
          @DocIncomingCreatedDTM          datetime,
          @DocIncomingDTM                 datetime,
          @ThicketMainFile                bit,
          @GetWebListForNormalization     bit,
          @VersioningEnabled              bit,
          @EnableMinorVersions            bit,
          @UserId                         int,
          @AttachmentOp                   int,
          @PutFlags                       int,
          @UpdateFlags                    int,
          @CharSet                        int,
          @ProgId                         nvarchar(255),
          @AuditMask                      bit,
          @WebParts                       bit,
          @Comment                        nvarchar(1023),
          @IsModerate                     bit,
          @NeedsDraftOwnerRestriction     bit,
          @WelcomePageUrl                nvarchar(260),
          @WelcomePageParameters         nvarchar(max),
          @VirusVendorID                 int,
          @VirusStatus                   int,
          @VirusInfo                     nvarchar(255),
          @LockTimeout                   int,
          @RefreshLock                   bit,
          @SharedLock                    bit,
          @FileFragmentsToDelete         bigint,
          @FileFragmentPartitionsToDelete nvarchar(max),
          @DocContentVerBump             int,
          @@DocId                        uniqueidentifier             OUTPUT,
          @@Level                        tinyint                      OUTPUT,
          @@DoclibRowId                  int                          OUTPUT,
          @@DocDTM                       datetime                     OUTPUT,
          @@DocUIVersion                 int                          OUTPUT,
          @@DocFlagsOut                  int                          OUTPUT,
          @@DocVersionOut                int                          OUTPUT,
          @RequestGuid                   uniqueidentifier = NULL      OUTPUT
    );
```

@DocSiteId: The site collection identifier (section 2.2.1.9) for the site collection containing the document to be updated.

@DocWebId: The site identifier (section 2.2.1.11) for the site containing the document.

@DocDirName: The directory name of the document location.

@DocLeafName: The leaf name of the document location.

@SendingContent: Specifies whether or not to store the document stream in the back-end database server. MUST be 1 if the document stream of the document is intended to be stored in the back-end database server; otherwise, it MUST be 0.

@DocMetaInfo: The metadata information for the document to be stored. If there is no metadata information for this document, then this parameter MUST be NULL.

@DocSize: Final size in bytes of the document stream of the document. This MUST be 0 if *@SendingContent* is 0.

@DocMetaInfoSize: Size in bytes of the document's metadata. This MUST be 0 if *@DocMetaInfo* is NULL.

@DocFileFormatMetaInfo: The file format metadata stream for the document.

@DocFileFormatMetaInfoSize: Size in bytes of the document's file format metadata.

@DocDirty: A bit flag specifying whether the document has dependencies, such as links to other items, which need to be updated. If this parameter is set to "1", the document has dependencies to be updated. This flag MUST be stored by the back-end database server for subsequent updating of the dependent information.

@DocVersion: The internal version number of the document content to check on update. If this value is NULL, or the current internal version number value is NULL, or this value does not equal the current internal version number value of the document content, then the entire document MUST NOT be updated. If the current internal version number value is NULL or *@PutFlags* contains 0x00800000 (from **Put Flags**), then the current internal version number MUST NOT be incremented. Otherwise it MUST be incremented by an implementation-specific amount.

@DocMetaInfoVersion: The internal version number of the document metadata to check on update. If this value is NULL or does not equal the current internal version number value of the document metadata, then the entire document MUST NOT be updated. If the document was updated, then this value MUST be incremented by an implementation-specific amount.

@DocFlags: A **document flag** value with metadata describing the document to be updated.

@DocIncomingCreatedDTM: The datetime in **UTC** of the creation time stamp of the document. If this value is null, the current creation date of the document MUST NOT be updated.

@DocIncomingDTM: The datetime in UTC of the current modification time stamp of the document. If this value is null, then the current last modification time stamp of the document MUST be set to the current UTC datetime of the back-end database server.

@ThicketMainFile: Specifies whether the document is a thicket main file. If this parameter is set to "1", then the document is a thicket main file. If this parameter is set to "0", then the document is not a thicket main file or a thicket supporting file. If this parameter is NULL, then this document is a thicket supporting file.

@GetWebListForNormalization: If this parameter is set to "1", then upon successful completion the procedure MUST return list of sites in a Site List For Normalization Result Set (section 3.1.5.130.1) whose immediate parent site is the site containing the document to be updated.

@VersioningEnabled: A bit flag specifying whether historical versioning is enabled for the list that contains this document. If this parameter is set to "0", then a new historical version MUST NOT be created as part of this update. This parameter MUST be ignored for ghosted items and documents that are not list items or in a document library.

@EnableMinorVersions: Specifies whether minor versions are enabled on the document. If this parameter is set to "1", then minor versions MUST be enabled; otherwise, minor versions MUST NOT be enabled. This parameter MUST be ignored for ghosted items and documents that are not list items or in a document library.

@UserId: The user identifier (section 2.2.1.13) of the current user that is making the request to the front-end web server.

@AttachmentOp: An integer containing **Attachments Flags** (section 2.2.3.1) describing the document.

@PutFlags: A **Put Flags** (section 2.2.2.7) value that specifies document update options.

@UpdateFlags: A bit field that tracks additional document change options. If this parameter is "1", then links that are no longer applicable after the document is updated MUST be deleted. If this parameter is not "1", it MUST be ignored.

@CharSet: An identifier for the **code page** to associate with the document. If this value is NULL, then there MUST be no code page associated with the document.

@ProgId: Specifies a preferred application to open this document. The **ProgId** is used to distinguish among different applications that save files with a given file extension (for example, different editing applications for HTML or XML files). If this value IS NULL, then there MUST be no preferred application for this document.

@AuditMask: A bit flag specifying whether to return the current **Audit Flags** (section 2.2.2.1) values for the document. If this parameter is set to "1", then Site Audit Mask Result Set MUST be returned on successful completion.

@WebParts: A bit flag specifying whether or not the update to the document includes updates to web parts. If this parameter is set to "1", then any links to web parts contained inside the document MUST be removed from the back-end database server.

@Comment: A description provided when a document is checked in or published, which could be displayed by a version management UI. This value MUST be updated for the document even if it is not currently checked out.

@IsModerate: A bit flag specifying whether moderation is in effect for this document. If this parameter is set to "1", moderation is in effect on this document, and moderation rules MUST be applied to prevent document publication and to implement an approval process when the document is updated.

@NeedsDraftOwnerRestriction: A bit flag specifying whether or not the draft owner for this document is the only user allowed to update it. If this parameter is set to "1" and a draft owner exists for the document, then the document MUST only be updated if *@UserId* is the draft owner.

@WelcomePageUrl: Specifies a welcome page to redirect to when the document is fetched. If this value is NULL, then there MUST be no redirection.

@WelcomePageParameters: Specifies query string or bookmark parameters, if any, to append to the URI specified by *@WelcomePageUrl*. If this value is NULL, then there MUST be no parameters added.

@VirusVendorID: If provided, specifies the vendor identifier of the virus scanner that processed this document.

@VirusStatus: A **Virus Status** (section 2.2.3.18) value specifying the current virus check status of this document.

@VirusInfo: A string containing a provider-specific message returned by the virus scanner when it last processed the document. This value MUST be NULL if the document does not exist or if the document has not been processed by a virus scanner.

@LockTimeout: An integer value specifying the number of minutes a **short-term lock** is to be maintained for the document. A value of 0 means the short-term lock MUST be released immediately. This value MUST be NULL if no short-term lock is being applied, refreshed, or released on the document.

@RefreshLock: Specifies whether to refresh a short-term lock on the document. If *@LockTimeout* is NULL, this parameter MUST be ignored.

- If this parameter is set to "1", then the existing short-term lock on the document MUST be set to remain for the number of minutes specified by *@LockTimeout*.
- If this parameter is "0", then the time remaining for an existing short-term lock on the document MUST remain unchanged.
- If there is no short-term lock on the document, then this parameter MUST be NULL.

@SharedLock: Specifies whether the short-term lock to obtain, refresh, or release SHOULD be of type shared or exclusive. If this parameter is "1", then the lock type MUST be shared. Otherwise, the lock type MUST be exclusive. If *@LockTimeout* is NULL, then this parameter MUST be ignored.

@FileFragmentsToDelete: An implementation-specific file fragment identifier. All implementation-specific file fragments with implementation-specific file fragment identifiers less than *@FileFragmentsToDelete* MUST be deleted. If this parameter is not NULL, then *@FileFragmentPartitionsToDelete* MUST NOT be NULL. If *@SystemUpdate* is 1 or *@SendingContent* is 0 or *@VirusStatus* is not NULL and not 3, then this parameter MUST be ignored.

@FileFragmentPartitionsToDelete: A string of comma-separated integers that specifies which implementation-specific file fragment partition identifier to delete. If *@SystemUpdate* is 1 or *@SendingContent* is 0 or *@VirusStatus* is not NULL and not 3, then this parameter MUST be ignored. The format MUST be as follows for file-fragment-partition-list.

```
    file-fragment-partition-list = *file-fragment-partition *("," file-fragment-partition)
    file-fragment-partition = *DIGIT
```

@DocContentVerBump: An integer value used to increase the version of the content stored in the back-end database server. This value is added to the current content version number only if *@SendingContent* is set to 1. If *@SendingContent* is set to 0, then this parameter MUST be ignored.

@@DocId: Output parameter. The **document identifier** (section 2.2.1.2) of the updated document.

@@Level: Output parameter. A **Publishing Level** type (section 2.2.2.6) value indicating the maximum publishing level of the updated document to be returned to the front-end web server if multiple publishing levels of the document are available.

@@DoclibRowId: Output parameter. The identifier of the list item representing the updated document. This value MUST be NULL if the document is not stored in a List.

@@DocDTM: Output parameter. The datetime in UTC of the last modification time stamp of the updated document. If *@DocIncomingDTM* is NULL, then this value MUST be set to the current UTC datetime according to the back-end database server, else the value MUST be set to *@DocIncomingDTM*.

@@DocUIVersion: Output parameter. The user interface version number associated with the updated document. This value MUST be NULL in the case of a document that does not exist.

@@DocFlagsOut: Output parameter. A document flags value, set to the document flags value of the updated document.

@@DocVersionOut: Output parameter. The internal version counter value of the updated document. If the document was not updated, this parameter MUST be set to NULL.

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

Return Values: The **proc_UpdateDocument** stored procedure returns an integer return code, which MUST be listed in the following table.

Value	Description
0	Successful execution.
3	The document does not exist or has been deleted.
5	The user accessing the document is not the owner of the draft, and ownership restriction of the draft is being enforced.
30	An IO error occurred while deleting implementation-specific file fragments.
33	Locking violation or mismatch of *@RefreshLock* and short-term lock status before lock timeout.
80	This document SHOULD NOT have a document stream, but a content file exists.
87	The list item for this document could not be added.
154	The minor UI version number has exceeded its limit.
158	Checkout is enforced, but the document is not checked out.
160	Document is at draft Level and the user accessing the document is not permitted to see Drafts or is not the draft owner.
173	User does not have permissions for the type of checkout desired.

Value	Description
212	The document is locked (either not the current version and is currently checked out, or not enough lock quota); the document cannot be updated.
288	The document is not the current version; the current user is not the owner; the document cannot be updated.
1150	Internal version number mismatch prevented document update.
1630	Unsupported file type.
1816	The site collection does not have enough quota to update the document.
4317	*@FileFragmentsToDelete* is NULL and *@FileFragmentPartitionsToDelete* is not NULL.
6002	A short-term lock could not be obtained or modified because a shared short-term lock already exists.
6009	A short-term lock could not be obtained or modified because an exclusive short-term lock already exists.

The **proc_UpdateDocument** stored procedure can return zero to three result sets in the order listed.

3.1.5.130.1 Site List For Normalization Result Set

The Site List For Normalization Result Set returns a list of URLs for the immediate child sites of the site containing the newly updated document. When the input parameter *@GetWebListForNormalization* is set to "1" and execution has been successful up to the point of updating the document, the site List For Normalization Result Set MUST be returned. The Site List For Normalization Result Set MUST contain one row for each subsite found.

The Site List For Normalization Result Set is defined in URL Result Set (section 2.2.5.25).

3.1.5.130.2 Site Audit Mask Result Set

The Site Audit Mask Result Set returns audit configuration information. When the input parameter *@AuditMask* is set to "1", the Site Audit Mask Result Set MUST be returned, and MUST contain only one row. The Site Audit Mask Result Set is defined in Site Audit Mask Result Set, section 2.2.5.20.

3.1.5.130.3 Lock Information Result Set

The Lock Information Result Set returns **short-term lock** information for the document. The Lock Information Result Set MUST be returned when a successful document update has been completed, and a document has either been checked in, remains checked out, has been locked, or remains locked. The Lock Information Result Set MUST NOT be returned if the document has not been successfully updated or *@LockTimeout* is NULL. The Lock Information Result Set MUST contain only one row.

```
tp_Login                        nvarchar(255),
CheckoutDate                    datetime,
{CheckoutExpires}               datetime;
```

tp_Login: The login name of the user to whom the document is checked out. If the document is currently checked in, then this MUST be NULL.

CheckoutDate: A **datetime** in **UTC** indicating when this document was checked out. If the document is currently checked in, then this MUST be NULL.

{CheckoutExpires}: A **datetime** in UTC indicating when the short-term lock for this document will expire. If the document is currently checked in or has a long term checkout, then this MUST be NULL.

3.1.5.131 proc_UpdateListItem

The **proc_UpdateListItem** stored procedure is invoked to update a list item in a document library or list.

```
PROCEDURE proc_UpdateListItem(
      @SiteId                       uniqueidentifier,
      @WebId                        uniqueidentifier,
      @ListId                       uniqueidentifier,
      @ItemId                       int,
      @RowOrdinal                   int,
      @Size                         int,
      @ExtraItemSize                int                 = NULL,
      @ItemName                     nvarchar(255)       = NULL,
      @UseNvarchar1ItemName         bit                 = 1,
      @ItemDirName                  nvarchar(256)       = NULL      OUTPUT,
      @ItemLeafName                 nvarchar(256)       = NULL      OUTPUT,
      @UserId                       int,
      @Level                        tinyint             = 1         OUTPUT,
      @TimeNow                      datetime,
      @NeedsAuthorRestriction       bit                 = 0,
      @NeedsDraftOwnerRestriction   bit                 = 0,
      @PreserveVersion              bit                 = 0,
      @IsMeetingsList               bit                 = NULL,
      @IsIssueList                  bit                 = NULL,
      @IsNotUserDisplayed           bit                 = NULL,
      @SystemUpdate                 bit                 = 0,
      @ChangeLevel                  bit                 = 0,
      @CheckinItem                  bit                 = 0,
      @NeedClone                    bit                 = 0,
      @MajorVersionsLimit           int                 = 0,
      @MajorMinorVersionsLimit      int                 = 0,
      @IsDocLib                     bit                 = 0,
      @CheckSchemaVersion           int                 = NULL,
      @SortTypeReversed             bit                 = NULL,
      @tp_Ordering                  varchar(512)        = NULL,
      @tp_ThreadIndex               varbinary(512)      = NULL,
      @tp_HasAttachment             bit                 = NULL,
      @tp_ModerationStatus          int                 = NULL,
      @tp_IsCurrent                 bit                 = NULL,
      @tp_ItemOrder                 float               = NULL,
      @tp_Version                   int                 = NULL,
      @tp_InstanceID                int                 = NULL,
      @tp_ContentTypeId             tContentTypeId      = NULL,
      @tp_CopySource                nvarchar(260)       = NULL,
      @tp_HasCopyDestinations       bit                 = NULL,
      @OnRestore                    bit                 = 0,
      @BumpLastDelete               bit                 = NULL,
      @CreateItemVersion            bit                 = 0,
      @UIVersion                    int                 = NULL,
      @NewUIVersion                 int                 = NULL      OUTPUT,
      @ReturnRowset                 bit                 = 1,
      @tp_Author                    int                 = NULL,
      @tp_Editor                    int                 = NULL,
      @tp_Created                   datetime            = NULL,
      @tp_Modified                  datetime            = NULL,
      @tp_WorkflowVersion           int                 = NULL,
      @nvarchar1                    nvarchar(255)       = NULL,
      @nvarchar2                    nvarchar(255)       = NULL,
      @nvarchar3                    nvarchar(255)       = NULL,
      @nvarchar4                    nvarchar(255)       = NULL,
      @nvarchar5                    nvarchar(255)       = NULL,
      @nvarchar6                    nvarchar(255)       = NULL,
      @nvarchar7                    nvarchar(255)       = NULL,
      @nvarchar8                     nvarchar(255)      = NULL,
      @nvarchar9                     nvarchar(255)      = NULL,
      @nvarchar10                    nvarchar(255)      = NULL,
      @nvarchar11                    nvarchar(255)      = NULL,
      @nvarchar12                    nvarchar(255)      = NULL,
      @nvarchar13                    nvarchar(255)      = NULL,
      @nvarchar14                    nvarchar(255)      = NULL,
      @nvarchar15                    nvarchar(255)      = NULL,
      @nvarchar16                    nvarchar(255)      = NULL,
      @nvarchar17                    nvarchar(255)      = NULL,
      @nvarchar18                    nvarchar(255)      = NULL,
      @nvarchar19                    nvarchar(255)      = NULL,
      @nvarchar20                    nvarchar(255)      = NULL,
      @nvarchar21                    nvarchar(255)      = NULL,
      @nvarchar22                    nvarchar(255)      = NULL,
      @nvarchar23                    nvarchar(255)      = NULL,
      @nvarchar24                    nvarchar(255)      = NULL,
      @nvarchar25                    nvarchar(255)      = NULL,
      @nvarchar26                    nvarchar(255)      = NULL,
      @nvarchar27                    nvarchar(255)      = NULL,
      @nvarchar28                    nvarchar(255)      = NULL,
      @nvarchar29                    nvarchar(255)      = NULL,
      @nvarchar30                    nvarchar(255)      = NULL,
      @nvarchar31                    nvarchar(255)      = NULL,
      @nvarchar32                    nvarchar(255)      = NULL,
      @nvarchar33                    nvarchar(255)      = NULL,
      @nvarchar34                    nvarchar(255)      = NULL,
      @nvarchar35                    nvarchar(255)      = NULL,
      @nvarchar36                    nvarchar(255)      = NULL,
      @nvarchar37                    nvarchar(255)      = NULL,
      @nvarchar38                    nvarchar(255)      = NULL,
      @nvarchar39                    nvarchar(255)      = NULL,
      @nvarchar40                    nvarchar(255)      = NULL,
      @nvarchar41                    nvarchar(255)      = NULL,
      @nvarchar42                    nvarchar(255)      = NULL,
      @nvarchar43                    nvarchar(255)      = NULL,
      @nvarchar44                    nvarchar(255)      = NULL,
      @nvarchar45                    nvarchar(255)      = NULL,
      @nvarchar46                    nvarchar(255)      = NULL,
      @nvarchar47                    nvarchar(255)      = NULL,
      @nvarchar48                    nvarchar(255)      = NULL,
      @nvarchar49                    nvarchar(255)      = NULL,
      @nvarchar50                    nvarchar(255)      = NULL,
      @nvarchar51                    nvarchar(255)      = NULL,
      @nvarchar52                    nvarchar(255)      = NULL,
      @nvarchar53                    nvarchar(255)      = NULL,
      @nvarchar54                    nvarchar(255)      = NULL,
      @nvarchar55                    nvarchar(255)      = NULL,
      @nvarchar56                    nvarchar(255)      = NULL,
      @nvarchar57                    nvarchar(255)      = NULL,
      @nvarchar58                    nvarchar(255)      = NULL,
      @nvarchar59                    nvarchar(255)      = NULL,
      @nvarchar60                    nvarchar(255)      = NULL,
      @nvarchar61                    nvarchar(255)      = NULL,
      @nvarchar62                    nvarchar(255)      = NULL,
      @nvarchar63                    nvarchar(255)      = NULL,
      @nvarchar64                    nvarchar(255)      = NULL,
      @int1                          int                = NULL,
      @int2                          int                = NULL,
      @int3                          int                = NULL,
      @int4                          int                = NULL,
      @int5                          int                = NULL,
      @int6                          int                = NULL,
      @int7                          int                = NULL,
      @int8                          int                = NULL,
      @int9                          int                = NULL,
      @int10                         int                = NULL,
      @int11                         int                = NULL,
      @int12                         int                = NULL,
      @int13                        int                = NULL,
      @int14                        int                = NULL,
      @int15                        int                = NULL,
      @int16                        int                = NULL,
      @float1                       float              = NULL,
      @float2                       float              = NULL,
      @float3                       float              = NULL,
      @float4                       float              = NULL,
      @float5                       float              = NULL,
      @float6                       float              = NULL,
      @float7                       float              = NULL,
      @float8                       float              = NULL,
      @float9                       float              = NULL,
      @float10                      float              = NULL,
      @float11                      float              = NULL,
      @float12                      float              = NULL,
      @datetime1                    datetime           = NULL,
      @datetime2                    datetime           = NULL,
      @datetime3                    datetime           = NULL,
      @datetime4                    datetime           = NULL,
      @datetime5                    datetime           = NULL,
      @datetime6                    datetime           = NULL,
      @datetime7                    datetime           = NULL,
      @datetime8                    datetime           = NULL,
      @bit1                         bit                = NULL,
      @bit2                         bit                = NULL,
      @bit3                         bit                = NULL,
      @bit4                         bit                = NULL,
      @bit5                         bit                = NULL,
      @bit6                         bit                = NULL,
      @bit7                         bit                = NULL,
      @bit8                         bit                = NULL,
      @bit9                         bit                = NULL,
      @bit10                        bit                = NULL,
      @bit11                        bit                = NULL,
      @bit12                        bit                = NULL,
      @bit13                        bit                = NULL,
      @bit14                        bit                = NULL,
      @bit15                        bit                = NULL,
      @bit16                        bit                = NULL,
      @uniqueidentifier1            uniqueidentifier   = NULL,
      @ntext1                       nvarchar(max)      = NULL,
      @ntext2                       nvarchar(max)      = NULL,
      @ntext3                       nvarchar(max)      = NULL,
      @ntext4                       nvarchar(max)      = NULL,
      @ntext5                       nvarchar(max)      = NULL,
      @ntext6                       nvarchar(max)      = NULL,
      @ntext7                       nvarchar(max)      = NULL,
      @ntext8                       nvarchar(max)      = NULL,
      @ntext9                       nvarchar(max)      = NULL,
      @ntext10                      nvarchar(max)      = NULL,
      @ntext11                      nvarchar(max)      = NULL,
      @ntext12                      nvarchar(max)      = NULL,
      @ntext13                      nvarchar(max)      = NULL,
      @ntext14                      nvarchar(max)      = NULL,
      @ntext15                      nvarchar(max)      = NULL,
      @ntext16                      nvarchar(max)      = NULL,
      @ntext17                      nvarchar(max)      = NULL,
      @ntext18                      nvarchar(max)      = NULL,
      @ntext19                      nvarchar(max)      = NULL,
      @ntext20                      nvarchar(max)      = NULL,
      @ntext21                      nvarchar(max)      = NULL,
      @ntext22                      nvarchar(max)      = NULL,
      @ntext23                      nvarchar(max)      = NULL,
      @ntext24                      nvarchar(max)      = NULL,
      @ntext25                      nvarchar(max)      = NULL,
      @ntext26                      nvarchar(max)      = NULL,
      @ntext27                      nvarchar(max)      = NULL,
      @ntext28                      nvarchar(max)      = NULL,
      @ntext29                        nvarchar(max)       = NULL,
      @ntext30                        nvarchar(max)       = NULL,
      @ntext31                        nvarchar(max)       = NULL,
      @ntext32                        nvarchar(max)       = NULL,
      @sql_variant1                   sql_variant         = NULL,
      @error_sql_variant1             int                 = 0,
      @sql_variant2                   sql_variant         = NULL,
      @error_sql_variant2             int                 = 0,
      @sql_variant3                   sql_variant         = NULL,
      @error_sql_variant3             int                 = 0,
      @sql_variant4                   sql_variant         = NULL,
      @error_sql_variant4             int                 = 0,
      @sql_variant5                   sql_variant         = NULL,
      @error_sql_variant5             int                 = 0,
      @sql_variant6                   sql_variant         = NULL,
      @error_sql_variant6             int                 = 0,
      @sql_variant7                   sql_variant         = NULL,
      @error_sql_variant7             int                 = 0,
      @sql_variant8                   sql_variant         = NULL,
      @error_sql_variant8             int                 = 0,
      @eventData                      varbinary(max)      = NULL,
      @acl                            varbinary(max)      = NULL,
      @IsFirstRow                     bit                 = 1,
      @RequestGuid                    uniqueidentifier    = NULL OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) for the site collection containing the list item to be updated.

@WebId: The site identifier (section 2.2.1.11) for the site containing the list item.

@ListId: The list identifier (section 2.2.1.5) of the List containing the list item.

@ItemId: The **item identifier** of the list item.

@RowOrdinal: The 0-based ordinal of the row to update in the set of rows representing this list item in the **AllUserData** table (section 2.2.7.3). If a list item requires multiple rows to represent it in the **AllUserData** table because it contains more defined data columns than will fit in a single row, then this parameter specifies which row to update with this call. If more than one row of data for the list item is to be updated, then the front-end web server MUST call **proc_UpdateListItem** once for each updated row. This parameter MUST NOT be NULL.

@Size: The new size, in bytes, of the list item row to be updated. This parameter MUST NOT be NULL.

@ExtraItemSize: The size, in bytes, of the predefined SQL parameter fields in the list item row.

@ItemName: The new display name for the list item.

@UseNvarchar1ItemName: If *@ItemName* is NULL, then this bit flag specifies use of the content of *@nvarchar1* for the new display name for the list item. If *@ItemName* is not NULL, then this parameter MUST be ignored.

@ItemDirName: An output parameter containing the directory name of the updated list item.

@ItemLeafName: An output parameter containing the leaf name of the updated list item.

@UserId: The user identifier (section 2.2.1.13) for the current user. This parameter is used for comparison when the *@NeedsAuthorRestriction* or *@NeedsDraftOwnerRestriction* parameters are set. If this parameter is NULL, then the user identifier stored as the list item's **editor** MUST be used as the current user.

@Level: A **Publishing Level** type (section 2.2.2.6) value specifying the publishing status of the updated list item.

@TimeNow: The current datetime in **UTC**. This parameter MUST NOT be NULL.

@NeedsAuthorRestriction: A bit flag specifying whether or not only the list item's **author** is permitted to update the list item. If this parameter is set to 1, then the current user MUST be the list item's author.

@NeedsDraftOwnerRestriction: A bit flag specifying whether or not only the list item's draft owner is permitted to update the list item. If this parameter is set to 1, the current user MUST be the list item's draft owner.

@PreserveVersion: A bit flag specifying whether to preserve the internal version number of the list item to be updated. If this parameter is set to 1, then the internal version number of the list item MUST NOT be incremented as part of the update. Otherwise, if the list item is updated, then the current internal version number MUST be incremented by an implementation-specific amount.

@IsMeetingsList: A bit flag specifying whether the list item is contained in a Meetings List (a List with a **List Server Template** (section 2.2.3.12) value of 200).

@IsIssueList: A bit flag specifying whether the list item is contained in an Issues List (a List with a **List Server Template** value of 1100). This parameter MUST be ignored.

@IsNotUserDisplayed: A bit flag specifying whether the display name of the current user is not to be displayed in logging events for this update. If this parameter is set to 1, then the display name of the current user MUST NOT be used when **proc_UpdateListItem** logs events due to the update, and the string "***" MUST be used instead.

@SystemUpdate: A bit flag specifying whether to leave the last modification time stamp and the list item editor unchanged during this update. If this parameter is set to 1, then the list item MUST be updated without affecting the current last modification time or list item editor.

@ChangeLevel: A bit flag specifying whether to recalculate the publishing level of the list item as part of the update. If this parameter is set to 1, then the list item's publishing level MUST be recalculated.

@CheckinItem: A bit flag specifying whether to set the publishing level for the updated list item to published. If this parameter is set to 1, then **proc_UpdateListItem** MUST set the updated List Item's publishing level to 1 (published).

@NeedClone: A bit flag specifying whether a copy of the list item as a draft version is made as part of this update. If this parameter is set to 1, then the publishing level specified by the *@Level* parameter MUST also be set to 1 (published).

@MajorVersionsLimit: The number of major versions to keep of the list item.

@MajorMinorVersionsLimit: The total combined number of major versions and minor versions to keep of the list item.

@IsDocLib: A bit flag specifying whether the list item to be updated is contained within a document library. If this parameter is set to 1, then the List specified by *@ListId* MUST be a document library.

@CheckSchemaVersion: This specifies a list schema version number to compare with the list schema version of the list item's containing List. If this parameter is not NULL and does not match the current list schema version number, then the list item MUST NOT be updated.

@SortTypeReversed: If this parameter is 1, then the sort behavior for this item MUST be changed to the opposite of the value of the list item's type. If this parameter is 0 or NULL, then the sort behavior for this item MUST NOT be changed.

@tp_Ordering: This specifies the threading structure for this list item in a legacy Discussion Board List (a list with a **List Base** type (section 2.2.3.11) of 3) as a concatenation of timestamp values in yyyyMMddHHmmss format. For all list items in lists with other **List Base** types, this parameter MUST be NULL.

@tp_ThreadIndex: This specifies the list item's position within a threaded legacy Discussion Board List (a List with a **List Base** type of 3) as a binary structure. For all List Items in Lists with other **List Base** types, this parameter MUST be NULL.

@tp_HasAttachment: A bit flag specifying whether the list item has an associated attachment.

@tp_ModerationStatus: A **Moderation Status** (section 2.2.3.13) value specifying the current moderation approval status of this list item.

@tp_IsCurrent: A bit flag specifying whether this is the current version of this publishing level of the list item.

@tp_ItemOrder: This specifies the implementation-specific order in which the list item SHOULD be displayed with other list items from the same list. This value can be the same as other list items in the list.

@tp_Version: The internal version number value of the list item to update. If this parameter is not NULL and does not equal the current value of the internal version number of the list item, then the item MUST NOT be updated.

@tp_InstanceID: This parameter MUST be ignored.

@tp_ContentTypeId: This specifies the content type identifier of the content type for this list item. **tContentTypeId** is defined in section 2.2.1.1.

@tp_CopySource: The URL used as a source for this list item. If this list item was not copied from a source list item, then this value MUST be NULL.

@tp_HasCopyDestinations: A bit flag specifying whether destination locations have been set for this list item to be copied to. If this list item does not have a destination location set, then this value MUST be 0.

@OnRestore: A bit flag specifying whether this list item is being inserted by an implementation-specific backup restore operation.

@BumpLastDelete: A bit flag specifying whether to update the **tp_LastDeleted** (section 2.2.5.12) property on the List specified by *@ListId*. If set to 0, then the **tp_ LastDeleted** property MUST NOT be updated.

@CreateItemVersion: A bit flag specifying whether to create a new version of the list item as part of this update.

@UIVersion: The UI version number to set for the list item.

@NewUIVersion: An output parameter returning the UI version number set for the list item, which MUST be NULL if one was not assigned.

@ReturnRowset: A bit flag specifying whether to return an Item Update Result Set (section 3.1.5.131.1).

@tp_Author: The user identifier to set as the author of the list item.

@tp_Editor: The user identifier to set as the last editor of the list item.

@tp_Created: A **datetime** in UTC to set as the list item's creation time stamp. If this parameter is NULL or *@SystemUpdate* is "1", then the current time stamp MUST NOT be updated.

@tp_Modified: A datetime in UTC to set as the list item's last modified time stamp. If this parameter is NULL or *@SystemUpdate* is "1", then the current time stamp MUST NOT be updated. If this parameter is NULL and *@SystemUpdate* is not "1", then *@TimeNow* SHOULD be set as the list item's last modified time stamp.

@tp_WorkflowVersion: If this list item is part of a **workflow**, then this specifies the value to set denoting the state of this list item within that workflow. If this list item is not part of a workflow, then this value MUST be NULL.

The next nine columns are duplicated a variable number of times, depending on the List Item's content type and field definitions, with each column referring to a separate **List Server Template**-defined field or user-defined field within the containing List. Each instance of these individual column names is differentiated by a suffix with a numeric value indicated in the column description, which replaces the placeholder "#" symbol below.

@nvarchar#: User-defined columns in the List containing values of type nvarchar. There are 64 columns numbered from 1 to 64. If the column does not contain data, then the value MUST be NULL.

@int#: User-defined columns in the List containing values of type int. There are 16 columns numbered from 1 to 16. If the column does not contain data, then the value MUST be NULL.

@float#: User-defined columns in the List containing values of type float. There are 12 columns numbered from 1 to 12. If the column does not contain data, then the value MUST be NULL.

@datetime#: User-defined columns in the List containing values of type datetime. There are eight columns numbered from 1 to 8. If the column does not contain data, then the value MUST be NULL.

@bit#: User-defined columns in the List containing values of type bit. There are 16 columns numbered from 1 to 16. If the column does not contain data, then the value MUST be NULL.

@uniqueidentifier1: A user-defined column in the List containing values of type uniqueidentifier. If the column does not contain data, then the value MUST be NULL.

@ntext#: User-defined columns in the List containing values of type nvarchar(max). There are 32 columns numbered from 1 to 32. If the column does not contain data, then the value MUST be NULL.

@sql_variant#: User-defined columns in the List containing values of type sql_variant. There are 8 columns numbered from 1 to 8. If the column does not contain data, then the value MUST be NULL.

@error_sql_variant#: An integer which specifies the type to be applied to the corresponding values specified as arguments for the parameter *@sql_variant#*. There are 8 columns numbered from 1 to 8. The following are valid values.

Value	Description
1	Convert the argument value to a **varbinary(2)**.
2	Convert the argument value to a **bit**.
3	Convert the argument value to a **float**.
4	Convert the argument value to a **datetime**.

@eventData: Contains implementation-specific **event (2)** data significant to the front-end web server but otherwise opaque to the back-end database server, to be stored by the back-end database server for eventual writing to a log file.

@acl: The binary serialization of the **WSS ACL Format** (section 2.2.4.6) access control list for the data supplied in *@eventData*, to be stored with the data.

@IsFirstRow: If *@RowOrdinal* is 0, then this parameter MUST be 1. Otherwise this parameter MUST be 0.

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

Return Values: The **proc_UpdateListItem** stored procedure returns an integer return code, which MUST be in the following table.

Value	Description
0	Successful execution.
3	Could not create a unique filename, or the specified site does not exist.
5	The current user does not have sufficient permissions to update the list item.
13	The list item to be added is not valid.
87	Unable to update the list item because the input parameters do not match an existing list item, or an error occurred during a table update operation.
160	A valid user identifier was not specified.
212	The database for the site is locked.
288	The list item cannot be published by the current user because a different user is listed as the draft owner.
1150	Failed to update the list item.
1638	The current schema version of the list does not match the value specified in *@CheckSchemaVersion*.
1639	Only a published list item can be copied.
1816	The site collection is over its allocated size quota.
4005	The specified list item was not found.

The **proc_UpdateListItem** stored procedure MUST return zero or one result sets as follows.

3.1.5.131.1 Item Update Result Set

The Item Update Result Set returns status information about the list item update. The Item Update Result Set MUST only be returned when the *@ReturnRowset* parameter is set to 1, and MUST contain a single row.

```
{ReturnCode}                    int,
{RowsAffected}                  int,
{Version}                       int,
{Author}                        int,
{Id}                            uniqueidentifier,
{AuditMask}                     int,
{InheritAuditMask}              int,
{GlobalAuditMask}               int,
{FullUrl}                       nvarchar(260);
```

{ReturnCode}: The return code value returned by **proc_UpdateListItem**.

{RowsAffected}: The count of the **AllUserData** table (section 2.2.7.3) rows updated by **proc_UpdateListItem**.

{Version}: The current value of the internal version number for the updated list item. This MUST be NULL if the specified list item was not found.

{Author}: The user identifier (section 2.2.1.13) for the author of the updated list item. This MUST be NULL if the specified list item was not found.

{Id}: The list item identifier (section 2.2.1.6) of the updated list item. This MUST be NULL if the specified list item was not found.

{AuditMask}: The **Audit Flags** (section 2.2.2.1) value for the updated list item. This MUST be NULL if the specified list item was not found.

{InheritAuditMask}: The **Audit Flags** value for the updated list item inherited from its containing List or document library. This MUST be NULL if the specified list item was not found.

{GlobalAuditMask}: The global **Audit Flags** value for the site collection specified by *@SiteId*. This MUST be NULL if the specified site collection was not found.

{FullUrl}: The store-relative form URL for the updated list item. This MUST be NULL if the specified list item was not found.

3.1.5.132 proc_UpdateListSettings

The **proc_UpdateListSettings** stored procedure is invoked to update list metadata.

```
PROCEDURE proc_UpdateListSettings(
      @SiteId                         uniqueidentifier,
      @WebId                          uniqueidentifier,
      @ListId                         uniqueidentifier,
      @BaseType                       int,
      @ServerTemplate                 int,
      @Title                          nvarchar(255),
      @ReadSecurity                   int,
      @WriteSecurity                  int,
      @NewDefaultView                 uniqueidentifier,
      @Description                    nvarchar(max),
      @TemplateDirName                nvarchar(256),
      @TemplateLeafName               nvarchar(128),
      @Fields                         tCompressedString,
      @Direction                      int,
      @EventSinkAssembly              nvarchar(255),
      @EventSinkClass                 nvarchar(255),
      @EventSinkData                  nvarchar(255),
      @Flags                          bigint,
      @ImageUrl                       nvarchar(255),
      @ThumbnailSize                  int,
      @WebImageWidth                  int,
      @WebImageHeight                 int,
      @Version                        int,
      @ConvertToGlobalMtgDataList     bit,
      @EmailAlias                     nvarchar(128),
      @SendToLocation                 nvarchar(512),
      @NavBarEid                      int,
      @AddOrRemoveFromNavBar          bit,
      @UseRootFolderForNav            bit,
      @MajorVersionCount              int,
      @MajorMinorVersionCount         int,
      @WorkFlowId                     uniqueidentifier,
      @RemoveSystemAlerts             bit,
      @ValidationFormula              nvarchar(1024),
      @ValidationMessage              nvarchar(1024),
      @LastItemModifiedDate           datetime,
      @RequestGuid                    uniqueidentifier   = NULL OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the list whose metadata is being updated.

@WebId: The site identifier (section 2.2.1.11) of the site containing the list.

@ListId: The list identifier (section 2.2.1.5) of the list to be updated.

@BaseType: A **List Base** type (section 2.2.3.11) value. If *@ConvertToGlobalMtgDataList* is equal to "1", then this value MUST be used to set the **List Base** type; otherwise, the value MUST be ignored.

@ServerTemplate: The identifier for the **List Server Template** (section 2.2.3.12) defining the base structure of this list. If *@ConvertToGlobalMtgDataList* is equal to "1", then this value MUST be used to set the **List Server Template**; otherwise, the value MUST be ignored.

@Title: A user-provided title text for the list. If this parameter is NULL, then the current value MUST NOT be changed.

@ReadSecurity: Specifies the security policy that MUST be set for read access on list items in the list. Valid read security values are listed in the following table.

Value	Description
1	Users with read permissions on this list can read all list items.
2	Users with read permissions on this list can only read their own list items.
NULL	There MUST be no changes to read security for the list.

@WriteSecurity: Specifies the security policy that MUST be set for write access on list items in the list. Valid write security values are listed in the following table.

Value	Description
1	Users with write permissions on this list can create list items and have write access to all list items.
2	Users with write permissions on this list can create list items and have write access to their own list items only.
4	Users have no write access to any list items.
NULL	There MUST be no changes to write security for the list.

@NewDefaultView: A view identifier (section 2.2.1.14) of a defined list view to be displayed by default when navigating to the list. If this parameter is NULL, then the current value MUST NOT be changed.

@Description: A user-provided readable text description for the list. If this parameter is NULL, then the current value MUST NOT be changed.

@TemplateDirName: Directory name for the **document template** associated with the list. This parameter can be NULL. The parameter MUST be ignored if *@TemplateLeafName* is NULL or is an empty string.

@TemplateLeafName: Leaf name for the document template associated with the list. If *@TemplateLeafName* is NULL, then there MUST be no change to the current document template. If *@TemplateLeafName* contains an empty string, then the current document template MUST be set to NULL. If *@TemplateLeafName* is not NULL and is not an empty string, then the current document template MUST be set to the **document identifier** (section 2.2.1.2) of the document template specified by *@TemplateDirName* and *@TemplateLeafName*.

@Fields: Specifies the field definitions for the list, consisting of the concatenation of an implementation-specific version string for the list template, followed by an XML representation of the field definitions. If this parameter is NULL, then the current value MUST NOT be changed.

@Direction: An enumerated value specifying the direction of text flow for front-end web server elements presented by this list. If this parameter is NULL, then the current value MUST NOT be changed. Otherwise, the parameter MUST be one of the following values.

Value	Description
0	No explicit direction is specified.
1	Text flow SHOULD be left to right.
2	Text flow SHOULD be right to left.

@EventSinkAssembly: A .NET assembly name for an event sink handler for the list. If this parameter is NULL, then the current value MUST NOT be changed. If this parameter is an empty string, then the current value MUST be set to NULL.

@EventSinkClass: A .NET assembly class identifier for an event sink handler for the List. If this parameter is NULL, then the current value MUST NOT be changed. If this parameter is an empty string, the current value MUST be set to NULL.

@EventSinkData: Data for an event sink handler for the List. If this parameter is NULL, then the current value MUST NOT be changed. If this parameter is an empty string, then the current value MUST be set to NULL.

@Flags: A **List Flags** (section 2.2.2.5) value for the list. If *@Fields* is not NULL, then this value MUST be bitwise OR with 0x800000 (LP_FIELDSCHEMAMODIFIED). If this parameter is NULL, then the current value MUST NOT be changed.

@ImageUrl: A **site-relative URL** holding an image associated with the list. If this parameter is NULL, then the current value MUST NOT be changed.

@ThumbnailSize: A value specifying the width in pixels used by lists that have a **List Server Template** value of 109 (Image Library Template) to determine the rendering size of an image thumbnail. If this parameter is NULL, then the current value MUST NOT be changed.

@WebImageWidth: A value specifying the width in pixels used by Lists that have a **List Server Template** value of 109 (Image Library Template) to determine the rendering width of an image. If this parameter is NULL, then the current value MUST NOT be changed.

@WebImageHeight: A value specifying the height in pixels used by Lists that have a **List Server Template** value of 109 (Image Library Template) to determine the rendering height of an image. If this parameter is NULL, then the current value MUST NOT be changed.

@Version: An integer used to match on the internal version counter of the List to be updated. This is used to prevent updates with out-of-date data. On successful execution, the internal version counter value of the List will be incremented by 1. If this parameter does not match the current version, then the list MUST NOT be updated.

@ConvertToGlobalMtgDataList: Used to convert the list's data to use global meeting data. If this parameter is set to 1, then the List data MUST be converted to global meeting data and the **List Server Template** specified by *@ServerTemplate* MUST NOT be any of the following.

Value	Description
200	Meeting

Value	Description
202	Meeting User
212	Homepage

If set to any other value or NULL, then the meeting data for this list MUST NOT be changed.

@EmailAlias: The email alias of the list. This alias is used to allow documents and list items to be sent directly to this list through an implementation-specific email-handling feature. If this parameter is NULL then the current value MUST NOT be changed.

@SendToLocation: An implementation-specific Send-To string holding a URL used for the list. If this parameter is NULL, then the current value MUST NOT be changed.

@NavBarEid: Specifies a **navigation node element identifier** for the parent **navigation node** of the node that will represent the list. If *@AddOrRemoveFromNavBar* is NULL or 0, then this parameter MUST be ignored. Otherwise, it MUST contain a valid navigation node element identifier.

@AddOrRemoveFromNavBar: Specifies whether to add or remove this list from the site's top link bar. If this parameter is set to "0" and the list was updated, then the list MUST be removed from the site's top link bar. If this parameter is NULL, then there MUST be no changes to the site's top link bar. If this parameter is set to "1" and the list was updated and the list is not in the site's top link bar, then the list MUST be added to the site's top link bar. To move nodes from one location to another, the navigation node SHOULD be first removed, and then re-added with the new desired setting or else duplicate node entries will occur.

@UseRootFolderForNav: Specifies whether the list's root folder is used for the list entry in the site's top link bar, instead of the default view page. If this parameter is set to "1" and the list was updated, then the root folder of the list MUST be set as the top link bar target. If this parameter is set to "0" and the list was updated, the default view page for the list MUST be set as the top link bar target. If the *@AddOrRemoveFromNavBar* parameter is set to NULL, then this parameter MUST also be NULL. Otherwise, this parameter can be NULL, which MUST have the effect of being set to "0". To change the root folder navigation from one node to another, the current navigation node SHOULD be first removed, and then the new one added with the desired setting or else duplicate node entries will occur.

@MajorVersionCount: Specifies the maximum number of major versions of List Items to be retained by the List.

@MajorMinorVersionCount: Specifies the maximum number of minor versions of List Items to be retained by the List.

@WorkFlowId: Specifies the **workflow** identifier (section 2.2.1.16) of the default workflow of the List.

@RemoveSystemAlerts: A bit flag specifying whether to remove system **alerts** for this List from the email subscriptions for the system administration user. If this parameter is set to "1", then system alerts on the List will be removed.

@ValidationFormula: The implementation-specific list validation formula to be used for items in this list. If this parameter is NULL, the current value MUST NOT be changed.

@ValidationMessage: The message to display to the user when the implementation-specific list validation formula fails. If this parameter is NULL, then the current value MUST NOT be changed.

@LastItemModifiedDate: A time stamp, in **UTC** format, that specifies when the list was last modified. If *@LastItemModifiedDate* is NULL, and any of *@Title*, *@Fields*, *@Flags*, or *@Description* is not NULL, **proc_UpdateListSettings** MUST use the current time. If *@LastItemModifiedDate* is NULL, and *@Title*, *@Fields*, *@Flags*, and *@Description* are also NULL, **proc_UpdateListSettings** MUST make no changes to this value.

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

Return Values: The **proc_UpdateListSettings** stored procedure returns an integer return code that MUST be listed in the following table.

Value	Description
0	Successful execution.
3	The site does not exist or list does not exist.
13	*@ConvertToGlobalMtgDataList* is set to 1 and *@ServerTemplate* is 200, 202, or 212 so list cannot use global meeting data.
15	The document template URL (combination of *@TemplateDirName* and *@TemplateLeafName*) is not valid.
52	Another list with *@Title* already exists.
1150	*@Version* did not match the current version of the list.
5069	*@ReadSecurity* is 0x00000002 (LSReadOnlyMyItems) and *@WriteSecurity* is not NULL and the list contains a field with an implementation-specific list validation constraint.

The **proc_UpdateListSettings** stored procedure MUST return no result sets.

3.1.5.133 proc_UpdateUserInfoInTableFromRowUpdater

The **proc_UpdateUserInfoInTableFromRowUpdater** stored procedure updates the user information data for the specified user.

```
PROCEDURE proc_UpdateUserInfoInTableFromRowUpdater(
      @SiteId                        uniqueidentifier,
      @UserId                        int,
      @Title                         nvarchar(255)      = NULL,
      @Email                         nvarchar(255)      = NULL,
      @IsActive                      bit                = NULL,
      @Locale                        int                = NULL,
      @CalendarType                  smallint           = NULL,
      @AdjustHijriDays               smallint           = NULL,
      @TimeZone                      smallint           = NULL,
      @Collation                     smallint           = NULL,
      @Time24                        bit                = NULL,
      @AltCalendarType               tinyint            = NULL,
      @CalendarViewOptions           tinyint            = NULL,
      @WorkDays                      smallint           = NULL,
      @WorkDayStartHour              smallint           = NULL,
      @WorkDayEndHour                smallint           = NULL,
      @MobileNumber                  nvarchar(127)      = NULL,
      @RequestGuid                   uniqueidentifier   = NULL OUTPUT
);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the user whose information is to be updated.

@UserId: The user identifier (section 2.2.1.13) for the user whose information is to be updated.

@Title: The user-friendly display name of the user. If this parameter is NULL, then the display name MUST be set to the empty string.

@Email: The email address of the user. If this parameter is NULL, then the email address MUST be set to the empty string.

@IsActive: A bit flag specifying whether the user is an active user in the site collection. This flag is set to "0" if the user is not an active user. This flag is set to "1" to indicate otherwise. If this flag is NULL, then the value MUST NOT be changed.

@Locale: An LCID specifying the preferred locale settings to be used when formatting and displaying UI for the user.

@CalendarType: The **Calendar** type (section 2.2.3.3) to be used when processing date values for this user.

@AdjustHijriDays: If the *@CalendarType* parameter value is "6", then this parameter specifies the number of days to extend or reduce the current month in Hijri calendars for this user.

@TimeZone: The Time Zone Identifier (section 2.2.3.17) for the time zone to be used when displaying time values for this user.

@Collation: The Collation Order (section 2.2.3.4) to be used when displaying information to this user.

@Time24: A bit flag which specifies whether to use a 24-hour time format when displaying time values to this user. If this parameter is set to "1", then the 24-hour time format is used; otherwise, the 12-hour time format is used.

@AltCalendarType: The **Calendar** type of an alternate calendar for processing date values for this user.

@CalendarViewOptions: A **Calendar View Options** type (section 2.2.4.1) which specifies the calendar display options setting for this user.

@WorkDays: A set of Workdays Flags (section 2.2.2.13) which specify the week days defined as the work week for this user.

@WorkDayStartHour: The start time of the work day for this user, in minutes from 12:00AM. For example, the value 480 indicates 8:00AM.

@WorkDayEndHour: The end time of the work day for this user, in minutes from 12:00AM. For example, the value 1020 indicates 5:00PM.

@MobileNumber: The user-friendly text field that specifies the mobile phone number of the user.

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

Return Values: The **proc_UpdateUserInfoInTableFromRowUpdater** stored procedure returns an integer return code which MUST be 0.

The **proc_UpdateUserInfoInTableFromRowUpdater** stored procedure MUST return no result sets.

3.1.5.134 proc_UrlToWebUrl

The **proc_UrlToWebUrl** stored procedure returns the store-relative form URL of the site that contains a specified store-relative form URL and is inside a specific site collection.

```
PROCEDURE proc_UrlToWebUrl(
      @WebSiteId                      uniqueidentifier,
      @Url                            nvarchar(260),
      @RequestGuid                    uniqueidentifier = NULL OUTPUT
);
```

@WebSiteId: The site collection identifier (section 2.2.1.9) for a site collection that contains the *@Url* parameter value. For successful execution, the site collection identifier MUST match an existing site collection which could hold the *@Url* parameter value. If the identifier does not match an existing site collection, then an integer value of 1168 MUST be returned along with a Web URL Result Set.

@Url: A store-relative form URL.

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

Return Values: The **proc_UrlToWebUrl** stored procedure returns an integer return code which MUST be listed in the following table.

Value	Description
0	Successful execution.
1168	The site collection specified by *@WebSiteId* does not exist or *@Url* is not contained in the specified site collection.

The **proc_UrlToWebUrl** stored procedure MUST return a single result set as follows.

3.1.5.134.1 Web URL Result Set

The Web URL Result Set returns the store-relative form URL of the site containing the *@Url* input parameter value. The Web URL Result Set MUST be returned and MUST contain one row.

```
    {WebUrl}                        nvarchar(256);
```

{WebUrl}: The store-relative form URL of the site that contains the *@Url* parameter value. This MUST be an empty string when the *@Url* parameter is invalid or is not contained in the specified site collection or the specified site collection does not exist. *@Url* is only checked for whether a valid subsite path exists within *@Url*.

3.1.5.135 proc_WriteChunkToAllDocStreams

The **proc_WriteChunkToAllDocStreams** stored procedure establishes new document content associated with the document specified by *@DocId* or appends to already existing document content established by an earlier invocation of **proc_WriteChunkToAllDocStreams**.

```
    PROCEDURE dbo.proc_WriteChunkToAllDocStreams(
        @SiteId   uniqueidentifier,
        @DocId    uniqueidentifier,
        @Offset int,
        @00 varbinary(max) = NULL,
        @01 varbinary(max) = NULL,
        @02 varbinary(max) = NULL,
        @03 varbinary(max) = NULL,
        @04 varbinary(max) = NULL,
        @05 varbinary(max) = NULL,
        @06 varbinary(max) = NULL,
        @07 varbinary(max) = NULL,
        @08 varbinary(max) = NULL,
        @09 varbinary(max) = NULL,
        @0A varbinary(max) = NULL,
        @0B varbinary(max) = NULL,
        @0C varbinary(max) = NULL,
        @0D varbinary(max) = NULL,
        @0E varbinary(max) = NULL,
        @0F varbinary(max) = NULL,
        @10 varbinary(max) = NULL,
        @11 varbinary(max) = NULL,
        @12 varbinary(max) = NULL,
        @13 varbinary(max) = NULL,
        @14 varbinary(max) = NULL,
        @15 varbinary(max) = NULL,
        @16 varbinary(max) = NULL,
        @17 varbinary(max) = NULL,
        @18 varbinary(max) = NULL,
        @19 varbinary(max) = NULL,
        @1A varbinary(max) = NULL,
        @1B varbinary(max) = NULL,
        @1C varbinary(max) = NULL,
        @1D varbinary(max) = NULL,
        @1E varbinary(max) = NULL,
        @1F varbinary(max) = NULL,
        @20 varbinary(max) = NULL,
        @21 varbinary(max) = NULL,
        @22 varbinary(max) = NULL,
        @23 varbinary(max) = NULL,
        @24 varbinary(max) = NULL,
        @25 varbinary(max) = NULL,
        @26 varbinary(max) = NULL,
        @27 varbinary(max) = NULL,
        @28 varbinary(max) = NULL,
        @29 varbinary(max) = NULL,
        @2A varbinary(max) = NULL,
        @2B varbinary(max) = NULL,
        @2C varbinary(max) = NULL,
        @2D varbinary(max) = NULL,
        @2E varbinary(max) = NULL,
        @2F varbinary(max) = NULL,
        @30 varbinary(max) = NULL,
        @31 varbinary(max) = NULL,
        @32 varbinary(max) = NULL,
        @33 varbinary(max) = NULL,
        @34 varbinary(max) = NULL,
        @35 varbinary(max) = NULL,
        @36 varbinary(max) = NULL,
        @37 varbinary(max) = NULL,
        @38 varbinary(max) = NULL,
        @39 varbinary(max) = NULL,
        @3A varbinary(max) = NULL,
        @3B varbinary(max) = NULL,
        @3C varbinary(max) = NULL,
        @3D varbinary(max) = NULL,
        @3E varbinary(max) = NULL,
        @3F varbinary(max) = NULL,
        @40 varbinary(max) = NULL,
        @41 varbinary(max) = NULL,
        @42 varbinary(max) = NULL,
        @43 varbinary(max) = NULL,
        @44 varbinary(max) = NULL,
        @45 varbinary(max) = NULL,
        @46 varbinary(max) = NULL,
        @47 varbinary(max) = NULL,
        @48 varbinary(max) = NULL,
        @49 varbinary(max) = NULL,
        @4A varbinary(max) = NULL,
        @4B varbinary(max) = NULL,
        @4C varbinary(max) = NULL,
        @4D varbinary(max) = NULL,
        @4E varbinary(max) = NULL,
        @4F varbinary(max) = NULL,
        @50 varbinary(max) = NULL,
        @51 varbinary(max) = NULL,
        @52 varbinary(max) = NULL,
        @53 varbinary(max) = NULL,
        @54 varbinary(max) = NULL,
        @55 varbinary(max) = NULL,
        @56 varbinary(max) = NULL,
        @57 varbinary(max) = NULL,
        @58 varbinary(max) = NULL,
        @59 varbinary(max) = NULL,
        @5A varbinary(max) = NULL,
        @5B varbinary(max) = NULL,
        @5C varbinary(max) = NULL,
        @5D varbinary(max) = NULL,
        @5E varbinary(max) = NULL,
        @5F varbinary(max) = NULL,
        @60 varbinary(max) = NULL,
        @61 varbinary(max) = NULL,
        @62 varbinary(max) = NULL,
        @63 varbinary(max) = NULL,
        @64 varbinary(max) = NULL,
        @65 varbinary(max) = NULL,
        @66 varbinary(max) = NULL,
        @67 varbinary(max) = NULL,
        @68 varbinary(max) = NULL,
        @69 varbinary(max) = NULL,
        @6A varbinary(max) = NULL,
        @6B varbinary(max) = NULL,
        @6C varbinary(max) = NULL,
        @6D varbinary(max) = NULL,
        @6E varbinary(max) = NULL,
        @6F varbinary(max) = NULL,
        @70 varbinary(max) = NULL,
        @71 varbinary(max) = NULL,
        @72 varbinary(max) = NULL,
        @73 varbinary(max) = NULL,
        @74 varbinary(max) = NULL,
        @75 varbinary(max) = NULL,
        @76 varbinary(max) = NULL,
        @77 varbinary(max) = NULL,
        @78 varbinary(max) = NULL,
        @79 varbinary(max) = NULL,
        @7A varbinary(max) = NULL,
        @7B varbinary(max) = NULL,
        @7C varbinary(max) = NULL,
        @7D varbinary(max) = NULL,
        @7E varbinary(max) = NULL,
        @7F varbinary(max) = NULL,
        @80 varbinary(max) = NULL,
        @81 varbinary(max) = NULL,
        @82 varbinary(max) = NULL,
        @83 varbinary(max) = NULL,
        @84 varbinary(max) = NULL,
        @85 varbinary(max) = NULL,
        @86 varbinary(max) = NULL,
        @87 varbinary(max) = NULL,
        @88 varbinary(max) = NULL,
        @89 varbinary(max) = NULL,
        @8A varbinary(max) = NULL,
        @8B varbinary(max) = NULL,
        @8C varbinary(max) = NULL,
        @8D varbinary(max) = NULL,
        @8E varbinary(max) = NULL,
        @8F varbinary(max) = NULL,
        @90 varbinary(max) = NULL,
        @91 varbinary(max) = NULL,
        @92 varbinary(max) = NULL,
        @93 varbinary(max) = NULL,
        @94 varbinary(max) = NULL,
        @95 varbinary(max) = NULL,
        @96 varbinary(max) = NULL,
        @97 varbinary(max) = NULL,
        @98 varbinary(max) = NULL,
        @99 varbinary(max) = NULL,
        @9A varbinary(max) = NULL,
        @9B varbinary(max) = NULL,
        @9C varbinary(max) = NULL,
        @9D varbinary(max) = NULL,
        @9E varbinary(max) = NULL,
        @9F varbinary(max) = NULL,
        @A0 varbinary(max) = NULL,
        @A1 varbinary(max) = NULL,
        @A2 varbinary(max) = NULL,
        @A3 varbinary(max) = NULL,
        @A4 varbinary(max) = NULL,
        @A5 varbinary(max) = NULL,
        @A6 varbinary(max) = NULL,
        @A7 varbinary(max) = NULL,
        @A8 varbinary(max) = NULL,
        @A9 varbinary(max) = NULL,
        @AA varbinary(max) = NULL,
        @AB varbinary(max) = NULL,
        @AC varbinary(max) = NULL,
        @AD varbinary(max) = NULL,
        @AE varbinary(max) = NULL,
        @AF varbinary(max) = NULL,
        @B0 varbinary(max) = NULL,
        @B1 varbinary(max) = NULL,
        @B2 varbinary(max) = NULL,
        @B3 varbinary(max) = NULL,
        @B4 varbinary(max) = NULL,
        @B5 varbinary(max) = NULL,
        @B6 varbinary(max) = NULL,
        @B7 varbinary(max) = NULL,
        @B8 varbinary(max) = NULL,
        @B9 varbinary(max) = NULL,
        @BA varbinary(max) = NULL,
        @BB varbinary(max) = NULL,
        @BC varbinary(max) = NULL,
        @BD varbinary(max) = NULL,
        @BE varbinary(max) = NULL,
        @BF varbinary(max) = NULL,
        @C0 varbinary(max) = NULL,
        @C1 varbinary(max) = NULL,
        @C2 varbinary(max) = NULL,
        @C3 varbinary(max) = NULL,
        @C4 varbinary(max) = NULL,
        @C5 varbinary(max) = NULL,
        @C6 varbinary(max) = NULL,
        @C7 varbinary(max) = NULL,
        @C8 varbinary(max) = NULL,
        @C9 varbinary(max) = NULL,
        @CA varbinary(max) = NULL,
        @CB varbinary(max) = NULL,
        @CC varbinary(max) = NULL,
        @CD varbinary(max) = NULL,
        @CE varbinary(max) = NULL,
        @CF varbinary(max) = NULL,
        @D0 varbinary(max) = NULL,
        @D1 varbinary(max) = NULL,
        @D2 varbinary(max) = NULL,
        @D3 varbinary(max) = NULL,
        @D4 varbinary(max) = NULL,
        @D5 varbinary(max) = NULL,
        @D6 varbinary(max) = NULL,
        @D7 varbinary(max) = NULL,
        @D8 varbinary(max) = NULL,
        @D9 varbinary(max) = NULL,
        @DA varbinary(max) = NULL,
        @DB varbinary(max) = NULL,
        @DC varbinary(max) = NULL,
        @DD varbinary(max) = NULL,
        @DE varbinary(max) = NULL,
        @DF varbinary(max) = NULL,
        @E0 varbinary(max) = NULL,
        @E1 varbinary(max) = NULL,
        @E2 varbinary(max) = NULL,
        @E3 varbinary(max) = NULL,
        @E4 varbinary(max) = NULL,
        @E5 varbinary(max) = NULL,
        @E6 varbinary(max) = NULL,
        @E7 varbinary(max) = NULL,
        @E8 varbinary(max) = NULL,
        @E9 varbinary(max) = NULL,
        @EA varbinary(max) = NULL,
        @EB varbinary(max) = NULL,
        @EC varbinary(max) = NULL,
        @ED varbinary(max) = NULL,
        @EE varbinary(max) = NULL,
        @EF varbinary(max) = NULL,
        @F0 varbinary(max) = NULL,
        @F1 varbinary(max) = NULL,
        @F2 varbinary(max) = NULL,
        @F3 varbinary(max) = NULL,
        @F4 varbinary(max) = NULL,
        @F5 varbinary(max) = NULL,
        @F6 varbinary(max) = NULL,
        @F7 varbinary(max) = NULL,
        @F8 varbinary(max) = NULL,
        @F9 varbinary(max) = NULL,
        @FA varbinary(max) = NULL,
        @FB varbinary(max) = NULL,
        @FC varbinary(max) = NULL,
        @FD varbinary(max) = NULL,
        @FE varbinary(max) = NULL,
        @FF varbinary(max) = NULL,
        @RequestGuid uniqueidentifier = NULL OUTPUT)
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the document.

@DocId: The **document identifier** (section 2.2.1.2) of the document.

@Offset: MUST be 0 if no document content corresponds to *@DocId*; otherwise, MUST be the length, in bytes, of the document content which corresponded to *@DocId* before the stored procedure was invoked.

@00 -- @FF : These are 256 optional parameters. The non-NULL parameters, concatenated in order according to their name, interpreted as a hexadecimal number, MUST be stored or appended to the document content. If one of the parameters is NULL, then all subsequent parameters MUST be NULL.

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

Return Values: The **proc_WriteChunkToAllDocStreams** stored procedure an integer return code which MUST be listed in the following table.

Value	Description
0	Successful execution.
29	Unexpected error.

The **proc_WriteChunkToAllDocStreams** stored procedure MUST NOT return a result set.

3.1.5.136 proc_WriteExternalTokenToAllDocStreams

The **proc_WriteExternalTokenToAllDocStreams** stored procedure establishes the token for an external storage system representing new document content to correspond to *@DocId*.

```
PROCEDURE dbo.proc_WriteExternalTokenToAllDocStreams (
@SiteId         uniqueidentifier,
@DocId          uniqueidentifier,
@Token          varbinary(max)
)
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the document.

@DocId: The document identifier (section 2.2.1.2) of the document.

@Token: The token representing the document content in an external storage system. If NULL, then this parameter is equivalent to the token value 0x.

Return Values: The **proc_WriteExternalTokenToAllDocStreams** stored procedure MUST return 0.

The **proc_WriteExternalTokenToAllDocStreams** stored procedure MUST NOT return a result set.

3.1.5.137 proc_WriteRBSTokenToAllDocStreams

The **proc_WriteRBSTokenToAllDocStreams** stored procedure establishes the Remote Blob Storage Identifier representing new document content to correspond to *@DocId*. For additional information regarding remote blob storage, see [MS-WSSO] section 2.1.2.3.8.

```
PROCEDURE dbo.proc_WriteRBSTokenToAllDocStreams(
@SiteId         uniqueidentifier,
@DocId          uniqueidentifier,
@Token          varbinary(64)
)
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site collection containing the document.

@DocId: The **document identifier** (section 2.2.1.2) of the document.

@Token: The remote blob storage identifier for the new document content. For additional information regarding remote blob storage, see [MS-WSSO] section 2.1.2.3.8.

Return Values: The **proc_WriteRBSTokenToAllDocStreams** stored procedure MUST return 0.

The **proc_WriteRBSTokenToAllDocStreams** stored procedure MUST NOT return a result set.

3.1.5.138 TVF_Docs_Url_Level

The **TVF_Docs_Url_Level** table value function is invoked to return a subset of rows from the **Docs View** (section 2.2.7.4).

```
FUNCTION [dbo].[TVF_Docs_Url_Level]
            (
                @SiteId uniqueidentifier,
                @DirName nvarchar(256),
                @LeafName nvarchar(128),
                @Level tinyint
            )
```

@SiteId: All the rows returned from the **Docs view** MUST have **SiteId** = *@SiteId*.

@DirName: All the rows returned from the **Docs view** MUST have **DirName** = *@DirName*.

@LeafName: All the rows returned from the **Docs view** MUST have **LeafName** = *@LeafName*.

@Level: All the rows returned from the **Docs view** MUST have **Level** = *@Level*.

Return Values: TVF_Docs_Url_Level returns all rows from the **Docs view** that are specified as follows.

```
(SiteId = @SiteId AND DirName = @DirName AND LeafName = @LeafName AND Level = @Level.)
```

3.1.5.139 TVF_UserData_ListItemLevelRow

The **TVF_UserData_ListItemLevelRow** table value function is invoked to return a subset of rows from the **UserData** view (section 2.2.7.8).

```
FUNCTION [dbo].[TVF_UserData_ListItemLevelRow]
            (
                @tp_ListId uniqueidentifier,
                @tp_Id int,
                @tp_Level tinyint,
                @tp_RowOrdinal tinyint
            )
```

@tp_ListId: All the rows returned from the **UserData** view MUST have **tp_ListId** = *@tp_ListId*.

@tp_Id: All the rows return from the **UserData** view MUST have **tp_Id** = *@ tp_Id*.

@tp_Level: All the rows return from the **UserData** view MUST have **tp_Level** = *@tp_Level*.

@tp_RowOrdinal: All the rows returned from the **UserData** view MUST have **tp_RowOrdinal** = *@tp_RowOrdinal*.

Return Values: TVF_UserData_ListItemLevelRow returns all rows from the **UserData** view that are specified as follows.

```
 (tp_ListId = @tp_ListId AND tp_Id = @tp_Id  AND tp_Level = @tp_Level AND  tp_RowOrdinal = 
@tp_RowOrdinal)
```

3.1.5.140 TVF_UserData_PId_DId_Level_Row

The **TVF_UserData_PId_DId_Level_Row** table value function is invoked to return a subset of rows from the **UserData** view (section 2.2.7.8).

```
FUNCTION [dbo].[TVF_UserData_PId_DId_Level_Row]
            (
                @tp_SiteId uniqueidentifier,
                @tp_ParentId uniqueidentifier,
                @tp_DocId uniqueidentifier,
                @tp_Level tinyint,
                @tp_RowOrdinal tinyint
            )
```

@tp_SiteId: All the rows returned from the **UserData** view MUST have **tp_SiteId** = *@tp_SiteId*.

@tp_ParentId: All the rows returned from the **UserData** view MUST have **tp_ParentId** = *@tp_ParentId*.

@tp_DocId: All the rows returned from the **UserData** view MUST have **tp_DocId** = *@tp_DocId*.

@tp_Level: All the rows returned from the **UserData** view MUST have **tp_Level** = *@tp_Level*.

@tp_RowOrdinal: All the rows returned from the **UserData** view MUST have **tp_RowOrdinal** = *@tp_RowOrdinal*.

Return Values: TVF_UserData_PId_DId_Level_Row returns all rows from the **UserData** view that are specified as follows.

```
(tp_SiteId = @tp_SiteId AND tp_ParentId = @tp_ParentId AND tp_DocId = @tp_DocId AND tp_Level
= @tp_Level AND    tp_RowOrdinal = @tp_RowOrdinal)
```

3.1.5.141 proc_HasCurrentPublishVersion

The **proc_HasCurrentPublishVersion** stored procedure is invoked to test whether the specified list item has a current published version.

```
PROCEDURE proc_HasCurrentPublishVersion (
    @ListId         uniqueidentifier,
    @ItemId         int,
    @RequestGuid    uniqueidentifier = null OUTPUT
);
```

@ListId: The list identifier (section 2.2.1.5) of the list that contains the specified list item.

@ItemId: The list item identifier (section 2.2.1.6) of the specified list item.

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

Return values: This stored procedure returns an integer return code, which MUST be included in the following table.

Value	Description
0	The specified list item does not have a current published version.
1	The specified list item has a current published version.

The **proc_HasCurrentPublishVersion** stored procedure MUST NOT return a result set.

3.1.5.142 proc_TransferStreamVersion

The **proc_TransferStreamVersion** stored procedure overwrites the document content corresponding to *@NewDocId* and *@UIVersion* with the document content corresponding to *@TempDocId*. It MUST NOT write document content for *@NewDocId* if no document corresponding to *@NewDocId* and *@UIVersion* existed before the stored procedure was invoked. The stored procedure MUST delete the document content corresponding to *@TempDocId* regardless of whether a document corresponding to *@NewDocId* and *@UIVersion* existed. The document content corresponding to *@TempDocId* MUST have been established by **proc_WriteChunkToAllDocStreams** (section 3.1.5.135), **proc_WriteExternalTokenToAllDocStreams** (section 3.1.5.136),or **proc_WriteRBSTokenToAllDocStreams** (section 3.1.5.137) prior to invoking this stored procedure.

```
PROCEDURE dbo.proc_TransferStreamVersion(
    @SiteId uniqueidentifier,
    @NewDocId uniqueidentifier,
    @UIVersion int,
    @TempDocId uniqueidentifier,
    @DocSize int,
    @IsExternal bit);
```

@SiteId: The site collection identifier (section 2.2.1.9) of the site containing the document.

@NewDocId: The **document identifier** (section 2.2.1.2) of the document to be overwritten.

@UIVersion: The UI version number associated with this document. This MUST NOT be NULL.

@TempDocId: The document identifier corresponding to the source document content.

@DocSize: The size in bytes of the document content corresponding to *@TempDocId*.

@IsExternal: If *@TempDocId* was established by **proc_WriteExternalTokenToAllDocStreams**, this MUST be 1; otherwise, it MUST be 0.

Return Values: The **proc_TransferStreamVersion** stored procedure returns 0.

The **proc_TransferStreamVersion** stored procedure MUST NOT return a result set.

3.1.6 Timer Events

If the execution timeout event is triggered, then the execution of the stored procedure is terminated and the call fails.

3.1.7 Other Local Events

None.

3.2 Web Front End Client Details

The front-end web server acts as a client when it calls the back-end database server requesting 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, but 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 completed 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 procedure(s). The site collection identifier (section 2.2.1.9) and the user identifier (section 2.2.1.13) for the user making the request are looked up by the front-end web server before calling additional stored procedure(s).

3.2.4 Higher-Layer Triggered Events

None.

3.2.5 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 result 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. However, unless otherwise specified, any data addition, removal, or modification MUST occur only by calling the listed stored procedure. SQL queries MUST NOT attempt to add, remove, or update data in any table or view in the content databases or Configuration Database (section 3.1.1), unless explicitly described in this section.

3.2.6 Timer Events

If the connection timeout event is triggered, the connection and the stored procedure call fails.

3.2.7 Other Local Events

None.

4 Protocol Examples

This section provides specific example scenarios for end-to-end file and permissions management against WSS. These examples describe in detail the process of communication between the various server components involved in the WSS deployment. In conjunction with the detailed protocol documentation described in the reference documents, this information is intended to provide a comprehensive view of how WSS front-end web server communicates with both EUC and BEDS systems.

4.1 File: Open File OM

This example describes the requests and responses made when the **WFE** opens an existing file stored as a document in a document library on the BEDS with a SharePoint Object Model call.

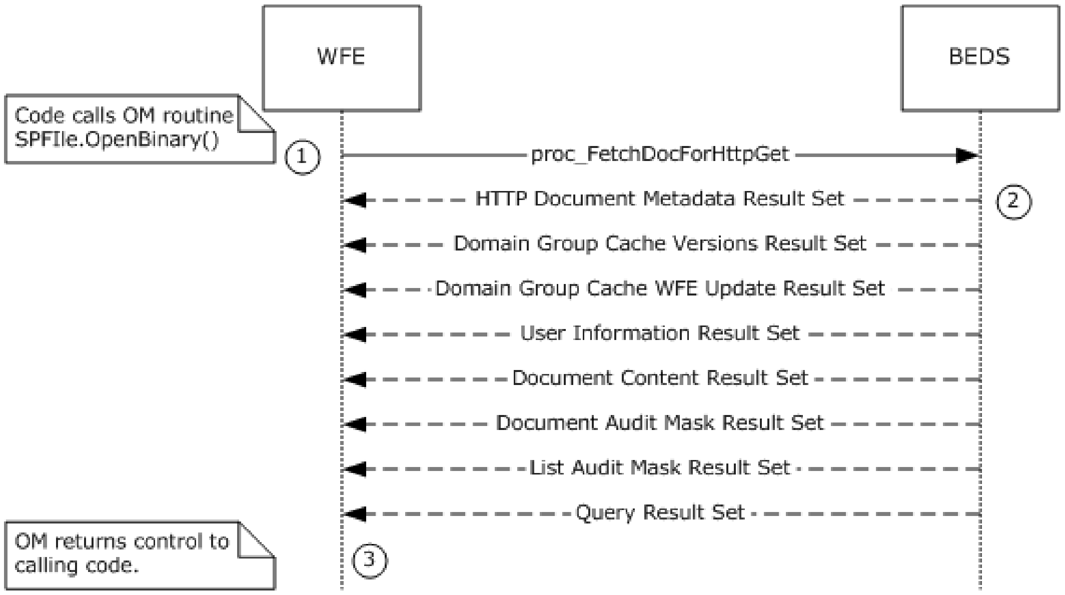
WFE
BEDS
Code calls OM routine SPFIle.OpenBinary()
1
proc_FetchDocForHttpGet
2
HTTP Document Metadata Result Set
Domain Group Cache Versions Result Set
Domain Group Cache WFE Update Result Set
User Information Result Set
Document Content Result Set
Document Audit Mask Result Set
List Audit Mask Result Set
Query Result Set
OM returns control to calling code.
3

Figure 1: Open File OM

This scenario is initiated by a call to the **Microsoft.SharePoint.SPFile.OpenBinary()** object model command. For simplicity's sake, this example assumes that the file is stored as a document in a document library, and that the requested version is a draft created by the same user who is opening the file. This example assumes that:

- The code has already instantiated the site collection (SPSite), site (SPWeb), and document library (SPList) objects containing the document to be opened.
- Auditing is disabled for the site collection.
- The current user has File Open permissions for the document.
- Site **groups** in the site collection do not include any domain groups as members.

The following actions happen:

1. The WFE builds a dynamic query that invokes the **proc_FetchDocForHttpGet** stored procedure.
2. The BEDS returns a return code of 0, and returns the following result sets:

- HTTP Document Metadata Result Set (section 3.1.5.19.1). This returns the document metadata needed to further process the document.
- Domain Group Cache Versions Result Set (section 2.2.5.4). The versions of the Domain Group cache on BEDS and WFE, used to determine if either the BEDS or WFE has more up-to-date information about external group membership in Roles, which are stored as site groups.
- Domain Group Cache WFE Update Result Set (section 2.2.5.5). Used to update the WFE's external group map cache if needed. Under our assumptions, this result set is empty, because no domain groups are members of any roles in the site collection.
- User Information Result Set (section 3.1.5.19.5). Used to establish that the current user has permissions to open the file.
- Document Content Stream Result Set (section 3.1.5.19.8). Includes the document stream containing the binary file content for the current version of the document visible to the user, along with additional document metadata.
- Site Audit Mask Result Set (section 2.2.5.20). Under our assumptions, auditing is not enabled on the site collection, so the **SiteGlobalAuditMask** column is NULL.
- List Audit Mask Result Set (section 3.1.5.19.10), containing auditing information for the document's containing document library. Under our assumptions, auditing is not enabled for the site collection, so the fields containing audit masks are NULL.
- A Dynamic Query Result Set containing a single row with a single unnamed column, holding the value of the *@Level* output parameter from the stored procedure **proc_FetchDocForHttpGet**. The value is 2, indicating that the content returned is from the latest draft version.

3. The OM returns control to the calling program with the array of bytes for the document stream of the requested file.

4.1.1 Determine a User's Permission Level to a Document

This example describes the requests made to determine a user's permissions on a document in a document library.

This example assumes:

- The user is authenticated and not anonymous.
- The user is a **member** of a role defined on the site.
- The document exists in a document library.

This scenario is initiated by a call to the object model command **SPListItem**.**EffectiveBasePermissions()**.

The following actions happen:

1. The WFE builds a dynamic query that invokes the proc_FetchDocForHttpGet stored procedure.
2. The BEDS returns a return code of 0, and returns a number of result sets, including the HTTP Document Metadata Result Set (section 3.1.5.19.1). The HTTP Document Metadata Result Set contains an ACL that specifies the permissions on this document.
3. The OM returns control to the calling program with the SPBasePermissions object containing the permissions the current user has on the requested document.

4.2 Group Add User to Site Group OM

This example describes the requests made when a user is added to a site **group** using SharePoint Object Model code running on the WFE.

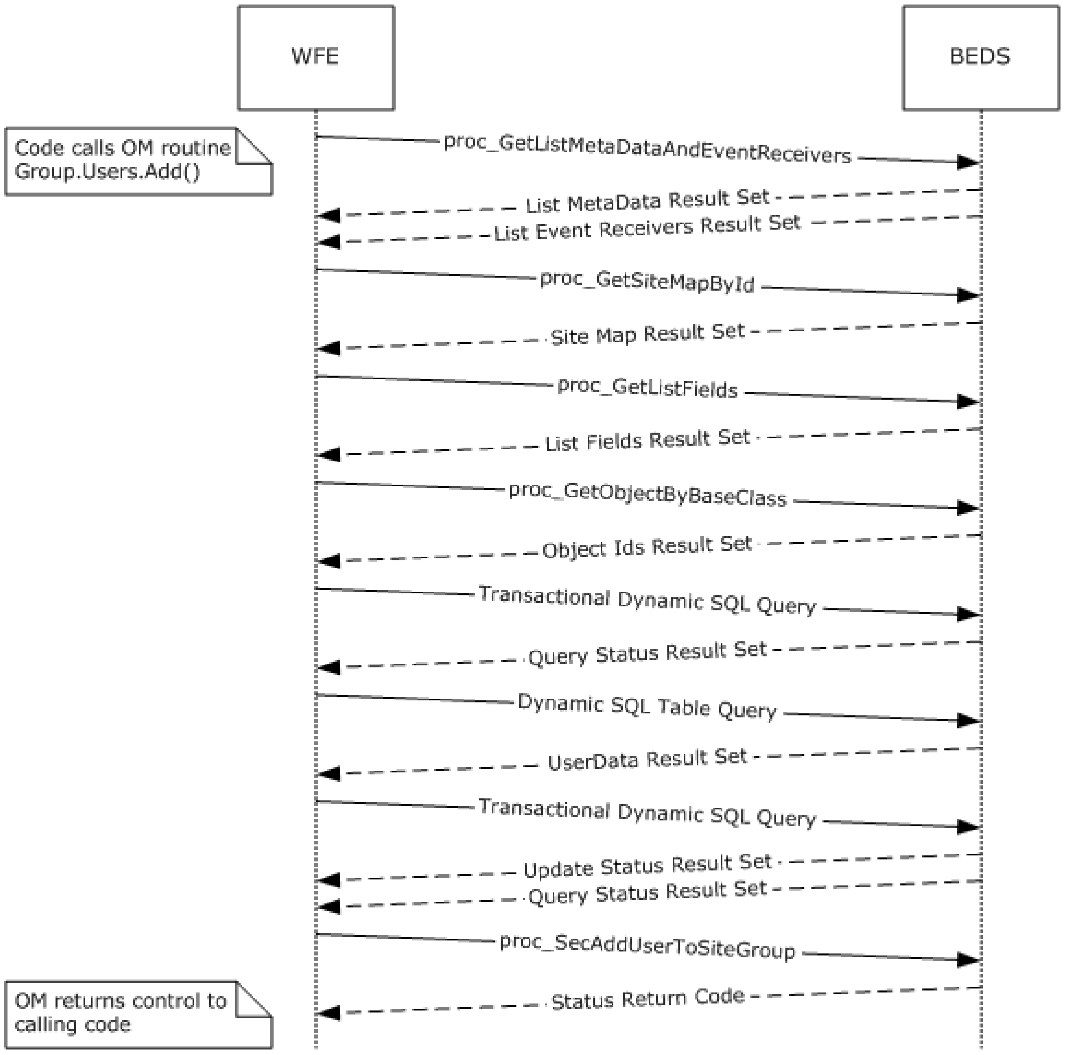

Figure 2: Group Add user to site Group OM

This scenario is initiated by a call to the object model command **SPGroup.Users.Add()**. For simplicity's sake, this example assumes that:

- The code has already instantiated the site collection (**SPSite**), Web (**SPWeb**), and Group (**SPGroup**) objects, which contain the site group for this session.
- The instantiated objects have not yet populated information about the user information list.
- The user to be added to the group is currently a user in the site collection.

The following actions happen**:**

1. The WFE first fetches the properties for the "User Information List" of the target site collection, a SharePoint list that contains information about users and groups registered in a site collection. It

does this by calling the stored procedure **proc_GetListMetaDataAndEventReceivers** (section 3.1.5.34) using TDS.

2. The BEDS returns two result sets, which include the List Metadata (section 2.2.5.12) and Event Receivers (section 2.2.5.9) for the specified user information list.

3. The WFE determines the site map for the site collection by calling the Configuration Database (section 3.1.1) stored procedure **proc_getSiteMapById** (section 3.1.5.41) using TDS.

4. The BEDS returns a single Site Map By Id Result Set (section 3.1.5.41.1), which includes the site map for the specified site collection.

5. If the WFE determines that the user information list has not been populated in the current **SPSite** object, it requests the list field information for the site collection's user information list by calling the **proc_GetListFields** stored procedure (section 3.1.5.33) using TDS.

6. The BEDS returns a single Fields Information Result Set (section 3.1.5.33.1), which includes the field information for the specified user information list.

7. If the WFE determines that it needs to populate the list of **SPFeatures**, the list of child objects that inherit from the base class SPFeatureDefinition will be populated by calling the Configuration Database (section 3.1.1) stored procedure **proc_getObjectsByBaseClass** (section 3.1.5.37) using TDS.

8. The BEDS returns a single Object ID Result Set (section 3.1.5.37.1), which includes a list of child object identifiers for the specified base class and parent object.

9. The WFE builds a transactional dynamic SQL query to add the user to the site collection and add the user's property information to the user information list. This query is sent to the SQL server using TDS. On the SQL server the following actions occur:

    1. The query begins a new SQL transaction.

    2. The query attempts to add the user to the site collection using the stored procedure **proc_SecAddUser** (section 3.1.5.53).

    3. The query checks if the user exists in the site collection's user information list.

    4. If the user is not found in the site collection's user information list, the query attempts to add the user's properties to the site collection's user information list using the stored procedure **proc_AddListItem** (section 3.1.5.4).

    5. The query rolls back the SQL transaction if the previous procedures were not successful, or it commits the transaction if they were successful.

10. The BEDS returns a single result set indicating the status of the actions within the query and the output parameters from the **proc_SecAddUser** command.

11. The WFE queries the list of users in the user information list for the site collection by building a SQL Batch call to the **UserData** view (section 2.2.7.8), which is then sent to the SQL server using TDS.

12. The BEDS returns a single result set with a list of items in the **UserData** view.

13. The WFE builds a transactional dynamic SQL query to add or update the list item that represents the user with the user's properties. This query is sent to the SQL server using TDS. On the SQL server the following actions occur:

    1. The query begins a new SQL transaction.

    2. The query checks if the user exists in the user information list for the site collection.

   3. If the user is not found in the user information list for the site collection, then the query attempts to add the user's properties to the site collection's user information list using the stored procedure **proc_AddListItem**.

   4. Otherwise, if the user is found in the site collection's user information list, the query attempts to update the user's properties in the site collection's user information list using the stored procedure **proc_UpdateListItem** (section 3.1.5.131).

   5. The query rolls back the SQL transaction if the previous procedures were not successful, or it commits the transaction if they were successful.

14. The BEDS returns two result sets, which contain the output and return codes from the **Add List Item** or **Update List Item** commands.

15. The WFE adds the user to the specified site group by calling the stored procedure **proc_SecAddUserToSiteGroup** (section 3.1.5.54) using TDS.

16. The BEDS supplies a return code indicating success or failure of the procedure.

4.3 Group Update Site Group Properties OM

This example describes the interactions made when properties are updated for a particular site **group**.

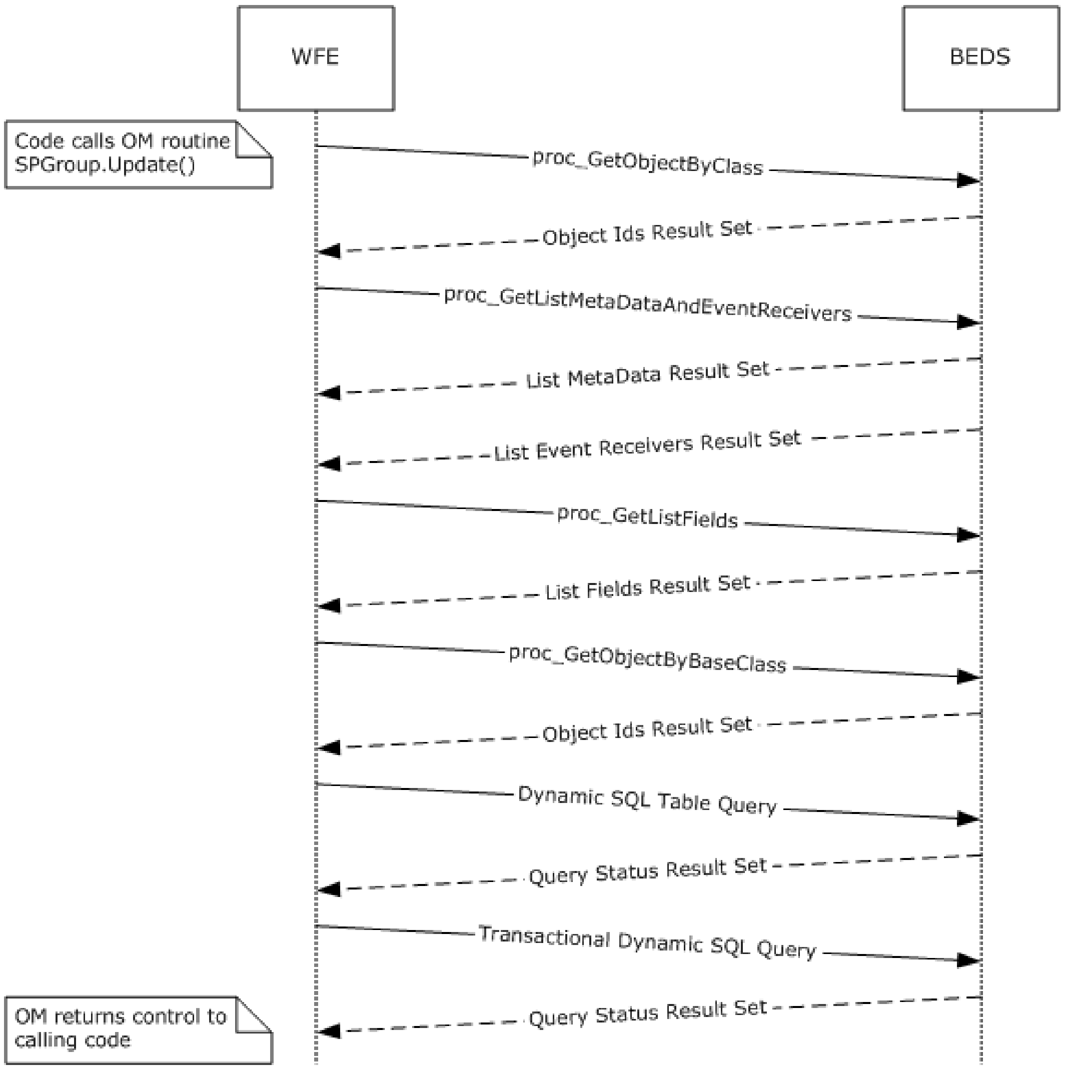

Figure 3: Group Update site Group Properties OM

This scenario is initiated by a call to the object model command **SPGroup.Update()**. For simplicity's sake, this example assumes that:

- The code has already instantiated the site collection (**SPSite**) and site (**SPWeb**) objects for this session.
- The site group to be updated is in the site collection and is a **member** of the site.

The following actions happen:

1. The WFE determines if the site has the directory management service enabled with a call to the Configuration Database (section 3.1.1) stored procedure **proc_GetObjectsByClass** (section 3.1.5.38) using TDS.
2. The BEDS returns a single **Object ID Result Set** (section 3.1.5.38.1) row, which includes a value set if the directory management service is enabled.
3. The WFE fetches the properties for the target site collection's user information list, a SharePoint list containing information about users and groups registered in a site collection. It does this by

calling the stored procedure **proc_GetListMetaDataAndEventReceivers** (section 3.1.5.34) using TDS.

4. The BEDS returns two result sets, which include the **List Metadata** (section 2.2.5.12) and **Event Receivers** (section 2.2.5.9) for the specified user information list.
5. If the WFE determines that the user information list has not been populated in the current **SPSite** object, it requests the list field information for the site collection's user information list by calling the **proc_GetListFields** stored procedure (section 3.1.5.33) using TDS.
6. The BEDS returns a single Field Information Result Set (section 3.1.5.33.1), which includes the field information for the specified user information list.
7. If the WFE determines that it needs to populate the list of **SPFeatures**, the list of child objects that inherit from the base class **SPFeatureDefinition** is populated by calling the Configuration Database (section 3.1.1) stored procedure **proc_getObjectsByBaseClass** (section 3.1.5.37) using TDS.
8. The BEDS returns a single object ID result set, which includes a list of child object identifiers for the specified base class and parent object.
9. The WFE builds a dynamic SQL query to select existing information for the site group using TDS.
10. The BEDS returns a single result set, which includes existing data for the user.
11. The WFE builds a transactional dynamic SQL query to update the site group information in the site collection. This query is sent to the SQL server using TDS. On the SQL server the following actions occur:
    1. The query begins a new SQL transaction.
    2. The query attempts to load the site group information using the stored procedure **proc_SecSetSiteGroupProperties** (section 3.1.5.117).
    3. If the site group is not in the list of site members, the query invokes **proc_AddListItem** (section 3.1.5.4) with the updated information.
    4. Otherwise, the query attempts to update the site group's properties in the site collection user information list, using the stored procedure **proc_UpdateListItem** (section 3.1.5.131).
    5. The query rolls back the SQL transaction if the previous procedures were not successful, or it commits the transaction if they were successful.
12. The BEDS returns a single result set, which indicates the return code status of the actions within the query.

4.4 Security: Add User to Document Library via Object Model

This example describes the requests made when a user is added to the "contributor" role of a document library that has its own scope for independently managed permissions.

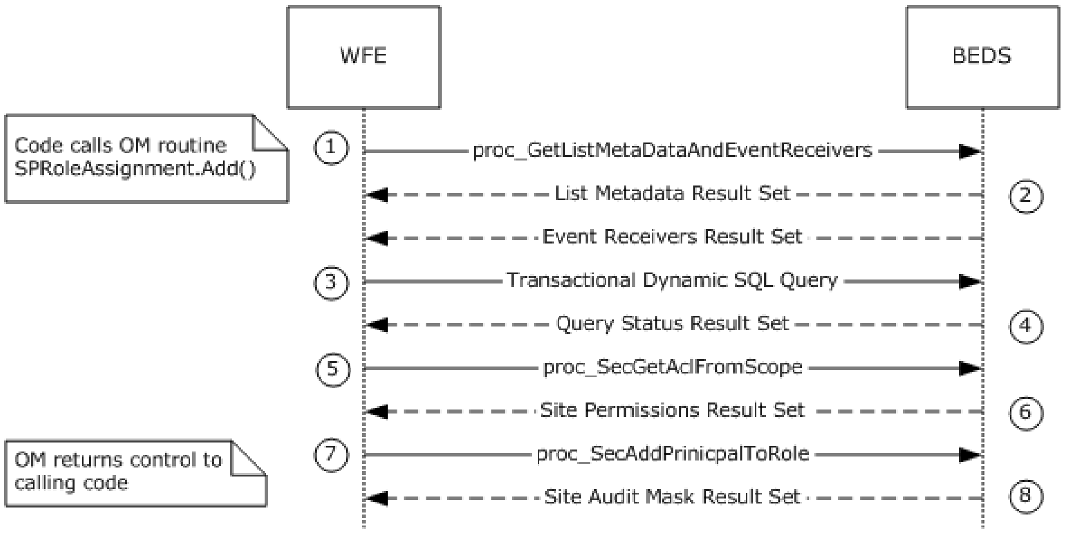
WFE
BEDS
Code calls OM routine SPRoleAssignment.Add()
1
proc_GetListMetaDataAndEventReceivers
List Metadata Result Set
2
Event Receivers Result Set
3
Transactional Dynamic SQL Query
Query Status Result Set
4
5
proc_SecGetAclFromScope
Site Permissions Result Set
6
OM returns control to calling code
7
proc_SecAddPrinicpalToRole
Site Audit Mask Result Set
8

Figure 4: Add user to Document Library via Object Model

This scenario is initiated by a call to the object model command **SPRoleAssignmentCollection.Add(SPRoleAssignment)**.

For simplicity's sake, this example assumes that the code has already instantiated the necessary site collection (**SPSite**), site (**SPWeb**), and list (**SPList**) objects, as well as the role (**SPRoleDefinition**), role bindings (**SPRoleDefinitionBindingCollection**) and role assignment (**SPRoleAssignment**) objects, in order to construct a representation of the role within the document library to which the user will be added.

1. The WFE first fetches the properties for the user information list of the site collection. The user information list is a SharePoint list containing information about users in the site collection. It does this by calling the stored procedure **proc_GetListMetaDataAndEventReceivers** (section 3.1.5.34) using TDS.

2. The BEDS returns two result sets. The List Metadata Result Set (section 2.2.5.12) returns a single row of data with the metadata for the list. The second result set is the Event Receivers Result Set (section 2.2.5.9). In this example, there are no registered event receivers, and so zero rows are returned.

3. The WFE builds a transactional dynamic SQL query to add the user to the document library's "Contributor" role and to add or update the user's property information in the user information list. This query is sent to the SQL server using TDS. On the SQL server, the following actions occur:

    1. The query begins a new SQL transaction.

    2. The query attempts to add the user to the site collection's user information list using the stored procedure **proc_SecAddUser** (section 3.1.5.53).

    3. The query checks if the user exists in the site collection's user information list.

    4. If the user is not found in the site collection's user information list, then the query attempts to add the user's properties to the site collection's user information list using the stored procedure **proc_AddListItem** (section 3.1.5.4).

    5. The query rolls back the SQL transaction if the previous procedures were not successful, or it commits the transaction if they were successful.

4. The BEDS returns a single result set indicating the status of the actions within the query and the output parameters from the **proc_SecAddUser** stored procedure.

5. The WFE fetches the list's scope ACL and anonymous user permission information by calling the stored procedure **proc_SecGetAclFromScope** (section 3.1.5.64) using TDS.

6. The BEDS returns an ACL and Permission Result Set (section 2.2.5.1), which lists ACL and permission information for the list's scope.

7. The WFE adds the user to the list's "contributor" role assignment membership site **group** by calling the stored procedure **proc_SecAddPrincipalToRole** (section 3.1.5.51) using TDS.

8. The BEDS returns a Site Audit Mask Result Set (section 3.1.5.51.1), which lists Auditing Flags information for the site collection and document library.

4.5 Security: Break Web Inheritance OM

This example describes the requests made to create unique security role assignments for a site, rather than inheriting them from a parent.

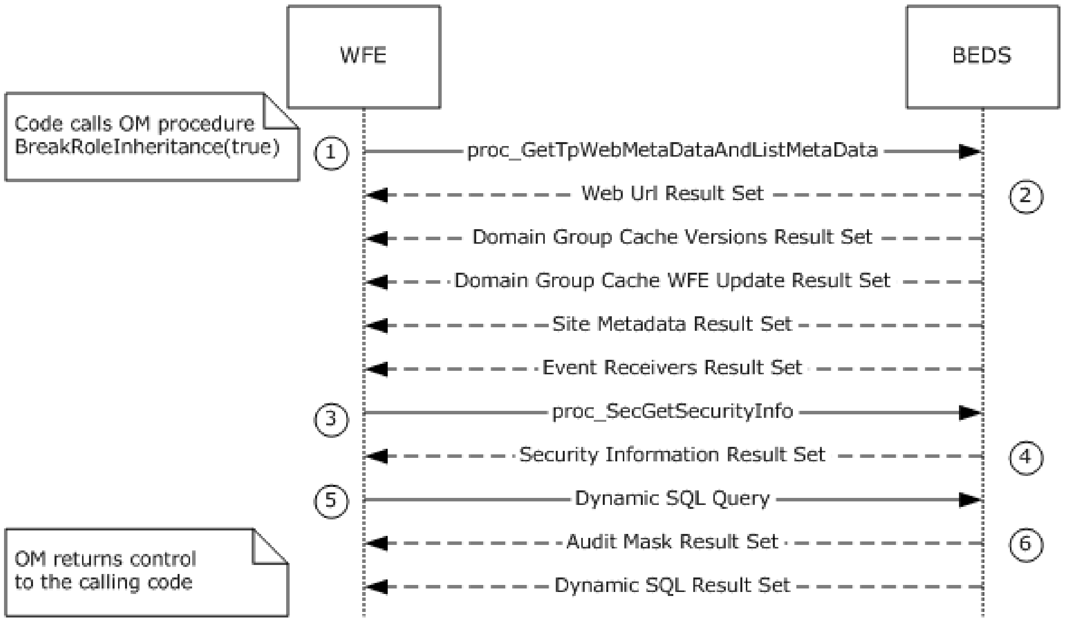
WFE
BEDS
Code calls OM procedure BreakRoleInheritance(true)
1
proc_GetTpWebMetaDataAndListMetaData
Web Url Result Set
2
Domain Group Cache Versions Result Set
Domain Group Cache WFE Update Result Set
Site Metadata Result Set
Event Receivers Result Set
3
proc_SecGetSecurityInfo
Security Information Result Set
4
5
Dynamic SQL Query
OM returns control to the calling code
Audit Mask Result Set
6
Dynamic SQL Result Set

Figure 5: Break Web Inheritance OM

This scenario is initiated by a call to the object model command **spweb.BreakRoleInheritance(true)**. For simplicity's sake, this example assumes that:

- The code has already instantiated the site collection (**SPSite**), the parent site and child subsite (**SPWeb**) objects for this session, and

- The child subsite is initially in the same scope as its parent site.

The following actions happen:

1. The WFE first retrieves metadata for the requested child site. It does this by calling the **proc_GetTpWebMetaDataAndListMetaData** (section 3.1.5.42) stored procedure using TDS.

2. The BEDS returns the following five result sets:

    - Web URL Result Set (section 3.1.5.42.1). This contains the store-relative form URL of the root of the requested child site.

- Domain Group Cache Versions Result Set (section 2.2.5.4). This contains information about the version numbers associated with the External Group Map Cache for the requested site.
- Domain Group Cache WFE Update Result Set (section 2.2.5.5). This returns the binary data needed to refresh the External Group Map Cache.
- Site Metadata Result Set (section 2.2.5.22). This contains metadata for the requested site.
- Event Receivers Result Set (section 2.2.5.9). This contains information about the event receivers defined for the requested site.

3. The WFE then retrieves security permissions information about the requested site. It does this by calling the **proc_SecGetSecurityInfo** (section 3.1.5.85) stored procedure using TDS.
4. The BEDS returns the Security Information Result Set (section 3.1.5.85.1), which consists of information about security permissions about the requested site.
5. The WFE then builds a Dynamic SQL Query to convert the requested site to use unique permissions (as opposed to inheriting those permission from the parent site). It does this by calling the **proc_SecChangeToUniqueScope** stored procedure (section 3.1.5.58) using TDS.
6. The BEDS returns the following two result sets:
   - SiteAudit Mask Result Set (section 2.2.5.20), containing information about the Audit Flags (section 2.2.2.1) set for the requested child site.
   - A Dynamic SQL Result Set, containing the scope identifier (section 2.2.1.8) of the new scope generated for the child site.

4.6 Site Collection Lookup

To allow SharePoint's data storage to scale out, site collections can be stored in many content databases. This example illustrates the protocol operations between the client (front-end web server) and server (back-end database server) needed to find and connect to a specific content database given a site collection URL. An existing connection to the Configuration Database (section 3.1.1) using lower-level protocols is assumed.

4.6.1 Retrieving the Farm Id

The example begins by calling **proc_GetObjectsByClass** (section 3.1.5.38) with the Farm Class ID.

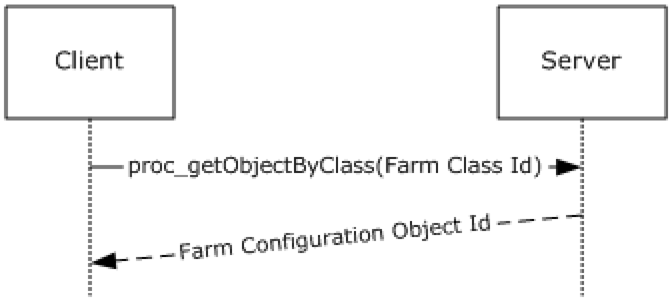
Client
Server
proc_getObjectByClass(Farm Class Id)
Farm Configuration Object Id

Figure 6: Retrieving the Farm ID

This call returns a result set including the Configuration Object ID (section 2.2.6.1.2) of the Farm Configuration Object. **proc_getObjectsByClass** can return result sets with multiple rows, but this implementation of the protocol only ever stores one Configuration Object (section 2.2.6.1) with the Farm Class ID and ignores all but the first row if multiple rows are returned.

4.6.2 Retrieving the Alternate URL Collection Ids

Next, the Farm Configuration Object Id and the Alternate URL Collection Class Id are passed to **proc_getObjectsByBaseClass** (section 3.1.5.37).

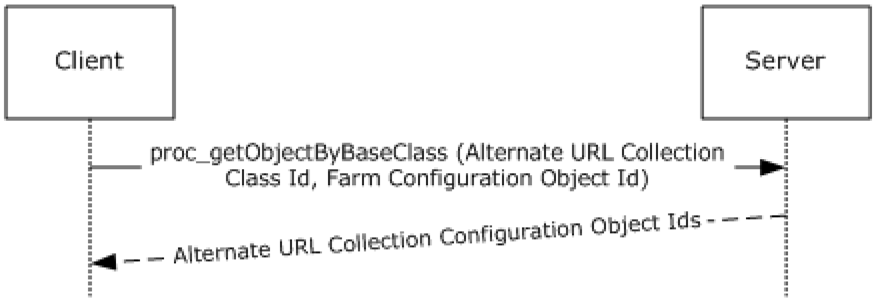
Client
Server
proc_getObjectByBaseClass (Alternate URL Collection Class Id, Farm Configuration Object Id)
Alternate URL Collection Configuration Object Ids

Figure 7: Web Service Lookup Operations

Because the implementation defined the returned Farm Configuration Object to be the parent of all Alternate URL Collections, this call returns the Configuration Object (section 2.2.6.1) **Id**s of all Alternate URL Collections stored in the Configuration Database (section 3.1.1).

4.6.3 Retrieving the Alternate URL Collections

The retrieved Alternate URL Collection Configuration Object Ids are then passed to **proc_getObject** (section 3.1.5.36) to retrieve the full contents of the Alternate URL Collection Configuration Objects.

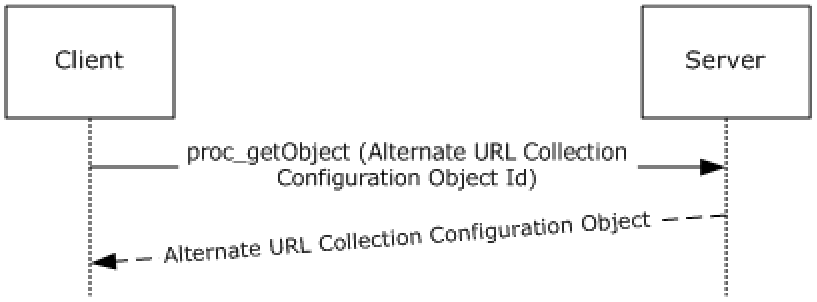
Client
Server
proc_getObject (Alternate URL Collection Configuration Object Id)
Alternate URL Collection Configuration Object

Figure 8: Retrieving the Alternate URL Collections

4.6.4 Alternate URL Matching

At this point, the client can execute the XPath query against the properties of each of the Alternate URL Collections (section 4.6.3) returned to extract all of the alternate URLs. Each of these URLs is then compared against the portion of the incoming URL beginning with the scheme component and ending with the authority component (i.e. "http://example.com:80"). If a match is found the Configuration Object (section 2.2.6.1) **ID** of the Alternate URL Collection containing the matching URL is stored for later use as the request Alternate URL Collections. Otherwise, the site collection lookup operation terminates.

4.6.5 Retrieving the Web Service Ids

Next, the Farm Configuration Object Id returned and the specified Web Service Class Id are passed to **proc_getObjectsByBaseClass**.

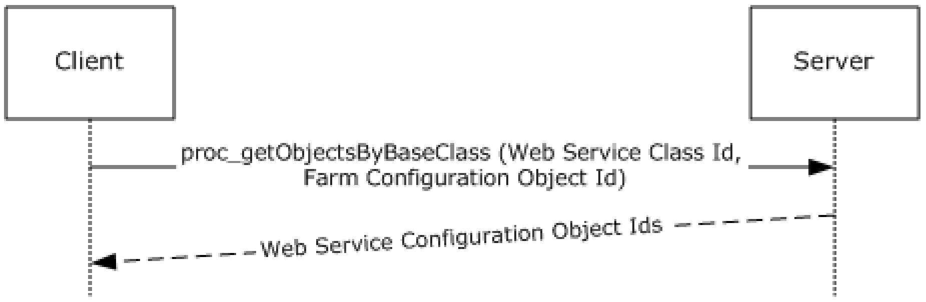
Client
Server
proc_getObjectsByBaseClass (Web Service Class Id, Farm Configuration Object Id)
Web Service Configuration Object Ids

Figure 9: Web Service Lookup Operations

Because the implementation defined the Farm Configuration Object to be the Parent of all Web Services, this call returns the Configuration Object Ids of all Web Services stored in the Configuration Database (section 3.1.1).

4.6.6 Retrieving the Web Application Ids

In this step, the specified **web application** Class Id is passed to **proc_getObjectsByBaseClass** (section 3.1.5.37) with each of the retrieved Web Service Configuration Object Ids. This returns the Configuration Object Ids of all web applications in the Configuration Database (section 3.1.1).

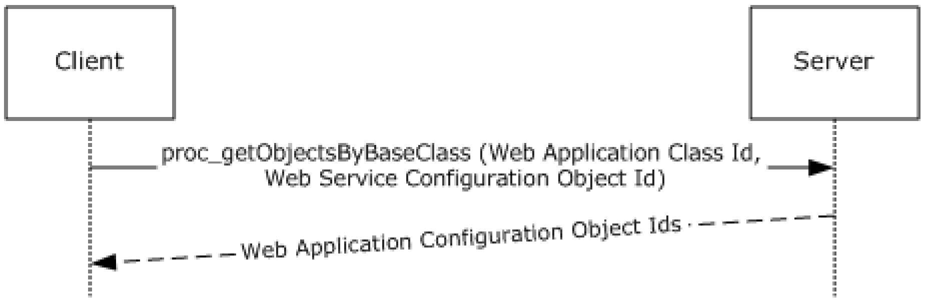
Client
Server
proc_getObjectsByBaseClass (Web Application Class Id, Web Service Configuration Object Id)
Web Application Configuration Object Ids

Figure 10: Retrieving the Web Application Ids

4.6.7 Retrieving the Web Applications

The retrieved Web Application Configuration Object Ids are then passed to **proc_getObject** (section 3.1.5.36) to retrieve the full contents of the Web Application Configuration Objects.

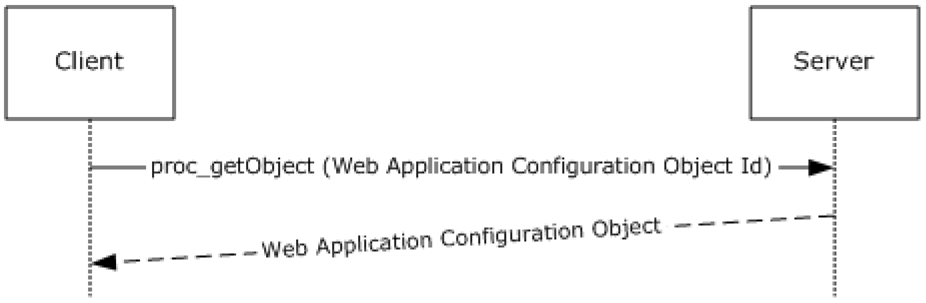
Client
Server
proc_getObject (Web Application Configuration Object Id)
Web Application Configuration Object

Figure 11: Retrieving the Web Applications

4.6.8 Web Application Lookup

At this point, the client can execute the Web Application Alternate URL Collection XPath Query to extract the Alternate URL Collection Ids (section 2.2.6.1.7.1) associated with each retrieved web application. These Alternate URL Collection IDs are compared against the Request Alternate URL Collection ID. If a match is found, the associated web application is used as the Request Web Application for the remainder of site collection lookup. Otherwise, site collection lookup terminates.

4.6.9 Prefix Matching

Web applications contain a set of site collection Prefixes which contain a name and a type. Site collection Lookup extracts these values from the Request Web Application Properties using XPath Queries.

The Prefix names are URL Path Components used to determine which portion of the incoming URL Path Component is the server-relative URL of the site collection. This is done by matching all of the Prefixes in the Request Web Application against the start of the Path Component of the incoming URL. If more than one Prefix matches the beginning of the incoming URL Path Component, the longest matching Prefix is used.

There are two types of Prefix: wildcard and explicit. A **web application** can contain any combination of both types.

4.6.9.1 Explicit Prefixes

An explicit prefix indicates that the portion of the Path Component up to and including the Prefix are included in the site collection server-relative URL. For example, if a user requests http://example.com/sitename/web/list/document.htm and if the **web application** corresponding to http://example.com contains an explicit prefix named "sitename", then "/sitename" is the server-relative URL of the site collection.

Incoming URL	Web Application Explicit Prefixes	Resulting site Collection Server-Relative URL
http://example.com/a/b/c.htm	"a"	"/a"
http://example.com/a/b/c.htm	"a", "a/b"	"/a/b"
http://example.com/a/b.htm	"a", "a/b"	"/a"
http://example.com/a/b.htm	"c"	<No Match>
http://example.com/a/b.htm	""	"/"

4.6.9.2 Wildcard Prefixes

A wildcard prefix indicates that the portion of the Path Component up to and including the first Path Component segment following the Prefix are included in the site collection name. For example, if a user makes a request for http://example.com/sites/sitename/web/list/document.htm, and if the **web application** corresponding to http://example.com contains a Wildcard Prefix named "sites", then "/sites/sitename" is the server-relative URL of the site collection.

Incoming URL	Web Application Wildcard Prefixes	Resulting site Collection Server-Relative URL
http://example.com/a/b/c/d.htm	"a", "a/b	"/a/b/c"
http://example.com/a/b.htm	"a", "a/b"	<No Match>
http://example.com/a/b.htm	""	"/a"

4.6.10 Site Collection Id Lookup

Once the site collection URL is determined, it is passed to **proc_getSiteMap** (section 3.1.5.40), along with the request **web application** ID.

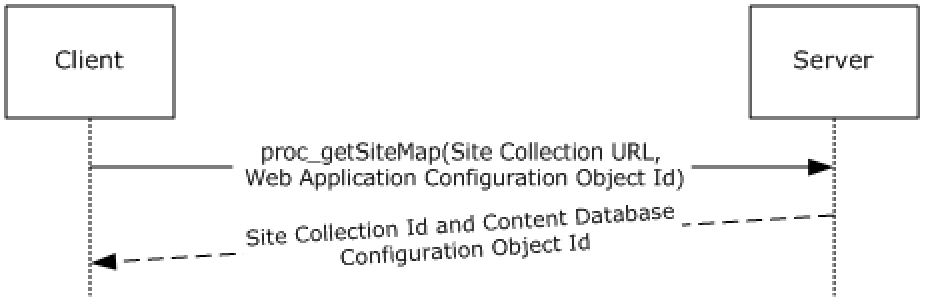
Client
Server
proc_getSiteMap(Site Collection URL, Web Application Configuration Object Id)
Site Collection Id and Content Database Configuration Object Id

Figure 12: site Collection Id Lookup

A site collection ID is returned along with the configuration object ID of the content database in which the site collection content is stored. If the specified combination site collection URL and web application ID cannot be found in the Configuration Database (section 3.1.1), the site collection does not exist and site collection lookup terminates.

4.6.11 Building Content Database Connection String

At this point, the only step that remains is to establish a connection to the content database. This requires a content database connection string, which is built by combining the following components.

4.6.11.1 Name

Once the content database ID is known, it is passed to **proc_GetObject** (section 3.1.5.36), which returns the content database configuration object (section 2.2.6.1.7.2).

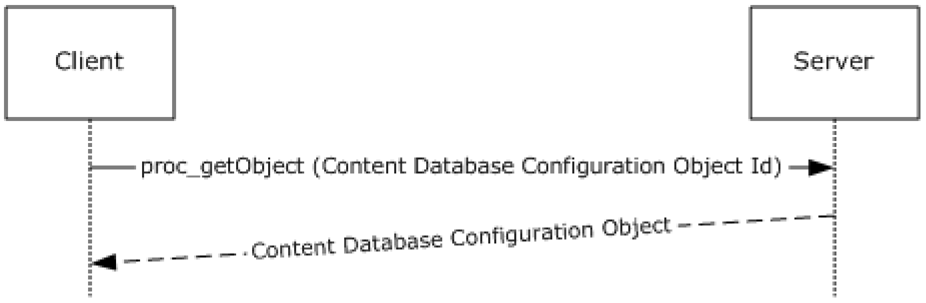
Client
Server
proc_getObject (Content Database Configuration Object Id)
Content Database Configuration Object

Figure 13: Name

The name field of the content database configuration object is used in the connection string as the content database name.

Credentials

The optional Username and Password are extracted from the Properties of the Content Database configuration object using XPath queries.

Instance

The **ParentId** of the content database configuration object is passed in another call to **proc_GetObject**, which returns the database service instance configuration object, the name field of which is used as the Database Server Instance.

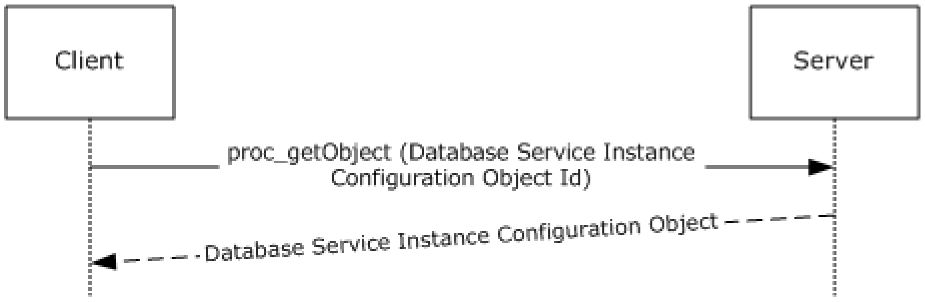
Client
Server
proc_getObject (Database Service Instance Configuration Object Id)
Database Service Instance Configuration Object

Figure 14: Database Server Instance

Server Address

The **ParentId** of the Database Service Instance Configuration Object is then passed to the one final call to **proc_getObject**, which returns a server configuration object. The name field of the server configuration object is used as the Server Address component of the connection string.

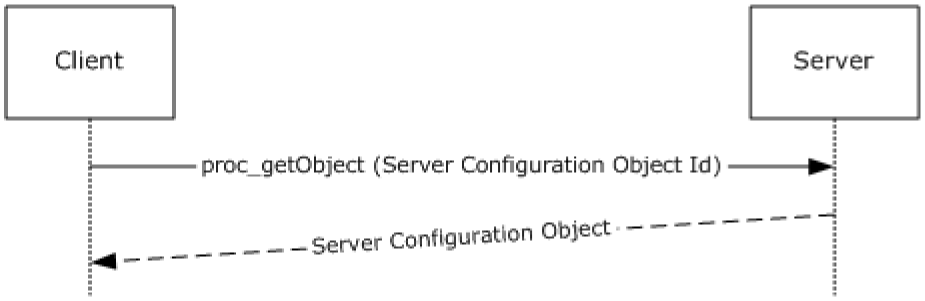
Client
Server
proc_getObject (Server Configuration Object Id)
Server Configuration Object

Figure 15: Server Address

At the end of this step, the incoming URL has been successfully translated into a site collection URL, a site collection Id, and a Content Database Connection String. These components are then ready to be used to call other stored procedures in this and other protocols.

4.7 User Update User Properties OM

This example describes the interactions made when site **member** properties are updated for a particular user.

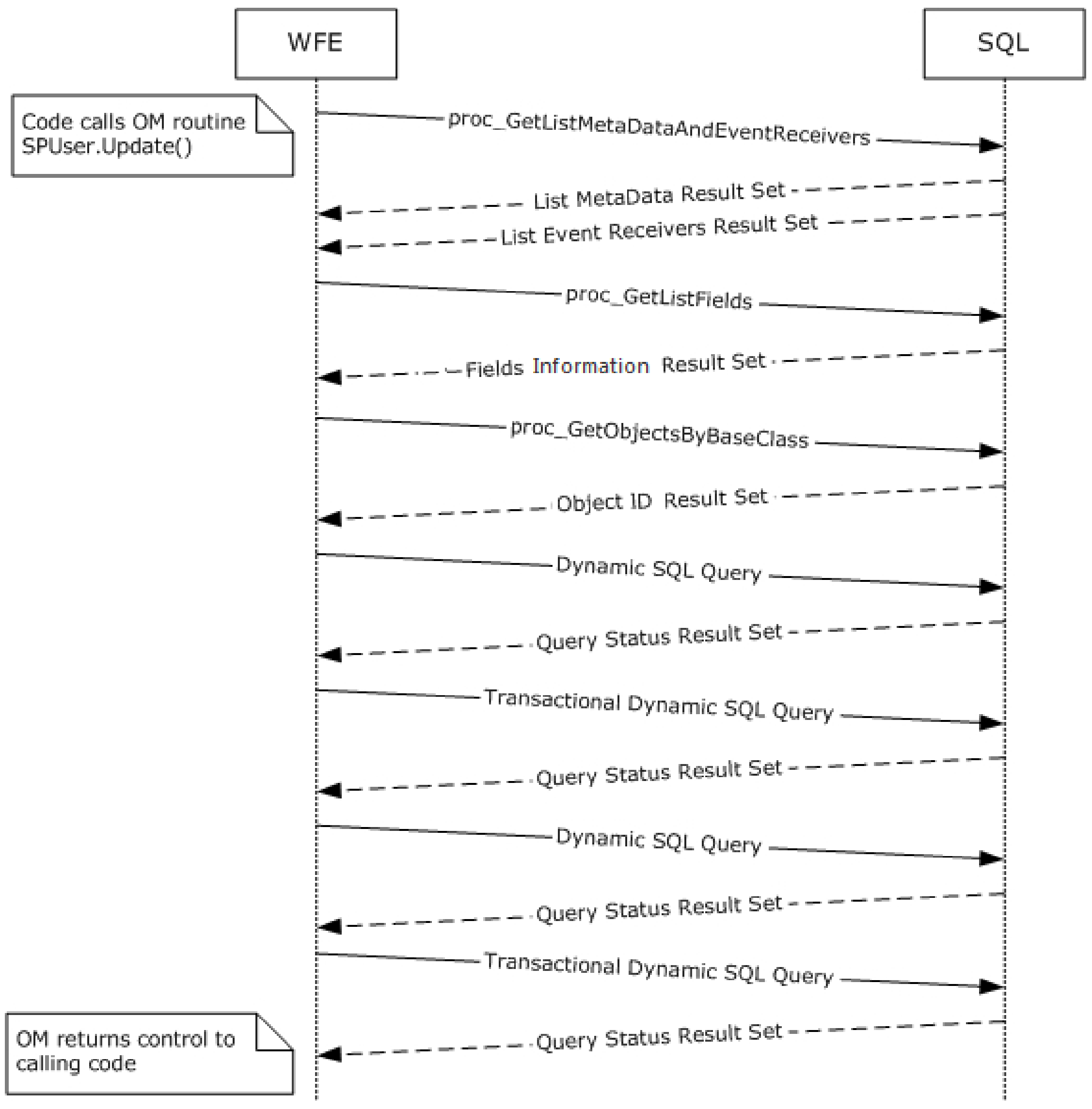
WFE
SQL
Code calls OM routine SPUser.Update()
proc_GetListMetaDataAndEventReceivers
List MetaData Result Set
List Event Receivers Result Set
proc_GetListFields
Fields Information Result Set
proc_GetObjectsByBaseClass
Object ID Result Set
Dynamic SQL Query
Query Status Result Set
Transactional Dynamic SQL Query
Query Status Result Set
Dynamic SQL Query
Query Status Result Set
Transactional Dynamic SQL Query
OM returns control to calling code
Query Status Result Set

Figure 16: user Update user Properties OM

This scenario is initiated by a call to the object model command **SPUser.Update()**. For simplicity's sake, this example assumes that:

- The code has already instantiated the site collection (**SPSite**) and Web (**SPWeb**) Objects for this session, and
- The user to be updated is in the site collection and a member of the site.

The following actions happen:

1. The WFE fetches the properties for the target site collection's user information list, a SharePoint list containing information about users and **groups** registered in a site collection. It does this by calling the stored procedure **proc_GetListMetaDataAndEventReceivers** (section 3.1.5.34) using TDS.

2. The BEDS returns two result sets, which include the List Metadata (section 3.1.5.34.1) and List Event Receivers (section 3.1.5.34.4) for the specified user information list.
3. If the WFE determines that the user information list has not been populated in the current **SPSite** object, it requests the list field information for the site collection's user information list by calling the stored procedure **proc_GetListFields** (section 3.1.5.33) using TDS.
4. The BEDS returns a single Fields Information Result Set (section 3.1.5.33.1), which includes the field information for the specified user information list.
5. If the WFE determines that it needs to populate the list of **SPFeatures**, the list of child objects that inherit from the base class **SPFeatureDefinition** is populated by calling the Configuration Database (section 2.2.6.1) stored procedure **proc_getObjectsByBaseClass** (section 3.1.5.38) using TDS.
6. The BEDS returns a single Object ID result set, which includes a list of child object identifiers for the specified base class and parent object.
7. The WFE builds a dynamic SQL query to select existing information for the user using TDS.
8. The BEDS returns a single result set, which includes existing data for the user.
9. The WFE builds a transactional dynamic SQL query to update the user information in the site. This query is sent to the SQL server using TDS. On the SQL server the following actions occur:
    1. The query begins a new SQL transaction.
    2. The query attempts to update the user information using the stored procedure **proc_SecUpdateUser** (section 3.1.5.123).
    3. The query attempts to update the user's properties in the site's user information list using the stored procedure **proc_UpdateListItem** (section 3.1.5.131).
    4. The query rolls back the SQL transaction if the previous procedures were not successful, or it commits the transaction if they were successful.
10. The BEDS returns a single result set, which indicates the return code status of the actions within the query.
11. The WFE builds a dynamic SQL query to select updated information for the user using TDS.
12. The BEDS returns a single result set, which includes the data for the user.
13. The WFE builds a transactional dynamic SQL query to update the user information in the site. This query is sent to the SQL server using TDS. On the SQL server the following actions occur:
    1. The query begins a new SQL transaction.
    2. The query attempts to update the site collection's user list data using the stored procedure **proc_UpdateListItem**.
    3. Then the query attempts to update the user's properties in the site collection's user information table using the stored procedure **proc_UpdateUserInfoInTableFromRowUpdater** (section 3.1.5.133).
    4. The query rolls back the SQL transaction if the previous procedures were not successful, or it commits the transaction if they were successful.
14. The BEDS returns a successful return code status.

4.8 Version Negotiation

The following scenario is an example of the protocol version negotiation sequence between a WFE and a content database where the content database version does not match that expected by the WFE.

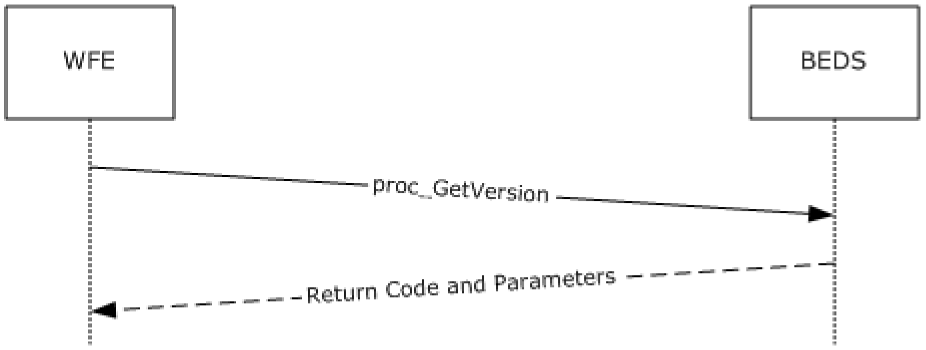

Figure 17: Version Negotiation

1. The WFE calls the **proc_GetVersion** (section 3.1.5.44) stored procedure in the content database on the BEDS, with the *@VersionId* parameter set to "6333368D-85F0-4EF5-8241-5252B12B2E50".

2. The **proc_GetVersion** stored procedure returns the output parameter *@Version* with a value of "4.0.145.0". The application has a pre-defined version of "4.1.6.0", which reflects the current state of the software. Any new database created by this application will have this version. The application determines that the version from the database is smaller (meaning it is older) than the version the WFE expects, and does not perform any further operations against the database.

4.9 File: Add File OM

This example describes the requests and responses made when the front-end web server adds an existing file to a document library on the back-end database server using an object model call.

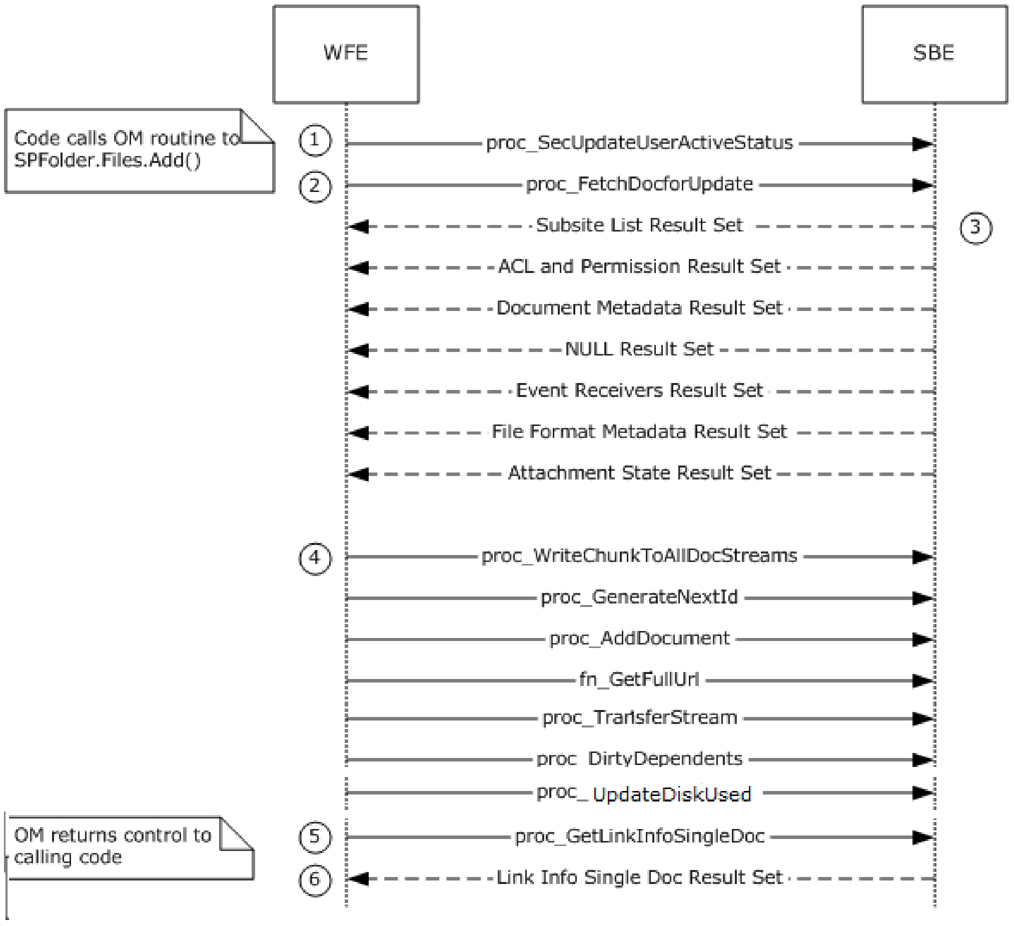
WFE
SBE
Code calls OM routine to SPFolder.Files.Add()
1
proc_SecUpdateUserActiveStatus
2
proc_FetchDocforUpdate
Subsite List Result Set
3
ACL and Permission Result Set
Document Metadata Result Set
NULL Result Set
Event Receivers Result Set
File Format Metadata Result Set
Attachment State Result Set
4
proc_WriteChunkToAllDocStreams
proc_GenerateNextId
proc_AddDocument
fn_GetFullUrl
proc_TransferStream
proc_DirtyDependents
proc_UpdateDiskUsed
OM returns control to calling code
5
proc_GetLinkInfoSingleDoc
6
Link Info Single Doc Result Set

Figure 18: File: Add File OM

This scenario is initiated by a call to the **SPFolder.Files.Add()** object model command. This example assumes that the file is stored on the local file system and that:

- The code has already instantiated the site collection (**SPSite**), site (**SPWeb**), document library (**SPList**), and document library folder (**SPFolder**) objects where the document will be stored.
- The code has already instantiated a **System.IO.FileStream** object around the desired file.
- The current user has File Open permissions for the document.

The following actions happen:

1. The front-end web server builds a dynamic query that invokes the **proc_SecUpdateUserActiveStatus** (section 3.1.5.124) stored procedure.
2. The front-end web server builds a dynamic query that invokes the **proc_FetchDocForUpdate** (section 3.1.5.21) stored procedure.
3. The back-end database server returns a return code of 0, and returns the following result sets:
    - Subsite List Result Set (section 3.1.5.21.1). This returns information on all subsites whose parent site is the desired location to store the document.

- ACL and Permission Result Set (section 2.2.5.1). This returns information about the permissions associated with the scope in effect for the document.
- Document Metadata Result Set (section 2.2.5.6). This returns the document metadata needed to further process the document.
- NULL Result Set (section 3.1.5.21.5). Used as a placeholder and contains no data.
- Event Receivers Result Set (1) (section 3.1.5.21.6). This returns event receiver information on the document if it already exists.
- File Format Metadata Result Set (section 3.1.5.21.10). This returns information about the format of the content of the document.
- Attachment State Result Set (section 3.1.5.21.17). This returns information about the attachment state of the document.

4. The front-end web server builds a transactional dynamic SQL query to store the document.
    1. The query begins a new SQL transaction.
    2. The query attempts to write the data in the corresponding table using the **proc_WriteChunkToAllDocStreams** (section 3.1.5.135) stored procedure.
    3. The query attempts to generate a new Id for the document using the **proc_GenerateNextId** (section 3.1.5.23) stored procedure.
    4. The query attempts to add the document using the **proc_AddDocument** (section 3.1.5.3) stored procedure.
    5. The query retrieves the full URL of the document using the **fn_GetFullUrl** (section 3.1.5.1) function.
    6. The query invokes the **proc_TransferStream** (section 3.1.5.126) stored procedure.
    7. The query attempts to dirty any dependent items in or using the document by calling the **proc_DirtyDependents** (section 3.1.5.15) stored procedure.
    8. The query rolls back the SQL transaction if the previous procedures were not successful.
    9. Assuming the previous procedures were successful, the query updates the disk spaced used for the site by calling the **proc_UpdateDiskUsed** stored procedure and commits the transaction.
5. The front-end web server invokes the **proc_GetLinkInfoSingleDoc** (section 3.1.5.31) stored procedure to provide link information and status for all forward links in the document and all backward links referencing the document in the site collection.
6. The back-end database server returns a status code of 0, and returns the following result set:
    - Link Info Single Doc Result Set (section 3.1.5.31.1). This returns information about links to or within the requested document.

4.10 Folder: Move Folder OM

This example describes the requests and responses made when a folder is moved within a list using an object model call.

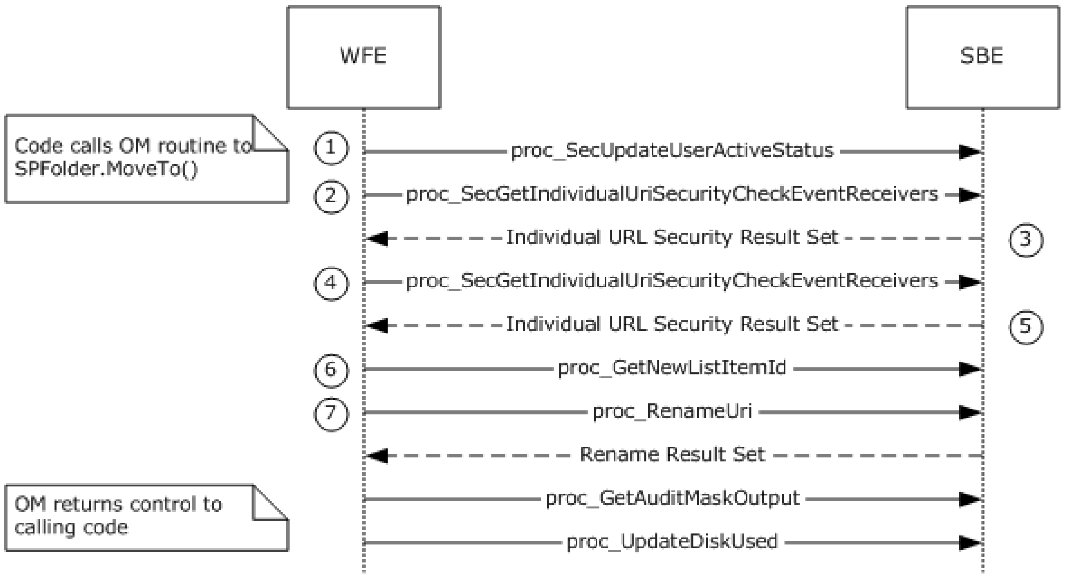
WFE
SBE
Code calls OM routine to SPFolder.MoveTo()
1
proc_SecUpdateUserActiveStatus
2
proc_SecGetIndividualUriSecurityCheckEventReceivers
Individual URL Security Result Set
3
4
proc_SecGetIndividualUriSecurityCheckEventReceivers
Individual URL Security Result Set
5
6
proc_GetNewListItemId
7
proc_RenameUri
Rename Result Set
OM returns control to calling code
proc_GetAuditMaskOutput
proc_UpdateDiskUsed

Figure 19: Move Folder OM

This scenario is initiated by a call to the **SPFolder.MoveTo()** object model command. This example assumes that:

- The code has already instantiated the site collection (**SPSite**), site (**SPWeb**), list (**SPList**), and list folder (**SPFolder**) objects.
- The folder destination location is valid.
- The current user has permissions to view the folder.

The following actions happen:

1. The front-end web server builds a dynamic query that invokes the **proc_SecUpdateUserActiveStatus** (section 3.1.5.124) stored procedure.
2. The front-end web server builds a dynamic query that invokes the **proc_SecGetIndividualUrlSecurityCheckEventReceivers** (section 3.1.5.74) stored procedure against the source folder.
3. The back-end database server returns a return code of 0, and returns the following result set:
   - Individual URL Security Result Set (section 3.1.5.74.1). This returns security information about the specific source folder.
4. The front-end web server builds a dynamic query that invokes the **proc_SecGetIndividualUrlSecurityCheckEventReceivers** stored procedure against the destination folder.
5. The back-end database server returns a return code of 0, and returns the following result set:
   - Individual URL Security Result Set (section 3.1.5.74.1). This returns security information about the specific destination folder.
6. The front-end web server builds a dynamic query that invokes the **proc_GetNewListItemId** stored procedure.
7. The front-end web server builds a transactional dynamic SQL query to move the folder.

1. The query begins a new SQL transaction.
2. The query attempts to rename the URL for the folder using the **proc_RenameUrl** (section 3.1.5.50) stored procedure. The back-end database server returns a return code of 0, and returns the Rename Result Set (section 3.1.5.50.1), which gives basic information about the old and new URLs for the renamed folder.
3. The query attempts to get **Audit Flags** (section 2.2.2.1) information about the new folder using the **proc_GetAuditMaskOutput** (section 3.1.5.26) stored procedure.
4. The query rolls back the SQL transaction if any of the previous procedures were not successful.
5. Assuming the previous procedures were successful, the query updates the disk spaced used for the site by calling the **proc_UpdateDiskUsed** stored procedure and commits the transaction.

5 Security

5.1 Security Considerations for Implementers

Security for this protocol is controlled by the access rights to the databases on the back-end database server, which is negotiated as part of the Tabular Data Stream [MS-TDS] protocol.

In a trusted subsystem model, the process running on the web front end server uses its own **security principal** identity to access the content database on the back-end database server on behalf of the user, rather than using the account of the user accessing the web front end as a database access account to access the content database. The database access account used by the web front end server needs to have access to the 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 [MS-TDS] connection to the content database, or when calling the stored procedures.

5.2 Index of Security Parameters

Security Parameter	Section
proc_SecAddPrincipalToRole	section 3.1.5.51
proc_SecAddRoleDef	section 3.1.5.52
proc_SecAddUser	section 3.1.5.53
proc_SecAddUserToSiteGroup	section 3.1.5.54
proc_SecAddWebMembership	section 3.1.5.55
proc_SecChangeToInheritedList	section 3.1.5.56
proc_SecChangeToInheritedWeb	section 3.1.5.57
proc_SecChangeToUniqueScope	section 3.1.5.58
proc_SecCheckDeletedAccounts	section 3.1.5.59
proc_SecCloneRoleDefinitions	section 3.1.5.60
proc_SecCreateSiteGroup	section 3.1.5.61
proc_SecDecCurrentUsersCount	section 3.1.5.62
proc_SecGetAccountStatus	section 3.1.5.63
proc_SecGetAclFromScope	section 3.1.5.64
proc_SecGetAllAclsForSite	section 3.1.5.65
proc_SecGetAllGroupsAndMembershipInfo	section 3.1.5.66
Proc_SecGetApplicationPrincipalAndUserToken	section 3.1.5.67
proc_SecGetCompleteWebRoleMemberList	section 3.1.5.68
proc_SecGetCurrentUsersCount	section 3.1.5.69
proc_SecGetDomainGroupMapData	section 3.1.5.70
proc_SecGetGroupById	section 3.1.5.71

Security Parameter	Section
proc_SecGetGroupOwner	section 3.1.5.72
proc_SecGetGroupSecurityScopes	section 3.1.5.73
proc_SecGetIndividualUrlSecurityCheckEventReceivers	section 3.1.5.74
Proc_SecGetItemsWithUniquePermissions	section 3.1.5.75
proc_SecGetPrincipalByEmail	section 3.1.5.76
proc_SecGetPrincipalById	section 3.1.5.77
Proc_SecGetPrincipalByIdEx	section 3.1.5.78
proc_SecGetPrincipalByLogin	section 3.1.5.79
proc_SecGetPrincipalByLogin20	section 3.1.5.80
proc_SecGetPrincipalDisplayInformation20	section 3.1.5.81
proc_SecGetRoleAssignments	section 3.1.5.82
proc_SecGetRoleBindingsForAllPrincipals	section 3.1.5.83
proc_SecGetRoleDefs	section 3.1.5.84
proc_SecGetSecurityInfo	section 3.1.5.85
proc_SecGetSiteAdmins	section 3.1.5.86
proc_SecGetSiteGroupById	section 3.1.5.87
proc_SecGetSiteGroupByTitle	section 3.1.5.88
proc_SecGetSiteGroupByTitle20	section 3.1.5.89
proc_SecGetUserAccountDirectoryPath	section 3.1.5.90
proc_SecGetUserPermissionOnGroup	section 3.1.5.91
Proc_SecGetWebsAndListsWithUniquePermissions	section 3.1.5.92
proc_SecListAllSiteMembers	section 3.1.5.93
proc_SecListAllWebMembers	section 3.1.5.94
proc_SecListGroupsInRole	section 3.1.5.95
proc_SecListScopeGroups	section 3.1.5.96
proc_SecListScopeUsers	section 3.1.5.97
proc_SecListSiteGroupMembership	section 3.1.5.98
proc_SecListSiteGroups	section 3.1.5.99
proc_SecListSiteGroupsContainingUser	section 3.1.5.100
proc_SecListSiteGroupsWhichUserOwns	section 3.1.5.101
proc_SecListUsersInRole	section 3.1.5.102
proc_SecMigrateUser	section 3.1.5.103

Security Parameter	Section
proc_SecReCalculateWebFGP	section 3.1.5.104
proc_SecRefreshToken	section 3.1.5.105
proc_SecRemoveGroup	section 3.1.5.106
proc_SecRemovePrincipalFromScope	section 3.1.5.108
proc_SecRemoveRoleDef	section 3.1.5.109
proc_SecRemoveUserFromScopeByLogin	section 3.1.5.110
proc_SecRemoveUserFromSite	section 3.1.5.111
proc_SecRemoveUserFromSiteGroup	section 3.1.5.112
proc_SecRemoveUserFromSiteGroupByLogin	section 3.1.5.113
proc_SecResetItemPerm	section 3.1.5.114
proc_SecResetWebToDefaultRoleDefinition	section 3.1.5.115
proc_SecResolvePrincipal	section 3.1.5.116
proc_SecSetSiteGroupProperties	section 3.1.5.117
proc_SecSetUserAccountDirectoryPath	section 3.1.5.118
proc_SecSetWebRequestAccess	section 3.1.5.119
proc_SecUpdateAnonymousPermMask	section 3.1.5.120
proc_SecUpdateDomainGroupMapData	section 3.1.5.121
proc_SecUpdateRoleDef	section 3.1.5.122
proc_SecUpdateUser	section 3.1.5.123
proc_SecUpdateUserActiveStatus	section 3.1.5.124

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 updates to those products.

- Microsoft SharePoint Foundation 2010

Exceptions, if any, are noted in this section. If an update version, service pack or Knowledge Base (KB) number appears with a product name, the behavior changed in that update. The new behavior also applies to subsequent updates unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

Unless otherwise specified, any statement of optional behavior in this specification that is prescribed using the terms "SHOULD" or "SHOULD NOT" implies product behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term "MAY" implies that the product does not follow the prescription.

<1> Section 2.2.1.10: For example, by default, SharePoint Foundation 2010 creates three implementation-specific site **groups** with default site group identifiers for use in role assignments when provisioning a new site **collection:** "Site Owners" (3), "Site Visitors" (4), and "Site Members" (5).

<2> Section 2.2.2.8: This flag is not used in SharePoint Foundation 2010.

<3> Section 2.2.8.3.3.2: SharePoint Foundation 2010 allows a filtering user interface for fields with the aggregation attribute set to merge, but only in list views where Filter=1 occurs in the view's query parameter.

<4> Section 2.2.8.3.3.2: SharePoint Foundation 2010 does include this attribute with a value of 0.

<5> Section 2.2.8.3.3.2: SharePoint Foundation 2010 sets the value MinusSign for this field.

<6> Section 2.2.8.3.3.3: SharePoint Foundation 2010 emits this data if present in the underlying template definition, even if the field type is not of the specified type.

<7> Section 2.2.8.3.3.3: SharePoint Foundation 2010 emits this data if present in the underlying template definition, even if the field type is not of the specified type.

<8> Section 2.2.8.3.3.3: SharePoint Foundation 2010 emits this data if present in the underlying template definition, even if the field type is not of the specified type.

<9> Section 2.2.8.3.3.3: SharePoint Foundation 2010 emits this data if present in the underlying template definition, even if the field type is not of the specified type.

<10> Section 2.2.8.3.3.3: SharePoint Foundation 2010 emits this data if present in the underlying template definition, even if the field type is not of the specified type.

<11> Section 3.1.5.20.10: SharePoint Foundation 2010 sets this value to 4 when the **SetupPath** is relative to the install location of SharePoint Foundation 2010 on the front-end web server (for example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\14).

<12> Section 3.1.5.20.11: SharePoint Foundation 2010 sets this value to 4 when the **SetupPath** is relative to the install location of SharePoint Foundation 2010 on the front-end web server (for example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\14).

<13> Section 3.1.5.21.14: SharePoint Foundation 2010 sets this value to 4 when the **SetupPath** is relative to the install location of SharePoint Foundation 2010 on the front-end web server (for example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\14).

<14> Section 3.1.5.21.15: SharePoint Foundation 2010 sets this value to 4 when the **SetupPath** is relative to the install location of SharePoint Foundation 2010 on the front-end web server (for example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\14).

<15> Section 3.1.5.44: SharePoint Foundation 2010 uses text-based SQL queries to get the version number which are equivalent in functionality to **proc_GetVersion**.

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