[MS-WSSFO3]:

Windows SharePoint Services (WSS): File Operations Database Communications Version 3 Protocol

---

Revision Summary

Date	Revision History	Revision Class	Comments
1/20/2012	0.1	New	Released new document.
4/11/2012	0.1	None	No changes to the meaning, language, or formatting of the technical content.
7/16/2012	0.1	None	No changes to the meaning, language, or formatting of the technical content.
9/12/2012	0.1	None	No changes to the meaning, language, or formatting of the technical content.
10/8/2012	1.0	Major	Significantly changed the technical content.
2/11/2013	2.0	Major	Significantly changed the technical content.
7/30/2013	2.0	None	No changes to the meaning, language, or formatting of the technical content.
11/18/2013	2.0	None	No changes to the meaning, language, or formatting of the technical content.
2/10/2014	2.1	Minor	Clarified the meaning of the technical content.
4/30/2014	2.2	Minor	Clarified the meaning of the technical content.
7/31/2014	2.3	Minor	Clarified the meaning of the technical content.
10/30/2014	2.4	Minor	Clarified the meaning of the technical content.
2/26/2016	3.0	Major	Significantly changed the technical content.
7/15/2016	3.0	None	No changes to the meaning, language, or formatting of the technical content.
9/14/2016	3.0	None	No changes to the meaning, language, or formatting of the technical content.
1/12/2017	3.1	Minor	Clarified the meaning of the technical content.

Table of Contents

1 Introduction

The File Operations Database Communications Version 3 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 (2)** 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. Importantly, 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.

Active Directory account creation mode: A type of account creation mode that retrieves and uses user accounts in a specific Active Directory Domain Services (AD DS) organizational unit.

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.

app: See **web application**.

app host header name: A unique name assigned to an app instance.

app principal: Designates an authenticated entity that is not a user.

app site domain identifier: A unique specifier containing six hexadecimal number values that is used to determine a particular host header for a subsite.

app web domain identifier: A unique specifier containing eight hexadecimal number values that is used to determine a particular host header for a subsite.

ASP.NET: A web server technology for dynamically rendering HTML pages using a combination of HTML, Javascript, CSS, and server-side logic. For more information, see [ASPNET].

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.

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

blog: A website that contains a series of posts about a subject and is arranged in reverse chronological order. Also referred to as web log.

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.

calculated column: A column (2) in a table that contains a formula that is copied automatically to each record in the column.

cascading style sheet (CSS): An extension to **HTML** that enables authors and users of HTML documents to attach style sheets to those documents, as described in [CSS-LEVEL1] and [CSS-LEVEL2]. A style sheet includes typographical information about the appearance of a page, including the font for text on the page.

Central Administration site: A SharePoint site that an administrator can use to manage all of the sites and servers in a server farm that is running SharePoint Products and Technologies.

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: The range of characters used to represent textual data within a MIME body part, as described in [RFC2046].

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. See also **check in**.

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.

class identifier (CLSID): A **GUID** that identifies a software component; for instance, a DCOM object class (4) or a COM class.

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.

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).

crawler: A process that browses and indexes content from a content source.

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.

customized: A document, column, or content type whose content is stored in a content database instead of a front-end file system. Also referred to as unghosted.

datasheet: A worksheet window that contains the source data for a Microsoft Graph chart object.

datetime: A data type that represents the date and time when a document can be normalized and indexed as a numeric value by a search application. The range and degree of granularity varies according to search application and implementation.

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.

displayed version: Document version information that is formatted for display in the user interface. The displayed version uses the format MajorVersion.MinorVersion, where MajorVersion is the published version number and MinorVersion is the draft version number, separated by a decimal point. See also **major version** and **minor version**.

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 (1)**, or **site (2)**. 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 (2)**, **list (1)**, 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 (1)**, 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.

expire: A process in which an object, such as an external data connection, becomes invalid because its allotted time period has ended.

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: (1) An element that can contain fields and other groups in the data source for an InfoPath form. Controls that contain other controls, such as repeating tables and sections, are bound to groups.

(2) 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 (2) and digital signing.

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.

Hypertext Transfer Protocol Secure (HTTPS): An extension of HTTP that securely encrypts and decrypts web page requests. In some older protocols, "Hypertext Transfer Protocol over Secure Sockets Layer" is still used (Secure Sockets Layer has been deprecated). For more information, see [SSL3] and [RFC5246].

Input Method Editor (IME): An application that is used to enter characters in written Asian languages by using a standard 101-key keyboard. An IME consists of both an engine that converts keystrokes into phonetic and ideographic characters and a dictionary of commonly used ideographic words.

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 (1).

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

(2) An organization of a region of cells into a tabular structure in a workbook.

list identifier: A GUID that is used to identify a **list (1)** 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 server template: A value that identifies the template that is used for a SharePoint list.

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.

manifest: A file that stores metadata about an expansion pack, such as the name of the expansion pack, the files and resources that are included in the expansion pack, and the dependencies that it has on other files and components.

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.

metadict: A dictionary that has strongly typed values.

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: (1) An authenticated entity that initiates a message or channel in a distributed system.

(2) An identifier of such an entity.

provisioned: A condition of an object that was created and deployed successfully.

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 (1) that indicates that an object cannot be changed or deleted. The object can only be accessed or displayed.

Really Simple Syndication (RSS): An XML-based syndication format for content, as described in [RSS2.0].

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.

rich text: Text that is formatted in the Rich Text Format, as described in [MSFT-RTF].

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.

Secure Sockets Layer (SSL): A security protocol that supports confidentiality and integrity of messages in client and server applications that communicate over open networks. SSL uses two keys to encrypt data-a public key known to everyone and a private or secret key known only to the recipient of the message. SSL supports server and, optionally, client authentication (2) using X.509 certificates (2). For more information, see [X509]. The SSL protocol is precursor to **Transport Layer Security (TLS)**. The TLS version 1.0 specification is based on SSL version 3.0 [SSL3].

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

security identifier (SID): An identifier for **security principals** in Windows 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.

site: (1) A group of related webpages that is hosted by a server on the World Wide Web or an intranet. Each website has its own entry points, metadata, administration settings, and workflows. Also referred to as web site.

(2) 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 subscription identifier: A GUID that is used to identify a site subscription.

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 (/).

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

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.

stream binary piece: A block of binary data containing the length (in bytes) of a data stream and the data stream itself.

stream identifier: A 64-bit integer that uniquely specifies a stream binary piece.

stream partition: A byte that identifies the container for a stream binary piece.

stream schema: A numeric selector that specifies the format of a file's stream binary pieces.

Structured Query Language (SQL): A database query and programming language that is widely used for accessing, querying, updating, and managing data in relational database systems.

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.

TCP/IP: A set of networking protocols that is widely used on the Internet and provides communications across interconnected networks of computers with diverse hardware architectures and various operating systems. It includes standards for how computers communicate and conventions for connecting networks and routing traffic.

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**.

Transact-Structured Query Language (T-SQL): A language that contains the commands that are used to manage instances of Microsoft SQL Server, create and manage all objects in an instance of SQL Server, and to insert, retrieve, modify, and delete all data in SQL Server tables. Transact-SQL is an extension of the language that is defined in the SQL standards that are published by the International Standards Organization (ISO) and the American National Standards Institute (ANSI).

Transport Layer Security (TLS): A security protocol that supports confidentiality and integrity of messages in client and server applications communicating over open networks. **TLS** supports server and, optionally, client authentication by using X.509 certificates (as specified in [X509]). **TLS** is standardized in the IETF TLS working group.

uncustomized: A condition of a document whose content is stored in a location other than the content database. If a document is uncustomized, the front-end web server determines the location of the content by using the SetupPath value for the document. Also referred to as ghosted.

Uniform Resource Identifier (URI): A string that identifies a resource. The URI is an addressing mechanism defined in Internet Engineering Task Force (IETF) Uniform Resource Identifier (URI): Generic Syntax [RFC3986].

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].

upgrade evaluation site collection: A copy of the current **site collection**, used to evaluate the functionality of a site collection after it is upgraded.

user account directory path: A string representation of the Lightweight Directory Access Protocol (LDAP) distinguished name for an 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 interface (UI) version: A single 4-byte integer that stores the version number that appears as a document version number in the user interface. The lower 9 bits correspond to the minor

version number of the displayed version. The remaining 23 bits correspond to the major version number of the displayed version. See also **displayed version**.

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: A structured HTML section of a Web Parts Page that contains zero or more Web Parts and can be configured to control the organization and format of those Web Parts.

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

website: A group of related webpages that is hosted by a server on the World Wide Web or an intranet. Each website has its own entry points, metadata, administration settings, and workflows. Also referred to as site.

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.

XML schema definition (XSD): The World Wide Web Consortium (W3C) standard language that is used in defining XML schemas. Schemas are useful for enforcing structure and constraining the types of data that can be used validly within other XML documents. XML schema definition refers to the fully specified and currently recommended standard for use in authoring XML schemas.

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", 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

[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

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

[CSS3UI] Celik, T., Ed., "CSS Basic User Interface Module Level 3 (CSS3 UI)", W3C Working Draft 17, January 2012, http://www.w3.org/TR/2012/WD-css3-ui-20120117/

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

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 (2)**. 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 specified in [MS-FPSE]), **Hypertext Transfer Protocol (HTTP)** (as specified in [RFC2616]), and WebDAV (as specified 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 (2)**, 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 (2), the front-end Web server confirms whether the data is already populated in the local objects that represent the specific user or groups (2). 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 (2), 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.

The relationship between Tabular Data Stream Protocol (TDS), SNMP Multiplexing (SMUX), **Transport Layer Security (TLS)** and **Secure Sockets Layer (SSL)**, and Transmission Control Protocol/Internet Protocol (**TCP/IP**) is illustrated in the following figure.

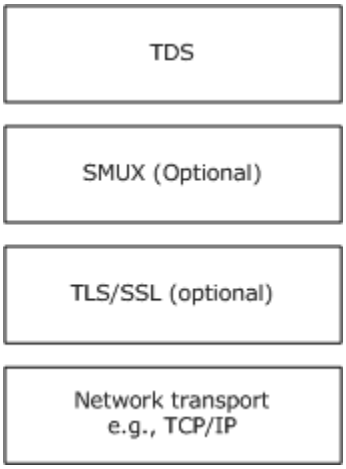


Figure 1: Protocol relationship

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 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 (2)**, **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 (2)** 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 it needs to connect with the back-end database server. The version information is stored in the Versions table (section 2.2.6.12).

1.8 Vendor-Extensible Fields

For complex types defined in this document, 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 2.2.7.3.3), 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 Common Data Types

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 is 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 and Enumerations

2.2.1.1 Simple Data Types

2.2.1.1.1 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).

2.2.1.1.2 Document Identifier

A **document identifier** is a **GUID**, as specified in [MS-DTYP] section 2.3.2.3, used to uniquely identify a document within a **site collection**. Specialized varieties of document identifiers include **site identifiers** and **list identifiers**.

2.2.1.1.3 Event Receiver Identifier

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

2.2.1.1.4 Feature Identifier

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

2.2.1.1.5 List Identifier

A list identifier is a variety of **document identifier**. The list identifier is a GUID used to uniquely identify a list within a site collection.

2.2.1.1.6 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.

2.2.1.1.7 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

2.2.1.1.8 Scope Identifier

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

2.2.1.1.9 Site Collection Identifier

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

2.2.1.1.10 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 and cannot overlap. Values of -1 and 0 are reserved to indicate invalid or unknown user or site group identifiers.<1>

2.2.1.1.11 Site Identifier

A site identifier is a variety of **document identifier**. The site identifier is a GUID used to uniquely identify a **site (2)** within a site collection.

2.2.1.1.12 SystemID

A **SystemID** is a binary value of arbitrary but limited length that uniquely identifies a **principal (2)**, stored on the BEDS as a T-SQL varbinary(512).

2.2.1.1.13 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 and cannot overlap. Certain user identifiers are predefined such as the system account (1073741823), the site collection **owner**, and the secondary site collection contact (2). Values of -1 and 0 are reserved to indicate invalid or unknown user or site group identifiers.

2.2.1.1.14 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.

2.2.1.1.15 Web Part Identifier

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

2.2.1.1.16 Workflow Identifier

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

2.2.1.1.17 tPermMask

A **tPermMask** is an 8-byte integer value used to specify the rights that can be assigned to a user or site group.

2.2.1.2 Enumerations

2.2.1.2.1 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 **Uniform Resource Locator (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").

2.2.1.2.2 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 (2)**.
7	A site collection.

2.2.1.2.3 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
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
23	Umm al-Qura

2.2.1.2.4 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 defined 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
7	Cyrillic_General
8	Czech
9	Danish_Norwegian

Value	Description
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
39	Azeri_Cyrillic_90
40	Azeri_Latin_90
41	Chinese_Hong_Kong_Stroke_90

Value	Description
42	Divehi_90
43	Indic_General_90
44	Kazakh_90
45	Macedonian_FYROM_90
46	Syriac_90
47	Tatar_90
48	Uzbek_Latin_90

2.2.1.2.5 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)**.
2	The event host is a **list (2)**.
3	The event host is a list item.
4	The event host is a content type.
5	The event host is a workflow (2).
6	The event host is a feature.

2.2.1.2.6 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.
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.

Value	Description
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 (2)** is deleted.
203	The event receiver is invoked before a site (2) is moved.
204	The event receiver is invoked before a site (2) is added.
501	The event receiver is invoked before a workflow (2) 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.
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 (2) is deleted.
10203	The event receiver is invoked after a site (2) is moved.

Value	Description
10204	The event receiver is invoked after a site (2) is **provisioned**.
10501	The event receiver is invoked after a workflow (2) is started.
10502	The event receiver is invoked after a workflow (2) is postponed.
10503	The event receiver is invoked after a workflow (2) is completed.
20000	The event receiver is invoked when an e-mail message is received by a list.
32766	The event receiver is context sensitive and inspects the **ContextType** value to perform a corresponding action.
32767	The event receiver is used as part of a workflow (2).

2.2.1.2.7 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 because of 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	The **root folder** in a list or document library.

2.2.1.2.8 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.
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 an **uncustomized** document that does not fall into any other category.

2.2.1.2.9 LinkSecurity Type

A **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. For more details, see [RFC2660]).
S	The URL begins with "https://" (an **SSL** link using the https: or snews: scheme).
U	The URL is of another unknown scheme.

2.2.1.2.10 LinkType Types

LinkType Types 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 a **Hypertext Markup Language (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 (2)**.
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.
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.

Value	Description
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 (2) 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.

2.2.1.2.11 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

2.2.1.2.12 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

Value	Description
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
121	Solution Catalog Template
122	No Code Workflows Public Template
123	Theme Catalog Template
124	DESIGN Catalog Template
125	APPDATA Catalog Template
130	Data Connection Library Template
140	Workflow History Template
150	Gantt Tasks Template
151	Help Library Template
160	Access Request Template
171	Tasks with Timeline and Hierarchy Template
175	Maintenance Logs Template
200	Meetings Template
201	Agenda Template
202	Meeting User Template
204	Decision (Meeting) Template

Value	Description
207	Meeting Objective Template
210	Textbox Template
211	Things To Bring (Meeting) Template
212	Homepage Library Template
301	Posts (Blog) Template
302	Comments (Blog) Template
303	Categories (Blog) Template
402	Facility Template
403	Whereabouts Template
404	Call Track Template
405	Circulation Template
420	Timecard Template
421	Holiday Template
499	IMEDic Template
600	External List Template
700	My Site Document Library Template
1100	Issue Tracking Template
1200	Admin Tasks
1220	Health Rules Template
1221	Health Reports Template
1230	Draft Apps Library in Developer Site Template

2.2.1.2.13 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.

Value	Description
4	The list item is scheduled for automatic approval at a future date.

2.2.1.2.14 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	The **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.

2.2.1.2.15 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.

2.2.1.2.16 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

2.2.1.2.17 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.

2.2.1.2.18 App Principal Permissions Enumeration

The following table lists the values and the descriptions for the **App Principal Permissions** enumeration.

Value	Description
0	No permissions.
1	Guest permissions.
2	Read permissions.
3	Write permissions.
4	Manage permissions.

Value	Description
5	Admin Permissions.

2.2.1.3 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
22	(GMT+07:00) Bangkok, Hanoi, Jakarta
23	(GMT+05:30) Chennai, Kolkata, Mumbai, New Delhi

Value	Meaning
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, Tashkent
48	(GMT+04:30) Kabul
49	(GMT+02:00) Cairo
50	(GMT+02:00) Harare, Pretoria
51	(GMT+03:00) Moscow, St. Petersburg, Volgograd

Value	Meaning
53	(GMT-01:00) Cape Verde Is.
54	(GMT+04:00) Baku
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

Value	Meaning
81	(GMT-04:00) Manaus
82	(GMT+03:00) Tbilisi
83	(GMT+02:00) Windhoek
84	(GMT+04:00) Yerevan
85	(GMT -03:00) Buenos Aires
86	(GMT) Casablanca
87	(GMT +05:00) Islamabad, Karachi
88	(GMT -04:30) Caracas
89	(GMT +04:00) Port Louis
90	(GMT -03:00) Montevideo
91	(GMT -04:00) Asuncion
92	(GMT +12:00) Petropavlovsk-Kamchatsky
93	(GMT) Coordinated Universal Time
94	(GMT +08:00) Ulaanbaatar
99	(GMT +11:00) Magadan

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.

Value	Description
0x00000080	Audit List Schema change operations.
0x00000100	Audit security change operations.
0x00000200	Audit undelete operations.
0x00000400	Audit workflow (2) 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 **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.

Value	Description
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 is 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 **customized**.
0x00000080	The document is, by default, a page which contains a personalized view showing the **current user**'s personalized and customized Web Parts.
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 is filtered out from file system-centric enumerations).
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 customized 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 **uncustomized** but SHOULD be customized 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	Currently unused and MUST be ignored.
0x00020000	There is a shared lock on the document.
0x00040000	The document is the Welcome page of the **site (2)**.
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

Value	Description
0x01	Folder
0x02	Site
0x80	Backward Link 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 **site (2)** 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 (2) 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** 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 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 (2) as a site template.
0x0000000000004000	The **List Server Template** for this list can only be instantiated in the root site (2) of a given site collection.

Value	Description
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 **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 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 (2) 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 has 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**.

Value	Description
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.
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 (2) 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 (2) 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 is shown only to users who have **permissions** to see documents in a draft state.
255	This document is checked out to a particular user and shown only 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 **user interface (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 user interface (UI) version to published.
0x00020000	Overwrite the document without updating the displayed version of the document.
0x00040000	Unpublish document: Change user interface (UI) version from published to draft.

Value	Description
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 the event 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** 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.

Value	Description
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.
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. 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. 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.

Value	Description
0x08000000	User-defined workflows (2) are disabled for this site collection.
0xF0000000	Unused.

2.2.2.10 Site Collection Upgrade Flags

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

Value	Description
0x00000001	Whether site collection upgrade is allowed on this site collection is controlled by the parent of the site collection.
0x00000002	Site collection upgrade is allowed on this site collection. This bit is ignored if bit 0x00000001 is set.
0x00000004	The site collection upgrade is in progress for this site collection.
0x00000008	Whether **upgrade evaluation site collection** is allowed to be created is controlled at this site collection. When this bit is set, use the setting at the parent of the site collection.
0x00000010	upgrade evaluation site collection is allowed to be created. This bit is ignored if bit 0x00000008 is not set.

2.2.2.11 Site Property Flags

Site Property Flags is a 4-byte, unsigned integer bit mask that tracks property flags applied to a **site (2)**. The site (2) 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 (2) allows display of implementation-specific user presence information in the front-end Web server.
0x00000002	This site (2) 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 (2).
0x00000008	This site (2) has disabled syndication of list items via **RSS**.
0x00000010	Unused.

Value	Description
0x00000020	The front-end Web server for this site (2) displays the "quick launch" navigational element.
0x00000040	The front-end Web server for this site (2) displays a hierarchical "tree view" navigational element.
0x00000080	Document parsing is disabled for this site (2).
0x00000100	This site (2) has not yet been provisioned with a site template.
0x00000200	List schema information can be cached for lists within this site (2).
0x00000400	This site (2) has at least one uniquely secured object within it.
0x00000800	Search indexing agents can index the rendered content from ASPX pages within this site (2).
0x00001000	Search indexing agents MUST NOT index the rendered content from ASPX pages within this site (2).
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 (2) has at least one custom action that resides at the site level.
0x00020000	The site (2) has at least one custom action that resides at the list level.
0x00040000	This site (2) has been initialized since upgrade occurred.
0x00080000	This site (2) isn't taken offline by the client.
0xFFFFE000	Unused.

2.2.2.12 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.
0x00000004	The user will not have permission to browse the **user information list**.

2.2.2.13 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 isn't 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.
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.

Value	Description
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 NOT 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.

2.2.2.14 Workdays Flag

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.

2.2.2.15 WSS Rights Mask

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.
AnonymousSearchAccessList	0x0000000000002000	Allow anonymous users to retrieve content of a list or document library through SharePoint search. The list permissions in the **site (2)** do not change.
	0x000000000000D000	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 (2), list, or folder.
ViewPages	0x0000000000020000	Allow viewing of pages in a site (2).
AddAndCustomizePages	0x0000000000040000	Allow addition, modification, or deletion of HTML pages or Web Part pages, and editing of the site (2) using a compatible editor.
ApplyThemeAndBorder	0x0000000000080000	Allow application of a theme or borders to the entire site (2).

Symbolic name	Value	Description
ApplyStyleSheets	0x0000000000100000	Allow application of a style sheet (.css file) to the site (2).
ViewUsageData	0x0000000000200000	Allow viewing of reports on site usage.
CreateSSCSite	0x0000000000400000	Allow creation of a site (2) using Self-Service Site Creation, an implementation-specific capability.
ManageSubwebs	0x0000000000800000	Allow creation of a **subsite** within the site (2) or site collection.
CreateGroups	0x0000000001000000	Allow creation of a **group (2)** of users that can be used anywhere within the site collection.
ManagePermissions	0x0000000002000000	Allow creation and modification of **permission levels** on the site (2) and assigning permissions to users and site groups.
BrowseDirectories	0x0000000004000000	Allow enumeration of documents and folders in a site (2) using [MS-FPSE] and WebDAV interfaces.
BrowseUserInfo	0x0000000008000000	Allow viewing the information about all users of the site (2).
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 (2) as well as manage content.
AnonymousSearchAccessWebLists	0x0000000080000000	Allow content of lists and document libraries in the site (2) to be retrievable for anonymous users through SharePoint search if the list or document library has **AnonymousSearchAccessList** set.
	0x0000000F00000000	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 (2).
ManageAlerts	0x0000004000000000	Allow management of **alerts** for all users of the site (2).
CreateAlerts	0x0000008000000000	Allow creation of e-mail alerts.
EditMyUserInfo	0x0000010000000000	Allow users to change their own user information, such as adding a picture.
	0x0000FE0000000000	Reserved.

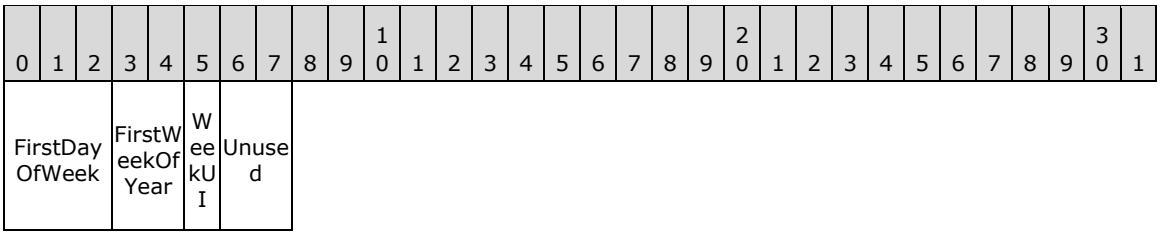
The Special permissions (0xXXXX000000000000) are specified as follows.

Symbolic name	Value	Description
	0x3FFF000000000000	Reserved.
EnumeratePermissions	0x4000000000000000	Allow enumeration of permissions on the site (2), list, folder, document, or list item.
	0x8000000000000000	Reserved.

2.2.3 Binary Structures

2.2.3.1 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.



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 is handled. The following are valid values for the **FirstWeekOfYear** value.

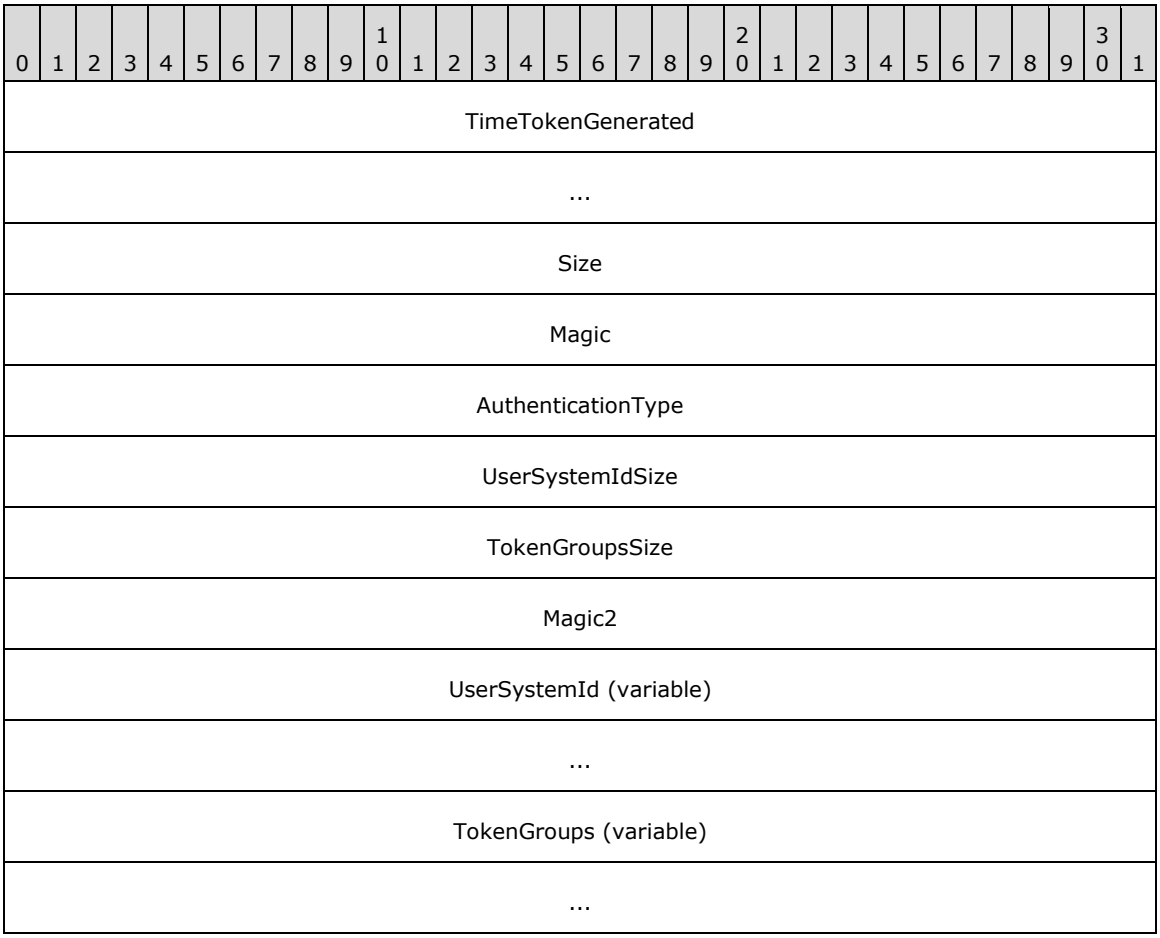
Bits	Description
00	The year starts on January 1.
01	The year starts with the first complete week.
10	The year starts with the first week of at least four days.

WeekUI (1 bit): If this bit is set, week numbers SHOULD be displayed in the front-end Web server.

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

2.2.3.2 External Group Token

The **External Group Token** structure is a variable-length structure associated with a **principal (2)** that contains a collection of the **systemIDs** for the external groups of which the principal is a **member**.



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.

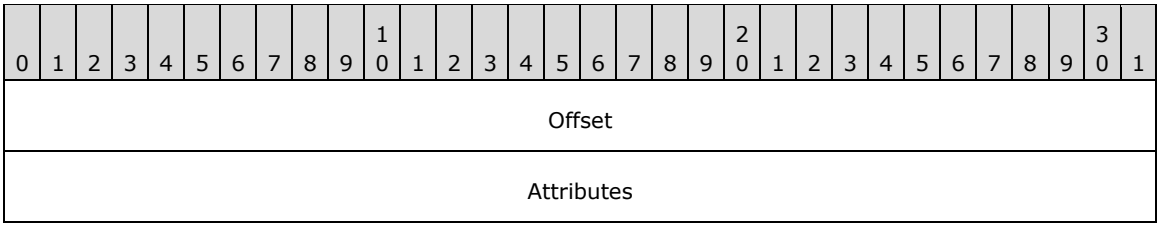
Magic2 (4 bytes): A 4-byte, unsigned integer, which MUST use the value 0xDADBDEDF.

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**.

2.2.3.3 Token Group Offset and Attributes

The **Token Group Offset and Attributes** structure specifies the length and offset of the serialized binary **systemID** for a corresponding external group within a **Token Groups** structure.

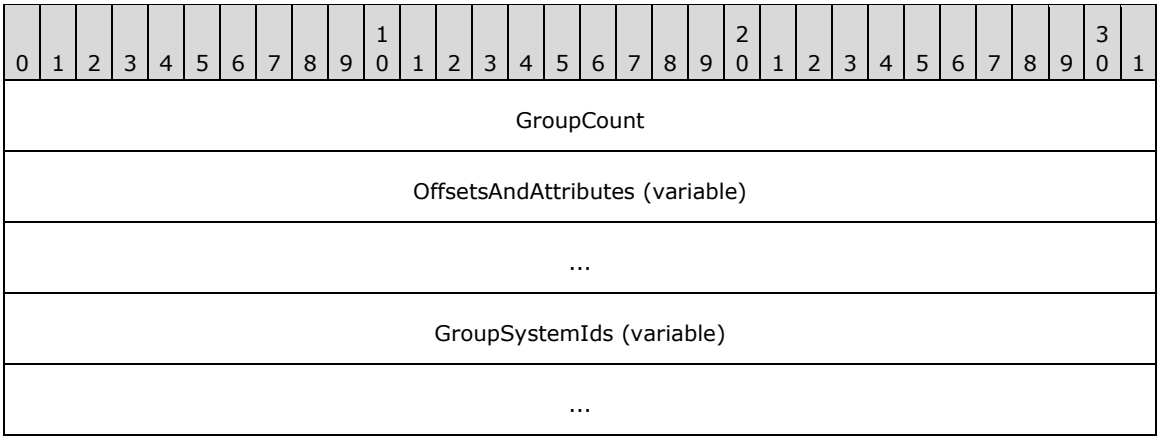


Offset (4 bytes): A 4-byte, unsigned integer value specifying the offset in bytes from the beginning of the **Token Groups** structure to the beginning of the serialized binary **systemID** for this external group.

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

2.2.3.4 Token Groups

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



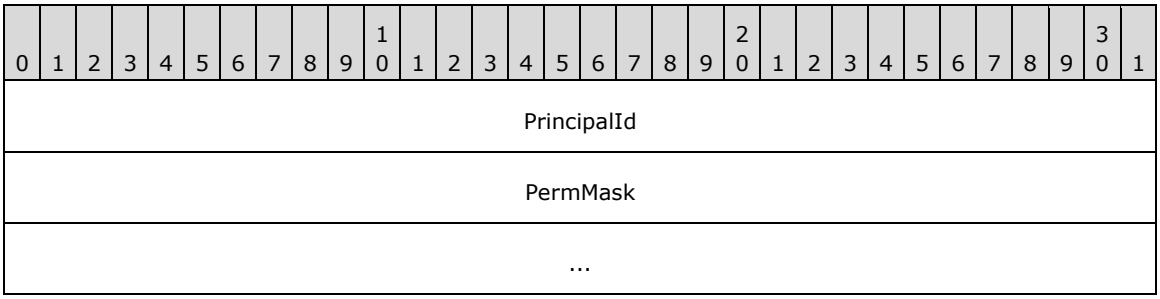
GroupCount (4 bytes): A 4-byte, unsigned integer value specifying the number of external groups' **SystemIDs** that are serialized in this structure.

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.

2.2.3.5 WSS ACE

An ACE structure specifying the individual access rights of a **principal (2)**.

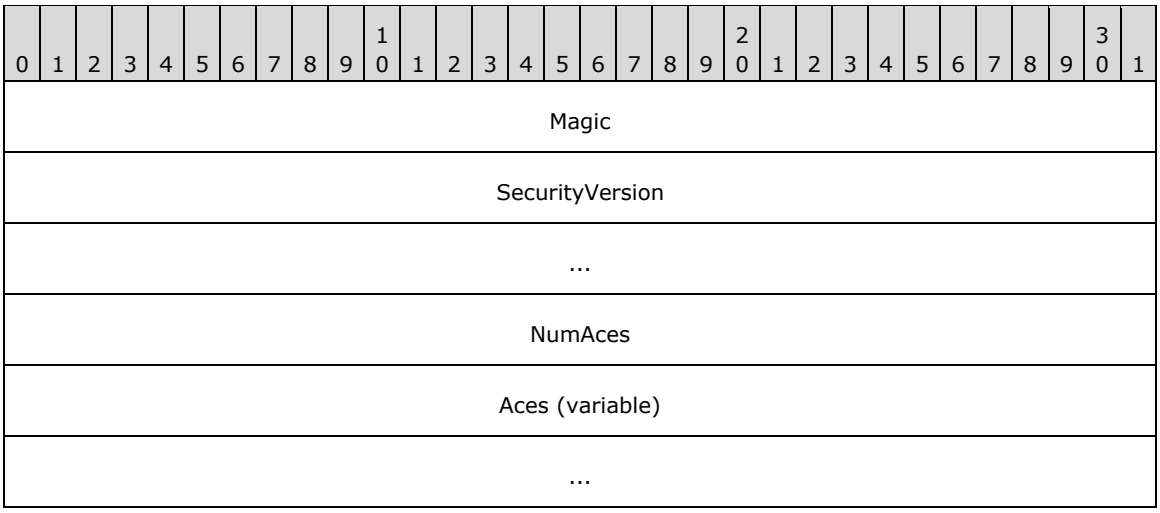


PrincipalId (4 bytes): A 4-byte, signed integer specifying the User Identifier of the principal for this ACE.

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

2.2.3.6 WSS ACL Format

The **WSS ACL Format** structure contains an array of **WSS_ACE**s.



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

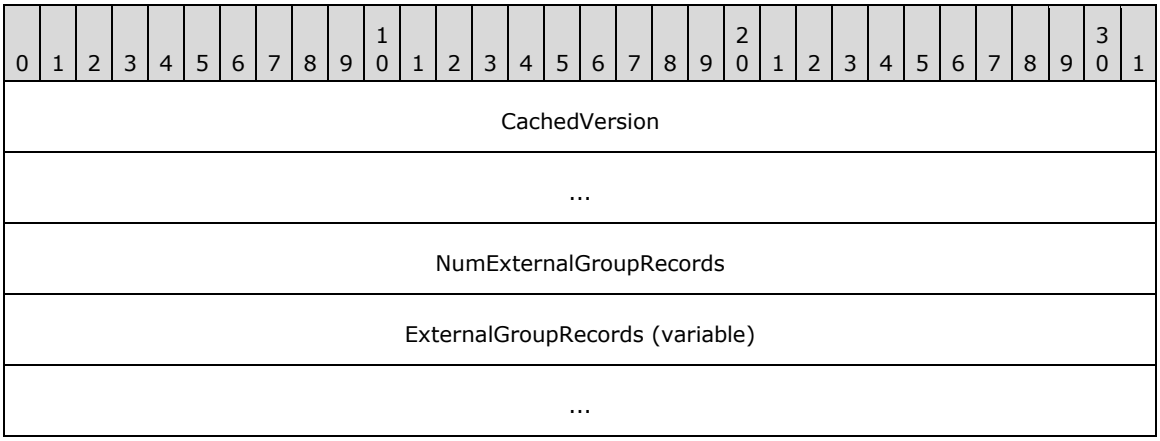
SecurityVersion (8 bytes): An 8-byte, signed integer specifying the site collection's security version value, which was used to compute the ACL.This value is not currently used.

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

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

2.2.3.7 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.



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

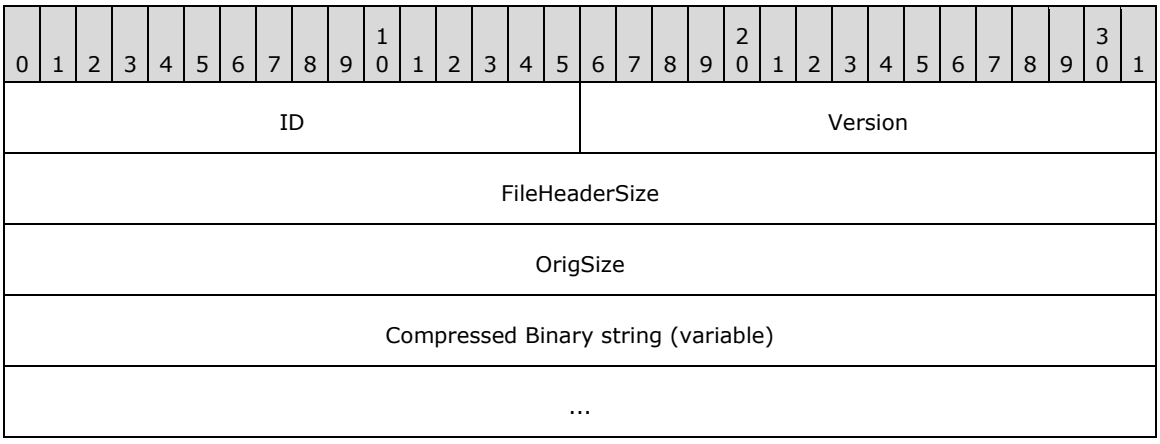
NumExternalGroupRecords (4 bytes): A 4-byte, signed integer specifying the count of **WSS External Group Records** present in this cache.

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

2.2.3.8 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.



ID (2 bytes): MUST be 0xA8A9.

Version (2 bytes): For Microsoft SharePoint Foundation 2013, MUST be 0x3031 (ASCII "01")

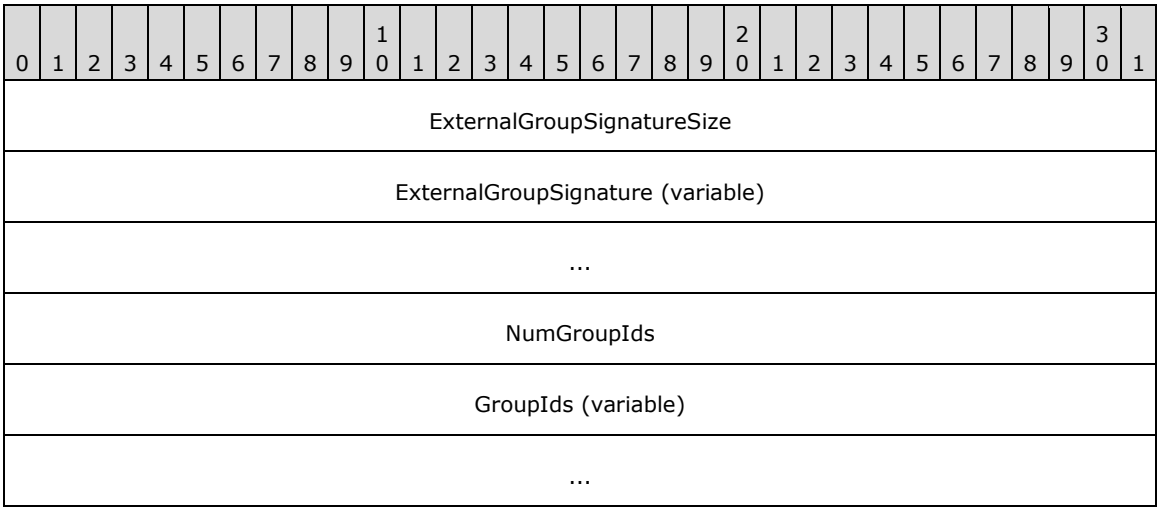
FileHeaderSize (4 bytes): A 4-byte, unsigned integer specifying the size of the header. In Microsoft SharePoint Foundation 2013, it is 0x0C000000.

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.

2.2.3.9 WSS External Group Record

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



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

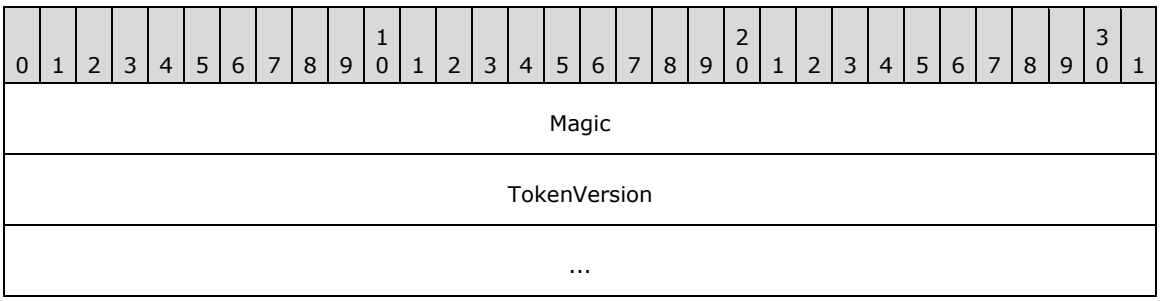
ExternalGroupSignature (variable): An array of bytes specifying the signature for the external group using **UTF-8** encoding. The number of bytes is specified by the **ExternalGroupSignatureSize** field.

NumGroupIds (4 bytes): A 4-byte, signed integer specifying the count of Site Group Identifiers 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.

2.2.3.10 WSS User Token

The **WSS User Token** structure contains an array of Site Group Identifiers.



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.

2.2.4 Result Sets

The following common result sets are used by this protocol.

2.2.4.1 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 2.2.3.6) **access control list (ACL)** for the security scope in effect.

AnonymousPermMask: A **WSS Rights Mask** (section 2.2.2.15) indicating the permissions granted to an anonymous user or a user who has no specific permissions on this document.

2.2.4.2 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**, site identifier, or list identifier for which the custom action resides.

Id: The custom action identifier.

Properties: The custom action data describing its functionality.

Version: The custom action version.

2.2.4.3 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** of the external group.

GroupId: The Site Group Identifier of the Site Group containing the given **domain group**.

2.2.4.4 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.

2.2.4.5 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 2.2.4.4), this MUST be NULL.

2.2.4.6 Document Content Metadata Result Set

The **Document Content Metadata Result Set** contains the metadata for the document's binary **stream**.

```
{Size}                          int,
{SiteRbsCollectionId}           int,
{Version}                       int,
{InteralVersion}                int,
{HistVersion}                   int,
{Id}                            uniqueidentifier,
{DirName}                       nvarchar(256),
{LeafName}                      nvarchar(128),
{ParentId}                      uniqueidentifier,
{SetupPathVersion}              tinyint,
{SetupPath}                     nvarchar(255),
{Dirty}                         bit,
{DocFlags}                      int,
{Level}                         tinyint,
{DoclibRowId}                   int,
{VirusVendorID}                 int,
{VirusStatus}                   int,
{VirusInfo}                     nvarchar(255),
{VirusInfoEx}                   varbinary(max),
{ContentVersion}                int,
{NextBSN}                       bigint,
{StreamSchema}                  byte,
{SiteId}                        uniqueidentifier
```

{Size}: The size of the document stream, in bytes.

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

{Version}: The current **displayed version** number of the document.

{InternalVersion}: The current **internal version number** of the document.

{HistVersion}: The user interface (UI) version number for the document.

{Id}: The **document identifier** of the requested document.

{DirName}: The **directory name** of the requested document.

{LeafName}: The **leaf name** of the requested document.

{ParentId}: The document identifier of the item's parent container.

{SetupPathVersion}: For an **uncustomized 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 uncustomized. 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).
15	Relative to the install location of Microsoft SharePoint Foundation 2013 on the front-end Web server (for example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\15).

{SetupPath}: For a document that has never been uncustomized, 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.

{Dirty}: This parameter MUST be set to one, if this document has had dependency update processing performed; otherwise, it MUST be set to zero. If the document does not have a content stream, the value is implementation-dependent and MUST be ignored.

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

{Level}: A publishing level value specifying the publishing status of this document.

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

{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 2.2.1.2.17) 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}: An anti-virus scanner specific message returned by the anti-virus scanner when it last processed the document.

{VirusInfoEx}: A binary value that specifies extensible virus information for this document, which is opaque to the back-end server.

{ContentVersion}: A counter used for internal conflict detection that is incremented any time a change is made to the binary contents of this document.

{NextBSN}: The current binary large object (BLOB) sequence number of the requested document.

{StreamSchema}: The current **stream schema** of the document being requested.

{SiteId}: The **site collection identifier** of the site collection containing the document being requested.

2.2.4.7 Document Content Stream Result Set

The **Document Content Stream Result Set** contains a row for each of the document's **stream binary pieces**.

```
{ExpirationUTC}              datetime,
{DocId}                      uniqueidentifier;
{SiteId}                     uniqueidentifier,
{Partition}                  byte,
{BSN}                        bigint,
{StreamId}                   bigint,
{Type}                       byte,
{Size}                       int,
{Content}                    varbinary(max),
{RbsResRefrence}             varbinary(800)
```

{ExpirationUTC}: The time stamp, in **UTC** format, of when this stream binary piece **expires**.

{DocId}: The **document identifier** of the requested document.

{SiteId}: The **site collection identifier** of the **site collection** containing the requested document.

{Partition}: The **stream partition** that this stream binary piece belongs to.

{BSN}: The **BLOB** sequence number of this stream binary piece.

{StreamId}: The **stream identifier** of this stream binary piece.

{Type}: The **stream** type of this stream binary piece.

{Size}: The size of the stream binary piece, in bytes.

{Content}: The data of the stream binary piece.

{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.

2.2.4.8 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),
VirusInfoEx,                    varbinary(max),
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,
{NextBSN}                       bigint,
{StreamSchema}                  byte,
{InternalVersion}               int
```

Id: The **document identifier** 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 used for internal conflict detection that is incremented any time a change is made to this document.

DocFlags: A **Doc Flags** value (section 2.2.2.3) 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}: This value MUST be 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, 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 the **Virus Status** (section 2.2.1.2.17).

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.

VirusInfoEx: A binary value that specifies extensible virus information for this document, which is opaque to the back-end server.

SetupPathVersion: For an **uncustomized** 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).
15	The **SetupPath** is relative to the install location of Microsoft SharePoint Foundation 2013 on the front-end Web server (for example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\15).

SetupPath: For a document that is now or once was uncustomized, 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 uncustomized, this contains the login name of the user that created the uncustomized 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 the 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** value (section 2.2.2.5) describing the list that contains this document.

Acl: The binary serialization of the **WSS ACL Format** (section 2.2.3.6) **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 2.2.2.15) 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 2.2.1.1.13) 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** value (section 2.2.2.6) 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 and the original document is in the same folder as the transformed document, this is the leaf name of the original 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 editors 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 identifier for a row in a document library for this document. If the requested document is not contained in a list, this value MUST be NULL.

{tp_DefaultWorkflowId}: The **Workflow Identifier** (section 2.2.1.1.16) corresponding to the **workflow** to be invoked if this document is in a moderated list and is submitted for approval as part of a **check in**. This value MUST be NULL.

ListId: The **List Identifier** (section 2.2.1.1.5) 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 used for internal conflict detection that is incremented any time a change is made to the metadict for this document.

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

ContentVersion: A counter used for internal conflict detection that is incremented any time a change is made to the binary contents of this document. 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 zero; otherwise, the value MUST be one.

{NextBSN}: The current **BLOB** sequence number of the document being returned.

{StreamSchema}: The current **stream schema** of the document being returned.

{InternalVersion}: The **internal version number** of the document being returned.

2.2.4.9 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 be returned only 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,
TimeCreated                 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),
VirusInfoEx,                varbinary(max),
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,
NextBSN                         bigint,
StreamSchema                    byte,
InternalVersion                 int
```

Id: The **document identifier** 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** (section 2.2.2.4).

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

TimeCreated: 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 used for internal conflict detection that is incremented any time a change is made to this document.

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}: This value MUST be 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, 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 the **Virus Status** section 2.2.1.2.17.

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.

VirusInfoEx: A binary value that specifies extensible virus information for this document, which is opaque to the back-end server.

SetupPathVersion: For an **uncustomized** 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).
15	The **SetupPath** is relative to the install location of Microsoft SharePoint Foundation 2013 on the front-end Web server (for example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\15).

SetupPath: For a document that is now or once was uncustomized, 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 uncustomized, this contains the login name of the user that created the uncustomized 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, 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** value (section 2.2.2.5) describing the list that contains this document. This value MUST be NULL.

Acl: The binary serialization of the **WSS ACL Format** (section 2.2.3.6) **ACL** for this document. This is either explicitly defined or inherited from the parent object of the document.

AnonymousPermMask: A **WSS Rights Mask** (section 2.2.2.15) 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 2.2.1.1.13) 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** (section 2.2.2.6) 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), 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, 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 and the original document is in the same folder as the transformed document, this is the leaf name of the original 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 editors 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 identifier for a row in a document library 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** (section 2.2.1.1.16) 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 2.2.1.1.5) 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 used for internal conflict detection that is incremented any time a change is made to the binary contents of this document. 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 zero; otherwise, it MUST be one.

{NextBSN}: The current **BLOB** sequence number of the document being returned.

{StreamSchema}: The current **stream schema** of the document being returned.

{InternalVersion}: The **internal version number** of the document being returned.

2.2.4.10 Empty Result Set

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

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

2.2.4.11 Event Receivers Result Set

The **Event Receivers Result Set** contains information about the **event receivers** defined for an **event host**. This 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,
RemoteUrl                   nvarchar(4000),
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** (section 2.2.1.1.3) of the event receiver.

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

SiteId: The **site collection identifier** for a **site collection** that contains the event host on which the event receiver is registered.

WebId: The **site identifier** for the **site (2)** that contains the event host on which the event receiver is registered.

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

HostType: The **Event Host Type** (section 2.2.1.2.5) 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. The value MUST be one of the following values.

Value	Description
0	If the **Event Receiver Type** (section 2.2.1.2.6) 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 zero, and less than or equal to 65535.

RemoteUrl: The **URL** of the remote event receiver.

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** or **content type identifier** 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 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** 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.

2.2.4.12 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 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** 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 to write the document; otherwise, it MUST be 0.

{BaseType}: The **List Base Type** 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 of the list or document library containing the document location.

{Level}: The **Publishing Level Type** 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 of the **principal (2)** 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.

2.2.4.13 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 which 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** 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.

2.2.4.14 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),
Followable                          bit;
```

tp_Id: The List Identifier 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** 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 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 (2)** or list has been flagged to cache all schema data; otherwise, it contains a Compressed structure. 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 2.2.7.3.5.

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 is left to right.
2	Text flow is right to left.

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

tp_Flags: A **List Flags** 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** 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, this flag is set to 0.

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

Acl: The binary serialization of the WSS ACL Format access control list (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 **e-mail address** of the list. This e-mail address is used to allow files to be sent directly to this list through an implementation-specific e-mail handling feature. This value MUST default to NULL if no e-mail inserts server has been associated with the list.

tp_WebFullUrl: The complete store-relative form URL to the site (2) that contains this list.

tp_WebId: The Site Identifier of the site (2) that contains the list.

tp_WebTitle: The title, for display in the front-end Web server, of the site (2) that contains this list.

tp_Web: The identifier of the **site template** for the site (2) that contains this list.

tp_WebLanguage: The **language code identifier (LCID)** of the display language of the site (2) that contains this list.

tp_WebCollation: The **collation order** for information in the site (2) 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.

{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 aren't automatically 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 aren't automatically 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 corresponding to the workflow (2) 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 (2), this value MUST be NULL.

{HasBackLookupRels}: This bit is set to 1 if this list has any relationship lookup **fields** with relationship delete behavior that cascades or relationship delete behavior that is restricted. Otherwise, it MUST be 0.

tp_ContentTypes: Compressed structure. 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 (2).

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

Followable: If this list is followable, then this value MUST be 1. Otherwise, it MUST be 0.

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**.

2.2.4.15 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 of the list containing the Web Part.

tp_Type: The **Page Type** (section 2.2.1.2.14) of the Web Part.

tp_Id: The **Web Part Identifier** (section 2.2.1.1.15) of the Web Part.

tp_Flags: A **View Flags** (section 2.2.2.13) 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 (2)** that contains this Web Part.

tp_BaseViewId: The **View Identifier** (section 2.2.1.1.14) 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 **Compressed Structure** (section 2.2.3.8), 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 zero.

tp_PageUrlID: The **document identifier** 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.

2.2.4.16 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}: MUST be NULL.

{Acl}: MUST be NULL.

{AnonymousPermMask}: MUST be NULL.

{IsAttachment}: MUST be 0.

{NeedManageListRight}: MUST be 0.

{BaseType}: MUST be NULL.

{ExcludedType}: MUST be NULL.

{ListFlags}: MUST be NULL.

{Level}: MUST be NULL.

{DraftOwnerId}: MUST be NULL.

{DocType}: MUST be NULL.

{bNeedNoThrottle}: MUST be NULL.

{ItemCount}: MUST be NULL.

{DoclibRowId}: MUST be 0.

{DocFlags}: MUST be 0.

2.2.4.17 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.

2.2.4.18 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.

2.2.4.19 Principal User Information Result Set

The **Principal User Information Result Set** returns information about a **principal (1)**. 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 of the principal. MUST NOT be NULL.

tp_SystemID: The SystemId 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 e-mail 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 User Information flags value of the principal. This parameter MUST NOT be NULL.

2.2.4.20 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.

2.2.4.21 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** 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 **LinkInfo Types** (section 2.2.1.2.10) 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 one specifies a server-relative URL. This value MUST be NULL for a backward link.

LinkStatus: The **Document Store Type** value (section 2.2.2.4) 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 this link 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 one so that the link can be distinguished from an explicit link to the welcome page; otherwise, this value MUST be zero.

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.

2.2.4.22 Site Audit Mask Result Set

The **Site Audit Mask Result Set** contains the context-sensitive identifier for a specified object, and the **Audit Flags** 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** of the object determines the type of identifier used for this value (for example, a Site Identifier or a List Identifier). 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.

2.2.4.23 Site Feature List Result Set

The **Site Feature List Result Set** contains the list of Feature Identifiers of a **site (2)**.

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).

2.2.4.24 Site Metadata Result Set

The **Site Metadata Result Set** contains metadata for a **site (2)** 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,
tp_MUILanguages              varchar(64),
tp_ContentLanguages          varchar(64),
{SiteHashKey}                binary(16),
{UserInfoListId}             uniqueidentifier,
{RootWebId}                  uniqueidentifier,
{SiteLastContentChange}      datetime,
{SiteLastSecurityChange}     datetime,
{RbsCollectionId}            int,
{DenyPermMask}               tPermMask;
```

Id: The Site Identifier for the site (2).

Title: The title of the site (2) for display in the front-end Web server.

Description: The description of the site (2) for display in the front-end Web server.

MetaInfoVersion: A counter incremented any time a change is made to the Metainfo for this site (2) 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 (2).

Language: The LCID associated with the site (2) which 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 (2) which 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 (2) which indicates an additional sorting order that is 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 2.2.1.3) for the time zone to be used when displaying time values for this site (2).

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** that is to be used when processing date values for this site (2).

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 (2).

AltCalendarType: The **Calendar Type** of an alternate calendar for processing date values for this site (2). If NULL, only the **CalendarType** value is used for this site (2).

CalendarViewOptions: A **Calendar View Options Type** that specifies the calendar display options setting for this site (2).

WorkDays: A set of **Workdays Flags** that specify the weekdays defined as the workweek for this site (2).

WorkDayStartHour: The start time of the workday in minutes after midnight for this site (2).

WorkDayEndHour: The end time of the workday in minutes after midnight for this site (2).

ProvisionConfig: An identifier of the site template used to provision this site (2). The following reserved site template identifiers are defined.

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

Flags: A **Site Property Flags** value describing the configuration of this site (2).

Author: The User Identifier of the user who is listed as creating this site (2).

AlternateCSSUrl: The URL for a custom CSS sheet file registered on the site (2) for use in pages of the site.

CustomizedCss: This contains an implementation-specific list of custom CSS files associated with this site (2).

CustomJSUrl: The URL for a custom JavaScript file registered on the site (2) for use in pages of the site.

AlternateHeaderUrl: The URL for a custom header HTML page registered on the site (2) 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 (2). This is NULL for sites using the native security implementation.

MasterUrl: The URL for the **master page** registered on the site (2) 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 (2) for use in pages of the site rendered on the front-end Web server.

{SiteLogoUrl}: The URL of an image that represents the site (2) for display in the front-end Web server.

{SiteLogoDescription}: The description of the image that represents the site (2) 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 (2) whose metadata is to be updated.

@DescriptionResource: The resource token or resource identifier of the description for the site (2) 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 (2). 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 automatically overwrite the existing translations made in all alternate languages.

@UIVersion: A number that represents the visual state of the site (2).

@ClientTag: A number that represents the state of the application files of the site (2).

AnonymousPermMask: A **WSS Rights Mask** that indicates the permissions granted to a user that is anonymous, or has no specific permissions, to this site (2). The value MUST be NULL for sites that do not exist.

{SiteFlags}: A **Site Collection Flags** value that indicates the settings for the site collection that contains this site (2).

{SitePortalURL}: The URL for a different site (2) registered on the site for use in **navigation structures** as a parent location.

{SitePortalName}: The display name of a different site (2) registered on the site for use in navigation structures as a parent location.

MeetingCount: If this site (2) 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 (2).

CachedNav: An implementation-specific serialization of navigation structure information to display in front-end Web server elements associated with this site (2).

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 (2).

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 (2) 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 (2) in this site's ancestor chain that does not inherit security settings from its **parent site**.

ScopeId: The Scope Identifier of the security scope containing the site (2).

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

Acl: The binary serialization of the WSS ACL Format ACL for this site (2). This is either explicitly defined or inherited from the parent object of the site (2).

{RequestAccessEmail}: The e-mail address for a user who has the authority to grant access to the site (2), if any. This can be used by the front-end Web server to generate an e-mail to request access for a user who does not have access to the site (2).

Ancestry: An implementation-specific serialization of the navigation structure information for the site (2).

ProductVersion: The implementation-specific version identifier used to create the site (2).

tp_Id: The User Identifier of the current user.

tp_SiteAdmin: Set to "1" if the current user is an administrator of the site (2).

tp_IsActive: Set to "1" if the current user is an active user in the site collection containing this site (2).

tp_Login: The login information of the current user.

tp_Email: The e-mail 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** for the current user was last updated.

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

tp_Token: A WSS User Token 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 (2).

{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 which 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.

tp_MUILanguages: A string that contains the distinct culture(s) for the user's preferred display language(s), separated by semicolon. This string can be NULL.

tp_ContentLanguages: A string that contains the distinct culture(s) for the user's preferred content language(s), separated by semicolon. This string can be NULL.

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

{UserInfoListId}: The **list identifier** of the user information list for the site collection containing this site (2).

{RootWebId}: The **site identifier** for the root site of the site collection containing this site (2).

{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.

{DenyPermMask}: A **WSS Rights Mask** (section 2.2.2.15) specifying the rights to deny to the current user.

2.2.4.25 Site MetaInfo Result Set

The **Site MetaInfo Result Set** returns metainfo about a **site (2)**.

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.

2.2.4.26 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 of the security scope.

Acl: The binary serialization of the WSS ACL Format ACL of the security scope.

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

2.2.4.27 URL Result Set

The **URL Result Set** returns a list of URLs, one for the root folder of each **site (2)**. 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).

2.2.4.28 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 this 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 of the **site (2)** which contains the relationship lookup field.

tp_Id: The list identifier of the list which contains the relationship lookup field.

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

DeleteBehavior: The relationship delete behavior on the relationship lookup field

2.2.4.29 App Principal Information Result Set

The **App Principal Information Result Set** returns information about an **app principal**.

If the **App Principal Info Result Set** is returned, it MUST return one row, as defined using the T-SQL syntax below:

```
{AppPrincipalName}              nvarchar(256),
{ReturnAppPrincipalInfo}        bit,
{WebTemplateId}                 int,
{WebTemplateConfigurationId}    int
```

{AppPrincipalName}: The app principal identifier.

{ReturnAppPrincipalInfo}: This value MUST be 1.

{WebTemplateId}: The identifier for the site template used in the **site definition** associated with the **stored procedure**.

{WebTemplateConfigurationId}: The value as defined by **ProvisionConfig** in section 2.2.4.24.

2.2.4.30 App Principal Permissions Result Set

The **App Principal Permissions Result Set** is returned to indicate what permissions the specified **app principal** has.

If the **App Principal Permissions Result Set** is returned, it MUST return at least one row, as defined using the T-SQL syntax below:

```
{Id}            int,
{Name}          nvarchar(256),
{Title}         nvarchar(256),
{Flag}          int,
{WebId}         uniqueidentifier,
```

```
{ListId}        uniqueidentifier,
{Perm}          int
```

{Id}: The app principal integer identifier in the **site collection**.

{Name}: The app principal identifier.

{Title}: The title of the app principal.

{Flag}: The app principal flag that specifies the state of the app principal. All valid values for this flag are specified in the following table.

Value	Description
0	No flags.
1	The app principal has been disabled.
2	The app principal has no tenant scoped permissions.
4	Allow App Only Policy.

{WebId}: The **site identifier** of the **site (2)** containing the document.

{ListId}: The list identifier of the list containing the document.

{Perm}: The permission value as defined in section 2.2.1.2.18.

2.2.5 SQL Structures

2.2.5.1 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.

2.2.5.1.1 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.

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. 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

2.2.5.1.2 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. **Id** is defined in T-SQL format as follows.

```
Id                              uniqueidentifier NOT NULL
```

2.2.5.1.3 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. **Name** is defined in T-SQL format as follows.

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

2.2.5.1.4 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. 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
```

2.2.5.1.5 Status

The **Status** property of a Configuration Object MUST conform to the data type defined in the Configuration Object **Status** section.

```
Status          int NOT NULL
```

2.2.5.1.6 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
```

2.2.5.1.7 Properties

Any configuration settings needed by an application that are not stored in one of the other Configuration Object field 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.

2.2.5.1.7.1 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.

2.2.5.1.7.2 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.

2.2.5.1.7.3 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

Property	XPath Query	Description
		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.

2.2.5.2 Dependencies

In addition to the ancestry hierarchy defined in **Parent** (section 2.2.5.1.4), 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.

2.2.6 Tables and Views

2.2.6.1 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 AS CASE WHEN BumpVersion <> 0 THEN CAST(1 as bit) ELSE CAST(0 as bit) END,
ListDataDirty                  bit              NOT NULL DEFAULT 0,
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,
VirusInfoEx,                   varbinary(max)   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,
NextBSN                         bigint            NULL,
StreamSchema                    tinyint           NULL);
```

Id: The **document identifier** of the document.

SiteId: The **Site Collection Identifier** (section 2.2.1.1.9) 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 2.2.1.1.11) of the site containing the document.

ListId: The **List Identifier** (section 2.2.1.1.5) 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 is sorted within a view. If this parameter is set to 1, 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 (2)**, the value MUST be 2. Other values are invalid.

Size: The size of the document stream in bytes. This parameter can be set to NULL or to zero for items such as sites, folders, document libraries, and lists.

ETagVersion: A counter used for conflict detection that is incremented any time a change is made to this document. 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 only used 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 only used 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 one if the document has dependencies that require further processing.

ListDataDirty: Set to one 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** value (section 2.2.2.3) specifying information about the document. See the **Doc Flags** section for details.

ThicketFlag: If set to one, 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 editors 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 2.2.6.3), **DeleteTransactionId** MUST equal 0x, because the **Docs View** only displays items that are not marked as deleted.

SetupPathVersion: For an **uncustomized** document, this parameter governs the **SetupPath** fragment relative to the setup path location. The following values are valid.

Value	Description
NULL	This is not an uncustomized document.

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).
15	The **SetupPath** is relative to the install location of Microsoft SharePoint Foundation 2013 on the front-end Web server (for example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\15).

SetupPath: For a document that is now or once was uncustomized, 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 uncustomized.

SetupPathUser: If this document is now or once was uncustomized, this contains the login name of the user who created the uncustomized document. This value is undefined for documents that were never uncustomized.

CheckoutUserId: If the document is checked out, this parameter contains the **User Identifier** (section 2.2.1.1.13) 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 one, 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, 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 2.2.1.2.17) 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.

VirusInfoEx: A binary value that specifies extensible virus information for this document, which is opaque to the back-end server.

MetaInfo: A **metadict** for the document. The metadict format is specified in the [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, 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 one, 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** value (section 2.2.2.6)). 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** value (section 2.2.2.1) 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, this is the **User Identifier** (section 2.2.1.1.13) 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 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 zero (file) and it is allowed to contain a stream.
- The **SetupPath** is not NULL and the document content is **customized**.

If neither condition applies, the value MUST be "0".

ScopeId: The **Scope Identifier** (section 2.2.1.1.8) 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 zero indicates a document with no dependencies.

ParentVersion: If the document is a transformed version of another document, 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, 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 and the original document is in the same folder as the transformed document, this is the leaf name of the original 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, the **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 used for internal conflict detection that is incremented any time a change is made to the binary contents of this document.

ListSchemaVersion: A counter which 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.

NextBSN: The current **BLOB** sequence number for the document, used for assigning BLOB sequence numbers to **stream binary pieces** belonging to this document.

StreamSchema: The **stream schema** of the stream belonging to this document, if it has one indicated by **HasStream**.

2.2.6.2 AllUserData Table

The **AllUserData** table stores data for all **list (1)** 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 (1) 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 is accessed by means of joins with the **Docs View** (section 2.2.6.3).

The **AllUserData** 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            tContentTypeId     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_AppAuthor            int                    NULL,
    tp AppEditor            int                    NULL,
    bit1                    bit            SPARSE  NULL,
    bit2                    bit            SPARSE  NULL,
    bit3                    bit            SPARSE  NULL,
    bit4                    bit            SPARSE  NULL,
    bit5                    bit            SPARSE  NULL,
    bit6                    bit            SPARSE  NULL,
    bit7                    bit            SPARSE  NULL,
    bit8                    bit            SPARSE  NULL,
    bit9                    bit            SPARSE  NULL,
    bit10                   bit            SPARSE  NULL,
    bit11                   bit            SPARSE  NULL,
```

```
    bit12                bit             SPARSE  NULL,
    bit13                bit             SPARSE  NULL,
    bit14                bit             SPARSE  NULL,
    bit15                bit             SPARSE  NULL,
    bit16                bit             SPARSE  NULL,
    bit17                bit             SPARSE  NULL,
    bit18                bit             SPARSE  NULL,
    bit19                bit             SPARSE  NULL,
    bit20                bit             SPARSE  NULL,
    bit21                bit             SPARSE  NULL,
    bit22                bit             SPARSE  NULL,
    bit23                bit             SPARSE  NULL,
    bit24                bit             SPARSE  NULL,
    bit25                bit             SPARSE  NULL,
    bit26                bit             SPARSE  NULL,
    bit27                bit             SPARSE  NULL,
    bit28                bit             SPARSE  NULL,
    bit29                bit             SPARSE  NULL,
    bit30                bit             SPARSE  NULL,
    bit31                bit             SPARSE  NULL,
    bit32                bit             SPARSE  NULL,
    bit33                bit             SPARSE  NULL,
    bit34                bit             SPARSE  NULL,
    bit35                bit             SPARSE  NULL,
    bit36                bit             SPARSE  NULL,
    bit37                bit             SPARSE  NULL,
    bit38                bit             SPARSE  NULL,
    bit39                bit             SPARSE  NULL,
    bit40                bit             SPARSE  NULL,
    bit41                bit             SPARSE  NULL,
    bit42                bit             SPARSE  NULL,
    bit43                bit             SPARSE  NULL,
    bit44                bit             SPARSE  NULL,
    bit45                bit             SPARSE  NULL,
    bit46                bit             SPARSE  NULL,
    bit47                bit             SPARSE  NULL,
    bit48                bit             SPARSE  NULL,
    bit49                bit             SPARSE  NULL,
    bit50                bit             SPARSE  NULL,
    bit51                bit             SPARSE  NULL,
    bit52                bit             SPARSE  NULL,
    bit53                bit             SPARSE  NULL,
    bit54                bit             SPARSE  NULL,
    bit55                bit             SPARSE  NULL,
    bit56                bit             SPARSE  NULL,
    bit57                bit             SPARSE  NULL,
    bit58                bit             SPARSE  NULL,
    bit59                bit             SPARSE  NULL,
    bit60                bit             SPARSE  NULL,
    bit61                bit             SPARSE  NULL,
    bit62                bit             SPARSE  NULL,
    bit63                bit             SPARSE  NULL,
    bit64                bit             SPARSE  NULL,
    bit65                bit             SPARSE  NULL,
    bit66                bit             SPARSE  NULL,
    bit67                bit             SPARSE  NULL,
    bit68                bit             SPARSE  NULL,
    bit69                bit             SPARSE  NULL,
    bit70                bit             SPARSE  NULL,
    bit71                bit             SPARSE  NULL,
    bit72                bit             SPARSE  NULL,
    bit73                bit             SPARSE  NULL,
    bit74                bit             SPARSE  NULL,
    bit75                bit             SPARSE  NULL,
    bit76                bit             SPARSE  NULL,
    bit77                bit             SPARSE  NULL,
    bit78                bit             SPARSE  NULL,
    bit79                bit             SPARSE  NULL,
    bit80                bit             SPARSE  NULL,
```

```
    bit81               bit             SPARSE  NULL,
    bit82               bit             SPARSE  NULL,
    bit83               bit             SPARSE  NULL,
    bit84               bit             SPARSE  NULL,
    bit85               bit             SPARSE  NULL,
    bit86               bit             SPARSE  NULL,
    bit87               bit             SPARSE  NULL,
    bit88               bit             SPARSE  NULL,
    bit89               bit             SPARSE  NULL,
    bit90               bit             SPARSE  NULL,
    bit91               bit             SPARSE  NULL,
    bit92               bit             SPARSE  NULL,
    bit93               bit             SPARSE  NULL,
    bit94               bit             SPARSE  NULL,
    bit95               bit             SPARSE  NULL,
    bit96               bit             SPARSE  NULL,
    bit97               bit             SPARSE  NULL,
    bit98               bit             SPARSE  NULL,
    bit99               bit             SPARSE  NULL,
    bit100              bit             SPARSE  NULL,
    bit101              bit             SPARSE  NULL,
    bit102              bit             SPARSE  NULL,
    bit103              bit             SPARSE  NULL,
    bit104              bit             SPARSE  NULL,
    bit105              bit             SPARSE  NULL,
    bit106              bit             SPARSE  NULL,
    bit107              bit             SPARSE  NULL,
    bit108              bit             SPARSE  NULL,
    bit109              bit             SPARSE  NULL,
    bit110              bit             SPARSE  NULL,
    bit111              bit             SPARSE  NULL,
    bit112              bit             SPARSE  NULL,
    bit113              bit             SPARSE  NULL,
    bit114              bit             SPARSE  NULL,
    bit115              bit             SPARSE  NULL,
    bit116              bit             SPARSE  NULL,
    bit117              bit             SPARSE  NULL,
    bit118              bit             SPARSE  NULL,
    bit119              bit             SPARSE  NULL,
    bit120              bit             SPARSE  NULL,
    bit121              bit             SPARSE  NULL,
    bit122              bit             SPARSE  NULL,
    bit123              bit             SPARSE  NULL,
    bit124              bit             SPARSE  NULL,
    bit125              bit             SPARSE  NULL,
    bit126              bit             SPARSE  NULL,
    bit127              bit             SPARSE  NULL,
    bit128              bit             SPARSE  NULL,
    bit129              bit             SPARSE  NULL,
    bit130              bit             SPARSE  NULL,
    bit131              bit             SPARSE  NULL,
    bit132              bit             SPARSE  NULL,
    bit133              bit             SPARSE  NULL,
    bit134              bit             SPARSE  NULL,
    bit135              bit             SPARSE  NULL,
    bit136              bit             SPARSE  NULL,
    bit137              bit             SPARSE  NULL,
    bit138              bit             SPARSE  NULL,
    bit139              bit             SPARSE  NULL,
    bit140              bit             SPARSE  NULL,
    bit141              bit             SPARSE  NULL,
    bit142              bit             SPARSE  NULL,
    bit143              bit             SPARSE  NULL,
    bit144              bit             SPARSE  NULL,
    bit145              bit             SPARSE  NULL,
    bit146              bit             SPARSE  NULL,
    bit147              bit             SPARSE  NULL,
    bit148              bit             SPARSE  NULL,
    bit149              bit             SPARSE  NULL,
```

```
    bit150                bit             SPARSE  NULL,
    bit151                bit             SPARSE  NULL,
    bit152                bit             SPARSE  NULL,
    bit153                bit             SPARSE  NULL,
    bit154                bit             SPARSE  NULL,
    bit155                bit             SPARSE  NULL,
    bit156                bit             SPARSE  NULL,
    bit157                bit             SPARSE  NULL,
    bit158                bit             SPARSE  NULL,
    bit159                bit             SPARSE  NULL,
    bit160                bit             SPARSE  NULL,
    bit161                bit             SPARSE  NULL,
    bit162                bit             SPARSE  NULL,
    bit163                bit             SPARSE  NULL,
    bit164                bit             SPARSE  NULL,
    bit165                bit             SPARSE  NULL,
    bit166                bit             SPARSE  NULL,
    bit167                bit             SPARSE  NULL,
    bit168                bit             SPARSE  NULL,
    bit169                bit             SPARSE  NULL,
    bit170                bit             SPARSE  NULL,
    bit171                bit             SPARSE  NULL,
    bit172                bit             SPARSE  NULL,
    bit173                bit             SPARSE  NULL,
    bit174                bit             SPARSE  NULL,
    bit175                bit             SPARSE  NULL,
    bit176                bit             SPARSE  NULL,
    bit177                bit             SPARSE  NULL,
    bit178                bit             SPARSE  NULL,
    bit179                bit             SPARSE  NULL,
    bit180                bit             SPARSE  NULL,
    bit181                bit             SPARSE  NULL,
    bit182                bit             SPARSE  NULL,
    bit183                bit             SPARSE  NULL,
    bit184                bit             SPARSE  NULL,
    bit185                bit             SPARSE  NULL,
    bit186                bit             SPARSE  NULL,
    bit187                bit             SPARSE  NULL,
    bit188                bit             SPARSE  NULL,
    bit189                bit             SPARSE  NULL,
    bit190                bit             SPARSE  NULL,
    bit191                bit             SPARSE  NULL,
    bit192                bit             SPARSE  NULL,
    bit193                bit             SPARSE  NULL,
    bit194                bit             SPARSE  NULL,
    bit195                bit             SPARSE  NULL,
    bit196                bit             SPARSE  NULL,
    bit197                bit             SPARSE  NULL,
    bit198                bit             SPARSE  NULL,
    bit199                bit             SPARSE  NULL,
    bit200                bit             SPARSE  NULL,
    bit201                bit             SPARSE  NULL,
    bit202                bit             SPARSE  NULL,
    bit203                bit             SPARSE  NULL,
    bit204                bit             SPARSE  NULL,
    bit205                bit             SPARSE  NULL,
    bit206                bit             SPARSE  NULL,
    bit207                bit             SPARSE  NULL,
    bit208                bit             SPARSE  NULL,
    bit209                bit             SPARSE  NULL,
    bit210                bit             SPARSE  NULL,
    bit211                bit             SPARSE  NULL,
    bit212                bit             SPARSE  NULL,
    bit213                bit             SPARSE  NULL,
    bit214                bit             SPARSE  NULL,
    bit215                bit             SPARSE  NULL,
    bit216                bit             SPARSE  NULL,
    bit217                bit             SPARSE  NULL,
    bit218                bit             SPARSE  NULL,
```

```
    bit219                bit             SPARSE  NULL,
    bit220                bit             SPARSE  NULL,
    bit221                bit             SPARSE  NULL,
    bit222                bit             SPARSE  NULL,
    bit223                bit             SPARSE  NULL,
    bit224                bit             SPARSE  NULL,
    bit225                bit             SPARSE  NULL,
    bit226                bit             SPARSE  NULL,
    bit227                bit             SPARSE  NULL,
    bit228                bit             SPARSE  NULL,
    bit229                bit             SPARSE  NULL,
    bit230                bit             SPARSE  NULL,
    bit231                bit             SPARSE  NULL,
    bit232                bit             SPARSE  NULL,
    bit233                bit             SPARSE  NULL,
    bit234                bit             SPARSE  NULL,
    bit235                bit             SPARSE  NULL,
    bit236                bit             SPARSE  NULL,
    bit237                bit             SPARSE  NULL,
    bit238                bit             SPARSE  NULL,
    bit239                bit             SPARSE  NULL,
    bit240                bit             SPARSE  NULL,
    bit241                bit             SPARSE  NULL,
    bit242                bit             SPARSE  NULL,
    bit243                bit             SPARSE  NULL,
    bit244                bit             SPARSE  NULL,
    bit245                bit             SPARSE  NULL,
    bit246                bit             SPARSE  NULL,
    bit247                bit             SPARSE  NULL,
    bit248                bit             SPARSE  NULL,
    bit249                bit             SPARSE  NULL,
    bit250                bit             SPARSE  NULL,
    bit251                bit             SPARSE  NULL,
    bit252                bit             SPARSE  NULL,
    bit253                bit             SPARSE  NULL,
    bit254                bit             SPARSE  NULL,
    bit255                bit             SPARSE  NULL,
    bit256                bit             SPARSE  NULL,
    bit257                bit             SPARSE  NULL,
    bit258                bit             SPARSE  NULL,
    bit259                bit             SPARSE  NULL,
    bit260                bit             SPARSE  NULL,
    bit261                bit             SPARSE  NULL,
    bit262                bit             SPARSE  NULL,
    bit263                bit             SPARSE  NULL,
    bit264                bit             SPARSE  NULL,
    bit265                bit             SPARSE  NULL,
    bit266                bit             SPARSE  NULL,
    bit267                bit             SPARSE  NULL,
    bit268                bit             SPARSE  NULL,
    bit269                bit             SPARSE  NULL,
    bit270                bit             SPARSE  NULL,
    bit271                bit             SPARSE  NULL,
    bit272                bit             SPARSE  NULL,
    bit273                bit             SPARSE  NULL,
    bit274                bit             SPARSE  NULL,
    bit275                bit             SPARSE  NULL,
    bit276                bit             SPARSE  NULL,
    bit277                bit             SPARSE  NULL,
    bit278                bit             SPARSE  NULL,
    bit279                bit             SPARSE  NULL,
    bit280                bit             SPARSE  NULL,
    bit281                bit             SPARSE  NULL,
    bit282                bit             SPARSE  NULL,
    bit283                bit             SPARSE  NULL,
    bit284                bit             SPARSE  NULL,
    bit285                bit             SPARSE  NULL,
    bit286                bit             SPARSE  NULL,
    bit287                bit             SPARSE  NULL,
```

```
bit288                bit             SPARSE  NULL,
bit289                bit             SPARSE  NULL,
bit290                bit             SPARSE  NULL,
bit291                bit             SPARSE  NULL,
bit292                bit             SPARSE  NULL,
bit293                bit             SPARSE  NULL,
bit294                bit             SPARSE  NULL,
bit295                bit             SPARSE  NULL,
bit296                bit             SPARSE  NULL,
bit297                bit             SPARSE  NULL,
bit298                bit             SPARSE  NULL,
bit299                bit             SPARSE  NULL,
bit300                bit             SPARSE  NULL,
bit301                bit             SPARSE  NULL,
bit302                bit             SPARSE  NULL,
bit303                bit             SPARSE  NULL,
bit304                bit             SPARSE  NULL,
bit305                bit             SPARSE  NULL,
bit306                bit             SPARSE  NULL,
bit307                bit             SPARSE  NULL,
bit308                bit             SPARSE  NULL,
bit309                bit             SPARSE  NULL,
bit310                bit             SPARSE  NULL,
bit311                bit             SPARSE  NULL,
bit312                bit             SPARSE  NULL,
bit313                bit             SPARSE  NULL,
bit314                bit             SPARSE  NULL,
bit315                bit             SPARSE  NULL,
bit316                bit             SPARSE  NULL,
bit317                bit             SPARSE  NULL,
bit318                bit             SPARSE  NULL,
bit319                bit             SPARSE  NULL,
bit320                bit             SPARSE  NULL,
bit321                bit             SPARSE  NULL,
bit322                bit             SPARSE  NULL,
bit323                bit             SPARSE  NULL,
bit324                bit             SPARSE  NULL,
bit325                bit             SPARSE  NULL,
bit326                bit             SPARSE  NULL,
bit327                bit             SPARSE  NULL,
bit328                bit             SPARSE  NULL,
bit329                bit             SPARSE  NULL,
bit330                bit             SPARSE  NULL,
bit331                bit             SPARSE  NULL,
bit332                bit             SPARSE  NULL,
bit333                bit             SPARSE  NULL,
bit334                bit             SPARSE  NULL,
bit335                bit             SPARSE  NULL,
bit336                bit             SPARSE  NULL,
bit337                bit             SPARSE  NULL,
bit338                bit             SPARSE  NULL,
bit339                bit             SPARSE  NULL,
bit340                bit             SPARSE  NULL,
bit341                bit             SPARSE  NULL,
bit342                bit             SPARSE  NULL,
bit343                bit             SPARSE  NULL,
bit344                bit             SPARSE  NULL,
bit345                bit             SPARSE  NULL,
bit346                bit             SPARSE  NULL,
bit347                bit             SPARSE  NULL,
bit348                bit             SPARSE  NULL,
bit349                bit             SPARSE  NULL,
bit350                bit             SPARSE  NULL,
bit351                bit             SPARSE  NULL,
bit352                bit             SPARSE  NULL,
bit353                bit             SPARSE  NULL,
bit354                bit             SPARSE  NULL,
bit355                bit             SPARSE  NULL,
bit356                bit             SPARSE  NULL,
```

```
    bit357              bit             SPARSE  NULL,
    bit358              bit             SPARSE  NULL,
    bit359              bit             SPARSE  NULL,
    bit360              bit             SPARSE  NULL,
    bit361              bit             SPARSE  NULL,
    bit362              bit             SPARSE  NULL,
    bit363              bit             SPARSE  NULL,
    bit364              bit             SPARSE  NULL,
    bit365              bit             SPARSE  NULL,
    bit366              bit             SPARSE  NULL,
    bit367              bit             SPARSE  NULL,
    bit368              bit             SPARSE  NULL,
    bit369              bit             SPARSE  NULL,
    bit370              bit             SPARSE  NULL,
    bit371              bit             SPARSE  NULL,
    bit372              bit             SPARSE  NULL,
    bit373              bit             SPARSE  NULL,
    bit374              bit             SPARSE  NULL,
    bit375              bit             SPARSE  NULL,
    bit376              bit             SPARSE  NULL,
    bit377              bit             SPARSE  NULL,
    bit378              bit             SPARSE  NULL,
    bit379              bit             SPARSE  NULL,
    bit380              bit             SPARSE  NULL,
    bit381              bit             SPARSE  NULL,
    bit382              bit             SPARSE  NULL,
    bit383              bit             SPARSE  NULL,
    bit384              bit             SPARSE  NULL,
    bit385              bit             SPARSE  NULL,
    bit386              bit             SPARSE  NULL,
    bit387              bit             SPARSE  NULL,
    bit388              bit             SPARSE  NULL,
    bit389              bit             SPARSE  NULL,
    bit390              bit             SPARSE  NULL,
    bit391              bit             SPARSE  NULL,
    bit392              bit             SPARSE  NULL,
    bit393              bit             SPARSE  NULL,
    bit394              bit             SPARSE  NULL,
    bit395              bit             SPARSE  NULL,
    bit396              bit             SPARSE  NULL,
    bit397              bit             SPARSE  NULL,
    bit398              bit             SPARSE  NULL,
    bit399              bit             SPARSE  NULL,
    bit400              bit             SPARSE  NULL,
    bit401              bit             SPARSE  NULL,
    bit402              bit             SPARSE  NULL,
    bit403              bit             SPARSE  NULL,
    bit404              bit             SPARSE  NULL,
    bit405              bit             SPARSE  NULL,
    bit406              bit             SPARSE  NULL,
    bit407              bit             SPARSE  NULL,
    bit408              bit             SPARSE  NULL,
    bit409              bit             SPARSE  NULL,
    bit410              bit             SPARSE  NULL,
    bit411              bit             SPARSE  NULL,
    bit412              bit             SPARSE  NULL,
    bit413              bit             SPARSE  NULL,
    bit414              bit             SPARSE  NULL,
    bit415              bit             SPARSE  NULL,
    bit416              bit             SPARSE  NULL,
    bit417              bit             SPARSE  NULL,
    bit418              bit             SPARSE  NULL,
    bit419              bit             SPARSE  NULL,
    bit420              bit             SPARSE  NULL,
    bit421              bit             SPARSE  NULL,
    bit422              bit             SPARSE  NULL,
    bit423              bit             SPARSE  NULL,
    bit424              bit             SPARSE  NULL,
    bit425              bit             SPARSE  NULL,
```

```
    bit426                bit             SPARSE  NULL,
    bit427                bit             SPARSE  NULL,
    bit428                bit             SPARSE  NULL,
    bit429                bit             SPARSE  NULL,
    bit430                bit             SPARSE  NULL,
    bit431                bit             SPARSE  NULL,
    bit432                bit             SPARSE  NULL,
    bit433                bit             SPARSE  NULL,
    bit434                bit             SPARSE  NULL,
    bit435                bit             SPARSE  NULL,
    bit436                bit             SPARSE  NULL,
    bit437                bit             SPARSE  NULL,
    bit438                bit             SPARSE  NULL,
    bit439                bit             SPARSE  NULL,
    bit440                bit             SPARSE  NULL,
    bit441                bit             SPARSE  NULL,
    bit442                bit             SPARSE  NULL,
    bit443                bit             SPARSE  NULL,
    bit444                bit             SPARSE  NULL,
    bit445                bit             SPARSE  NULL,
    bit446                bit             SPARSE  NULL,
    bit447                bit             SPARSE  NULL,
    bit448                bit             SPARSE  NULL,
    bit449                bit             SPARSE  NULL,
    bit450                bit             SPARSE  NULL,
    bit451                bit             SPARSE  NULL,
    bit452                bit             SPARSE  NULL,
    bit453                bit             SPARSE  NULL,
    bit454                bit             SPARSE  NULL,
    bit455                bit             SPARSE  NULL,
    bit456                bit             SPARSE  NULL,
    bit457                bit             SPARSE  NULL,
    bit458                bit             SPARSE  NULL,
    bit459                bit             SPARSE  NULL,
    bit460                bit             SPARSE  NULL,
    bit461                bit             SPARSE  NULL,
    bit462                bit             SPARSE  NULL,
    bit463                bit             SPARSE  NULL,
    bit464                bit             SPARSE  NULL,
    bit465                bit             SPARSE  NULL,
    bit466                bit             SPARSE  NULL,
    bit467                bit             SPARSE  NULL,
    bit468                bit             SPARSE  NULL,
    bit469                bit             SPARSE  NULL,
    bit470                bit             SPARSE  NULL,
    bit471                bit             SPARSE  NULL,
    bit472                bit             SPARSE  NULL,
    bit473                bit             SPARSE  NULL,
    bit474                bit             SPARSE  NULL,
    bit475                bit             SPARSE  NULL,
    bit476                bit             SPARSE  NULL,
    bit477                bit             SPARSE  NULL,
    bit478                bit             SPARSE  NULL,
    bit479                bit             SPARSE  NULL,
    bit480                bit             SPARSE  NULL,
    bit481                bit             SPARSE  NULL,
    bit482                bit             SPARSE  NULL,
    bit483                bit             SPARSE  NULL,
    bit484                bit             SPARSE  NULL,
    bit485                bit             SPARSE  NULL,
    bit486                bit             SPARSE  NULL,
    bit487                bit             SPARSE  NULL,
    bit488                bit             SPARSE  NULL,
    bit489                bit             SPARSE  NULL,
    bit490                bit             SPARSE  NULL,
    bit491                bit             SPARSE  NULL,
    bit492                bit             SPARSE  NULL,
    bit493                bit             SPARSE  NULL,
    bit494                bit             SPARSE  NULL,
```

```
    bit495                bit             SPARSE  NULL,
    bit496                bit             SPARSE  NULL,
    bit497                bit             SPARSE  NULL,
    bit498                bit             SPARSE  NULL,
    bit499                bit             SPARSE  NULL,
    bit500                bit             SPARSE  NULL,
    bit501                bit             SPARSE  NULL,
    bit502                bit             SPARSE  NULL,
    bit503                bit             SPARSE  NULL,
    bit504                bit             SPARSE  NULL,
    bit505                bit             SPARSE  NULL,
    bit506                bit             SPARSE  NULL,
    bit507                bit             SPARSE  NULL,
    bit508                bit             SPARSE  NULL,
    bit509                bit             SPARSE  NULL,
    bit510                bit             SPARSE  NULL,
    bit511                bit             SPARSE  NULL,
    bit512                bit             SPARSE  NULL,
    bit513                bit             SPARSE  NULL,
    bit514                bit             SPARSE  NULL,
    bit515                bit             SPARSE  NULL,
    bit516                bit             SPARSE  NULL,
    bit517                bit             SPARSE  NULL,
    bit518                bit             SPARSE  NULL,
    bit519                bit             SPARSE  NULL,
    bit520                bit             SPARSE  NULL,
    bit521                bit             SPARSE  NULL,
    bit522                bit             SPARSE  NULL,
    bit523                bit             SPARSE  NULL,
    bit524                bit             SPARSE  NULL,
    bit525                bit             SPARSE  NULL,
    bit526                bit             SPARSE  NULL,
    bit527                bit             SPARSE  NULL,
    bit528                bit             SPARSE  NULL,
    bit529                bit             SPARSE  NULL,
    bit530                bit             SPARSE  NULL,
    bit531                bit             SPARSE  NULL,
    bit532                bit             SPARSE  NULL,
    bit533                bit             SPARSE  NULL,
    bit534                bit             SPARSE  NULL,
    bit535                bit             SPARSE  NULL,
    bit536                bit             SPARSE  NULL,
    bit537                bit             SPARSE  NULL,
    bit538                bit             SPARSE  NULL,
    bit539                bit             SPARSE  NULL,
    bit540                bit             SPARSE  NULL,
    bit541                bit             SPARSE  NULL,
    bit542                bit             SPARSE  NULL,
    bit543                bit             SPARSE  NULL,
    bit544                bit             SPARSE  NULL,
    bit545                bit             SPARSE  NULL,
    bit546                bit             SPARSE  NULL,
    bit547                bit             SPARSE  NULL,
    bit548                bit             SPARSE  NULL,
    bit549                bit             SPARSE  NULL,
    bit550                bit             SPARSE  NULL,
    bit551                bit             SPARSE  NULL,
    bit552                bit             SPARSE  NULL,
    bit553                bit             SPARSE  NULL,
    bit554                bit             SPARSE  NULL,
    bit555                bit             SPARSE  NULL,
    bit556                bit             SPARSE  NULL,
    bit557                bit             SPARSE  NULL,
    bit558                bit             SPARSE  NULL,
    bit559                bit             SPARSE  NULL,
    bit560                bit             SPARSE  NULL,
    bit561                bit             SPARSE  NULL,
    bit562                bit             SPARSE  NULL,
    bit563                bit             SPARSE  NULL,
```

```
    bit564                bit             SPARSE  NULL,
    bit565                bit             SPARSE  NULL,
    bit566                bit             SPARSE  NULL,
    bit567                bit             SPARSE  NULL,
    bit568                bit             SPARSE  NULL,
    bit569                bit             SPARSE  NULL,
    bit570                bit             SPARSE  NULL,
    bit571                bit             SPARSE  NULL,
    bit572                bit             SPARSE  NULL,
    bit573                bit             SPARSE  NULL,
    bit574                bit             SPARSE  NULL,
    bit575                bit             SPARSE  NULL,
    bit576                bit             SPARSE  NULL,
    bit577                bit             SPARSE  NULL,
    bit578                bit             SPARSE  NULL,
    bit579                bit             SPARSE  NULL,
    bit580                bit             SPARSE  NULL,
    bit581                bit             SPARSE  NULL,
    bit582                bit             SPARSE  NULL,
    bit583                bit             SPARSE  NULL,
    bit584                bit             SPARSE  NULL,
    bit585                bit             SPARSE  NULL,
    bit586                bit             SPARSE  NULL,
    bit587                bit             SPARSE  NULL,
    bit588                bit             SPARSE  NULL,
    bit589                bit             SPARSE  NULL,
    bit590                bit             SPARSE  NULL,
    bit591                bit             SPARSE  NULL,
    bit592                bit             SPARSE  NULL,
    bit593                bit             SPARSE  NULL,
    bit594                bit             SPARSE  NULL,
    bit595                bit             SPARSE  NULL,
    bit596                bit             SPARSE  NULL,
    bit597                bit             SPARSE  NULL,
    bit598                bit             SPARSE  NULL,
    bit599                bit             SPARSE  NULL,
    bit600                bit             SPARSE  NULL,
    bit601                bit             SPARSE  NULL,
    bit602                bit             SPARSE  NULL,
    bit603                bit             SPARSE  NULL,
    bit604                bit             SPARSE  NULL,
    bit605                bit             SPARSE  NULL,
    bit606                bit             SPARSE  NULL,
    bit607                bit             SPARSE  NULL,
    bit608                bit             SPARSE  NULL,
    bit609                bit             SPARSE  NULL,
    bit610                bit             SPARSE  NULL,
    bit611                bit             SPARSE  NULL,
    bit612                bit             SPARSE  NULL,
    bit613                bit             SPARSE  NULL,
    bit614                bit             SPARSE  NULL,
    bit615                bit             SPARSE  NULL,
    bit616                bit             SPARSE  NULL,
    bit617                bit             SPARSE  NULL,
    bit618                bit             SPARSE  NULL,
    bit619                bit             SPARSE  NULL,
    bit620                bit             SPARSE  NULL,
    bit621                bit             SPARSE  NULL,
    bit622                bit             SPARSE  NULL,
    bit623                bit             SPARSE  NULL,
    bit624                bit             SPARSE  NULL,
    bit625                bit             SPARSE  NULL,
    bit626                bit             SPARSE  NULL,
    bit627                bit             SPARSE  NULL,
    bit628                bit             SPARSE  NULL,
    bit629                bit             SPARSE  NULL,
    bit630                bit             SPARSE  NULL,
    bit631                bit             SPARSE  NULL,
    bit632                bit             SPARSE  NULL,
```

```
    bit633                bit             SPARSE  NULL,
    bit634                bit             SPARSE  NULL,
    bit635                bit             SPARSE  NULL,
    bit636                bit             SPARSE  NULL,
    bit637                bit             SPARSE  NULL,
    bit638                bit             SPARSE  NULL,
    bit639                bit             SPARSE  NULL,
    bit640                bit             SPARSE  NULL,
    bit641                bit             SPARSE  NULL,
    bit642                bit             SPARSE  NULL,
    bit643                bit             SPARSE  NULL,
    bit644                bit             SPARSE  NULL,
    bit645                bit             SPARSE  NULL,
    bit646                bit             SPARSE  NULL,
    bit647                bit             SPARSE  NULL,
    bit648                bit             SPARSE  NULL,
    bit649                bit             SPARSE  NULL,
    bit650                bit             SPARSE  NULL,
    bit651                bit             SPARSE  NULL,
    bit652                bit             SPARSE  NULL,
    bit653                bit             SPARSE  NULL,
    bit654                bit             SPARSE  NULL,
    bit655                bit             SPARSE  NULL,
    bit656                bit             SPARSE  NULL,
    bit657                bit             SPARSE  NULL,
    bit658                bit             SPARSE  NULL,
    bit659                bit             SPARSE  NULL,
    bit660                bit             SPARSE  NULL,
    bit661                bit             SPARSE  NULL,
    bit662                bit             SPARSE  NULL,
    bit663                bit             SPARSE  NULL,
    bit664                bit             SPARSE  NULL,
    bit665                bit             SPARSE  NULL,
    bit666                bit             SPARSE  NULL,
    bit667                bit             SPARSE  NULL,
    bit668                bit             SPARSE  NULL,
    bit669                bit             SPARSE  NULL,
    bit670                bit             SPARSE  NULL,
    bit671                bit             SPARSE  NULL,
    bit672                bit             SPARSE  NULL,
    bit673                bit             SPARSE  NULL,
    bit674                bit             SPARSE  NULL,
    bit675                bit             SPARSE  NULL,
    bit676                bit             SPARSE  NULL,
    bit677                bit             SPARSE  NULL,
    bit678                bit             SPARSE  NULL,
    bit679                bit             SPARSE  NULL,
    bit680                bit             SPARSE  NULL,
    bit681                bit             SPARSE  NULL,
    bit682                bit             SPARSE  NULL,
    bit683                bit             SPARSE  NULL,
    bit684                bit             SPARSE  NULL,
    bit685                bit             SPARSE  NULL,
    bit686                bit             SPARSE  NULL,
    bit687                bit             SPARSE  NULL,
    bit688                bit             SPARSE  NULL,
    bit689                bit             SPARSE  NULL,
    bit690                bit             SPARSE  NULL,
    bit691                bit             SPARSE  NULL,
    bit692                bit             SPARSE  NULL,
    bit693                bit             SPARSE  NULL,
    bit694                bit             SPARSE  NULL,
    bit695                bit             SPARSE  NULL,
    bit696                bit             SPARSE  NULL,
    bit697                bit             SPARSE  NULL,
    bit698                bit             SPARSE  NULL,
    bit699                bit             SPARSE  NULL,
    bit700                bit             SPARSE  NULL,
    bit701                bit             SPARSE  NULL,
```

```
bit702                bit             SPARSE  NULL,
bit703                bit             SPARSE  NULL,
bit704                bit             SPARSE  NULL,
bit705                bit             SPARSE  NULL,
bit706                bit             SPARSE  NULL,
bit707                bit             SPARSE  NULL,
bit708                bit             SPARSE  NULL,
bit709                bit             SPARSE  NULL,
bit710                bit             SPARSE  NULL,
bit711                bit             SPARSE  NULL,
bit712                bit             SPARSE  NULL,
bit713                bit             SPARSE  NULL,
bit714                bit             SPARSE  NULL,
bit715                bit             SPARSE  NULL,
bit716                bit             SPARSE  NULL,
bit717                bit             SPARSE  NULL,
bit718                bit             SPARSE  NULL,
bit719                bit             SPARSE  NULL,
bit720                bit             SPARSE  NULL,
bit721                bit             SPARSE  NULL,
bit722                bit             SPARSE  NULL,
bit723                bit             SPARSE  NULL,
bit724                bit             SPARSE  NULL,
bit725                bit             SPARSE  NULL,
bit726                bit             SPARSE  NULL,
bit727                bit             SPARSE  NULL,
bit728                bit             SPARSE  NULL,
bit729                bit             SPARSE  NULL,
bit730                bit             SPARSE  NULL,
bit731                bit             SPARSE  NULL,
bit732                bit             SPARSE  NULL,
bit733                bit             SPARSE  NULL,
bit734                bit             SPARSE  NULL,
bit735                bit             SPARSE  NULL,
bit736                bit             SPARSE  NULL,
bit737                bit             SPARSE  NULL,
bit738                bit             SPARSE  NULL,
bit739                bit             SPARSE  NULL,
bit740                bit             SPARSE  NULL,
bit741                bit             SPARSE  NULL,
bit742                bit             SPARSE  NULL,
bit743                bit             SPARSE  NULL,
bit744                bit             SPARSE  NULL,
bit745                bit             SPARSE  NULL,
bit746                bit             SPARSE  NULL,
bit747                bit             SPARSE  NULL,
bit748                bit             SPARSE  NULL,
bit749                bit             SPARSE  NULL,
bit750                bit             SPARSE  NULL,
bit751                bit             SPARSE  NULL,
bit752                bit             SPARSE  NULL,
bit753                bit             SPARSE  NULL,
bit754                bit             SPARSE  NULL,
bit755                bit             SPARSE  NULL,
bit756                bit             SPARSE  NULL,
bit757                bit             SPARSE  NULL,
bit758                bit             SPARSE  NULL,
bit759                bit             SPARSE  NULL,
bit760                bit             SPARSE  NULL,
bit761                bit             SPARSE  NULL,
bit762                bit             SPARSE  NULL,
bit763                bit             SPARSE  NULL,
bit764                bit             SPARSE  NULL,
bit765                bit             SPARSE  NULL,
bit766                bit             SPARSE  NULL,
bit767                bit             SPARSE  NULL,
bit768                bit             SPARSE  NULL,
bit769                bit             SPARSE  NULL,
bit770                bit             SPARSE  NULL,
```

```
    bit771              bit             SPARSE  NULL,
    bit772              bit             SPARSE  NULL,
    bit773              bit             SPARSE  NULL,
    bit774              bit             SPARSE  NULL,
    bit775              bit             SPARSE  NULL,
    bit776              bit             SPARSE  NULL,
    bit777              bit             SPARSE  NULL,
    bit778              bit             SPARSE  NULL,
    bit779              bit             SPARSE  NULL,
    bit780              bit             SPARSE  NULL,
    bit781              bit             SPARSE  NULL,
    bit782              bit             SPARSE  NULL,
    bit783              bit             SPARSE  NULL,
    bit784              bit             SPARSE  NULL,
    bit785              bit             SPARSE  NULL,
    bit786              bit             SPARSE  NULL,
    bit787              bit             SPARSE  NULL,
    bit788              bit             SPARSE  NULL,
    bit789              bit             SPARSE  NULL,
    bit790              bit             SPARSE  NULL,
    bit791              bit             SPARSE  NULL,
    bit792              bit             SPARSE  NULL,
    bit793              bit             SPARSE  NULL,
    bit794              bit             SPARSE  NULL,
    bit795              bit             SPARSE  NULL,
    bit796              bit             SPARSE  NULL,
    bit797              bit             SPARSE  NULL,
    bit798              bit             SPARSE  NULL,
    bit799              bit             SPARSE  NULL,
    bit800              bit             SPARSE  NULL,
    bit801              bit             SPARSE  NULL,
    bit802              bit             SPARSE  NULL,
    bit803              bit             SPARSE  NULL,
    bit804              bit             SPARSE  NULL,
    bit805              bit             SPARSE  NULL,
    bit806              bit             SPARSE  NULL,
    bit807              bit             SPARSE  NULL,
    bit808              bit             SPARSE  NULL,
    bit809              bit             SPARSE  NULL,
    bit810              bit             SPARSE  NULL,
    bit811              bit             SPARSE  NULL,
    bit812              bit             SPARSE  NULL,
    bit813              bit             SPARSE  NULL,
    bit814              bit             SPARSE  NULL,
    bit815              bit             SPARSE  NULL,
    bit816              bit             SPARSE  NULL,
    bit817              bit             SPARSE  NULL,
    bit818              bit             SPARSE  NULL,
    bit819              bit             SPARSE  NULL,
    bit820              bit             SPARSE  NULL,
    bit821              bit             SPARSE  NULL,
    bit822              bit             SPARSE  NULL,
    bit823              bit             SPARSE  NULL,
    bit824              bit             SPARSE  NULL,
    bit825              bit             SPARSE  NULL,
    bit826              bit             SPARSE  NULL,
    bit827              bit             SPARSE  NULL,
    bit828              bit             SPARSE  NULL,
    bit829              bit             SPARSE  NULL,
    bit830              bit             SPARSE  NULL,
    bit831              bit             SPARSE  NULL,
    bit832              bit             SPARSE  NULL,
    bit833              bit             SPARSE  NULL,
    bit834              bit             SPARSE  NULL,
    bit835              bit             SPARSE  NULL,
    bit836              bit             SPARSE  NULL,
    bit837              bit             SPARSE  NULL,
    bit838              bit             SPARSE  NULL,
    bit839              bit             SPARSE  NULL,
```

```
bit840                bit             SPARSE  NULL,
bit841                bit             SPARSE  NULL,
bit842                bit             SPARSE  NULL,
bit843                bit             SPARSE  NULL,
bit844                bit             SPARSE  NULL,
bit845                bit             SPARSE  NULL,
bit846                bit             SPARSE  NULL,
bit847                bit             SPARSE  NULL,
bit848                bit             SPARSE  NULL,
bit849                bit             SPARSE  NULL,
bit850                bit             SPARSE  NULL,
bit851                bit             SPARSE  NULL,
bit852                bit             SPARSE  NULL,
bit853                bit             SPARSE  NULL,
bit854                bit             SPARSE  NULL,
bit855                bit             SPARSE  NULL,
bit856                bit             SPARSE  NULL,
bit857                bit             SPARSE  NULL,
bit858                bit             SPARSE  NULL,
bit859                bit             SPARSE  NULL,
bit860                bit             SPARSE  NULL,
bit861                bit             SPARSE  NULL,
bit862                bit             SPARSE  NULL,
bit863                bit             SPARSE  NULL,
bit864                bit             SPARSE  NULL,
bit865                bit             SPARSE  NULL,
bit866                bit             SPARSE  NULL,
bit867                bit             SPARSE  NULL,
bit868                bit             SPARSE  NULL,
bit869                bit             SPARSE  NULL,
bit870                bit             SPARSE  NULL,
bit871                bit             SPARSE  NULL,
bit872                bit             SPARSE  NULL,
bit873                bit             SPARSE  NULL,
bit874                bit             SPARSE  NULL,
bit875                bit             SPARSE  NULL,
bit876                bit             SPARSE  NULL,
bit877                bit             SPARSE  NULL,
bit878                bit             SPARSE  NULL,
bit879                bit             SPARSE  NULL,
bit880                bit             SPARSE  NULL,
bit881                bit             SPARSE  NULL,
bit882                bit             SPARSE  NULL,
bit883                bit             SPARSE  NULL,
bit884                bit             SPARSE  NULL,
bit885                bit             SPARSE  NULL,
bit886                bit             SPARSE  NULL,
bit887                bit             SPARSE  NULL,
bit888                bit             SPARSE  NULL,
bit889                bit             SPARSE  NULL,
bit890                bit             SPARSE  NULL,
bit891                bit             SPARSE  NULL,
bit892                bit             SPARSE  NULL,
bit893                bit             SPARSE  NULL,
bit894                bit             SPARSE  NULL,
bit895                bit             SPARSE  NULL,
bit896                bit             SPARSE  NULL,
bit897                bit             SPARSE  NULL,
bit898                bit             SPARSE  NULL,
bit899                bit             SPARSE  NULL,
bit900                bit             SPARSE  NULL,
bit901                bit             SPARSE  NULL,
bit902                bit             SPARSE  NULL,
bit903                bit             SPARSE  NULL,
bit904                bit             SPARSE  NULL,
bit905                bit             SPARSE  NULL,
bit906                bit             SPARSE  NULL,
bit907                bit             SPARSE  NULL,
bit908                bit             SPARSE  NULL,
```

```
    bit909                bit             SPARSE  NULL,
    bit910                bit             SPARSE  NULL,
    bit911                bit             SPARSE  NULL,
    bit912                bit             SPARSE  NULL,
    bit913                bit             SPARSE  NULL,
    bit914                bit             SPARSE  NULL,
    bit915                bit             SPARSE  NULL,
    bit916                bit             SPARSE  NULL,
    bit917                bit             SPARSE  NULL,
    bit918                bit             SPARSE  NULL,
    bit919                bit             SPARSE  NULL,
    bit920                bit             SPARSE  NULL,
    bit921                bit             SPARSE  NULL,
    bit922                bit             SPARSE  NULL,
    bit923                bit             SPARSE  NULL,
    bit924                bit             SPARSE  NULL,
    bit925                bit             SPARSE  NULL,
    bit926                bit             SPARSE  NULL,
    bit927                bit             SPARSE  NULL,
    bit928                bit             SPARSE  NULL,
    bit929                bit             SPARSE  NULL,
    bit930                bit             SPARSE  NULL,
    bit931                bit             SPARSE  NULL,
    bit932                bit             SPARSE  NULL,
    bit933                bit             SPARSE  NULL,
    bit934                bit             SPARSE  NULL,
    bit935                bit             SPARSE  NULL,
    bit936                bit             SPARSE  NULL,
    bit937                bit             SPARSE  NULL,
    bit938                bit             SPARSE  NULL,
    bit939                bit             SPARSE  NULL,
    bit940                bit             SPARSE  NULL,
    bit941                bit             SPARSE  NULL,
    bit942                bit             SPARSE  NULL,
    bit943                bit             SPARSE  NULL,
    bit944                bit             SPARSE  NULL,
    bit945                bit             SPARSE  NULL,
    bit946                bit             SPARSE  NULL,
    bit947                bit             SPARSE  NULL,
    bit948                bit             SPARSE  NULL,
    bit949                bit             SPARSE  NULL,
    bit950                bit             SPARSE  NULL,
    bit951                bit             SPARSE  NULL,
    bit952                bit             SPARSE  NULL,
    bit953                bit             SPARSE  NULL,
    bit954                bit             SPARSE  NULL,
    bit955                bit             SPARSE  NULL,
    bit956                bit             SPARSE  NULL,
    bit957                bit             SPARSE  NULL,
    bit958                bit             SPARSE  NULL,
    bit959                bit             SPARSE  NULL,
    bit960                bit             SPARSE  NULL,
    bit961                bit             SPARSE  NULL,
    bit962                bit             SPARSE  NULL,
    bit963                bit             SPARSE  NULL,
    bit964                bit             SPARSE  NULL,
    bit965                bit             SPARSE  NULL,
    bit966                bit             SPARSE  NULL,
    bit967                bit             SPARSE  NULL,
    bit968                bit             SPARSE  NULL,
    bit969                bit             SPARSE  NULL,
    bit970                bit             SPARSE  NULL,
    bit971                bit             SPARSE  NULL,
    bit972                bit             SPARSE  NULL,
    bit973                bit             SPARSE  NULL,
    bit974                bit             SPARSE  NULL,
    bit975                bit             SPARSE  NULL,
    bit976                bit             SPARSE  NULL,
    bit977                bit             SPARSE  NULL,
```

```
bit978                bit             SPARSE  NULL,
bit979                bit             SPARSE  NULL,
bit980                bit             SPARSE  NULL,
bit981                bit             SPARSE  NULL,
bit982                bit             SPARSE  NULL,
bit983                bit             SPARSE  NULL,
bit984                bit             SPARSE  NULL,
bit985                bit             SPARSE  NULL,
bit986                bit             SPARSE  NULL,
bit987                bit             SPARSE  NULL,
bit988                bit             SPARSE  NULL,
bit989                bit             SPARSE  NULL,
bit990                bit             SPARSE  NULL,
bit991                bit             SPARSE  NULL,
bit992                bit             SPARSE  NULL,
bit993                bit             SPARSE  NULL,
bit994                bit             SPARSE  NULL,
bit995                bit             SPARSE  NULL,
bit996                bit             SPARSE  NULL,
bit997                bit             SPARSE  NULL,
bit998                bit             SPARSE  NULL,
bit999                bit             SPARSE  NULL,
bit1000               bit             SPARSE  NULL,
datetime1             datetime        SPARSE  NULL,
datetime2             datetime        SPARSE  NULL,
datetime3             datetime        SPARSE  NULL,
datetime4             datetime        SPARSE  NULL,
datetime5             datetime        SPARSE  NULL,
datetime6             datetime        SPARSE  NULL,
datetime7             datetime        SPARSE  NULL,
datetime8             datetime        SPARSE  NULL,
datetime9             datetime        SPARSE  NULL,
datetime10            datetime        SPARSE  NULL,
datetime11            datetime        SPARSE  NULL,
datetime12            datetime        SPARSE  NULL,
datetime13            datetime        SPARSE  NULL,
datetime14            datetime        SPARSE  NULL,
datetime15            datetime        SPARSE  NULL,
datetime16            datetime        SPARSE  NULL,
datetime17            datetime        SPARSE  NULL,
datetime18            datetime        SPARSE  NULL,
datetime19            datetime        SPARSE  NULL,
datetime20            datetime        SPARSE  NULL,
datetime21            datetime        SPARSE  NULL,
datetime22            datetime        SPARSE  NULL,
datetime23            datetime        SPARSE  NULL,
datetime24            datetime        SPARSE  NULL,
datetime25            datetime        SPARSE  NULL,
datetime26            datetime        SPARSE  NULL,
datetime27            datetime        SPARSE  NULL,
datetime28            datetime        SPARSE  NULL,
datetime29            datetime        SPARSE  NULL,
datetime30            datetime        SPARSE  NULL,
datetime31            datetime        SPARSE  NULL,
datetime32            datetime        SPARSE  NULL,
datetime33            datetime        SPARSE  NULL,
datetime34            datetime        SPARSE  NULL,
datetime35            datetime        SPARSE  NULL,
datetime36            datetime        SPARSE  NULL,
datetime37            datetime        SPARSE  NULL,
datetime38            datetime        SPARSE  NULL,
datetime39            datetime        SPARSE  NULL,
datetime40            datetime        SPARSE  NULL,
datetime41            datetime        SPARSE  NULL,
datetime42            datetime        SPARSE  NULL,
datetime43            datetime        SPARSE  NULL,
datetime44            datetime        SPARSE  NULL,
datetime45            datetime        SPARSE  NULL,
datetime46            datetime        SPARSE  NULL,
```

```
datetime47              datetime          SPARSE  NULL,
datetime48              datetime          SPARSE  NULL,
datetime49              datetime          SPARSE  NULL,
datetime50              datetime          SPARSE  NULL,
datetime51              datetime          SPARSE  NULL,
datetime52              datetime          SPARSE  NULL,
datetime53              datetime          SPARSE  NULL,
datetime54              datetime          SPARSE  NULL,
datetime55              datetime          SPARSE  NULL,
datetime56              datetime          SPARSE  NULL,
datetime57              datetime          SPARSE  NULL,
datetime58              datetime          SPARSE  NULL,
datetime59              datetime          SPARSE  NULL,
datetime60              datetime          SPARSE  NULL,
datetime61              datetime          SPARSE  NULL,
datetime62              datetime          SPARSE  NULL,
datetime63              datetime          SPARSE  NULL,
datetime64              datetime          SPARSE  NULL,
datetime65              datetime          SPARSE  NULL,
datetime66              datetime          SPARSE  NULL,
datetime67              datetime          SPARSE  NULL,
datetime68              datetime          SPARSE  NULL,
datetime69              datetime          SPARSE  NULL,
datetime70              datetime          SPARSE  NULL,
datetime71              datetime          SPARSE  NULL,
datetime72              datetime          SPARSE  NULL,
datetime73              datetime          SPARSE  NULL,
datetime74              datetime          SPARSE  NULL,
datetime75              datetime          SPARSE  NULL,
datetime76              datetime          SPARSE  NULL,
datetime77              datetime          SPARSE  NULL,
datetime78              datetime          SPARSE  NULL,
datetime79              datetime          SPARSE  NULL,
datetime80              datetime          SPARSE  NULL,
datetime81              datetime          SPARSE  NULL,
datetime82              datetime          SPARSE  NULL,
datetime83              datetime          SPARSE  NULL,
datetime84              datetime          SPARSE  NULL,
datetime85              datetime          SPARSE  NULL,
datetime86              datetime          SPARSE  NULL,
datetime87              datetime          SPARSE  NULL,
datetime88              datetime          SPARSE  NULL,
datetime89              datetime          SPARSE  NULL,
datetime90              datetime          SPARSE  NULL,
datetime91              datetime          SPARSE  NULL,
datetime92              datetime          SPARSE  NULL,
datetime93              datetime          SPARSE  NULL,
datetime94              datetime          SPARSE  NULL,
datetime95              datetime          SPARSE  NULL,
datetime96              datetime          SPARSE  NULL,
datetime97              datetime          SPARSE  NULL,
datetime98              datetime          SPARSE  NULL,
datetime99              datetime          SPARSE  NULL,
datetime100             datetime          SPARSE  NULL,
datetime101             datetime          SPARSE  NULL,
datetime102             datetime          SPARSE  NULL,
datetime103             datetime          SPARSE  NULL,
datetime104             datetime          SPARSE  NULL,
datetime105             datetime          SPARSE  NULL,
datetime106             datetime          SPARSE  NULL,
datetime107             datetime          SPARSE  NULL,
datetime108             datetime          SPARSE  NULL,
datetime109             datetime          SPARSE  NULL,
datetime110             datetime          SPARSE  NULL,
datetime111             datetime          SPARSE  NULL,
datetime112             datetime          SPARSE  NULL,
datetime113             datetime          SPARSE  NULL,
datetime114             datetime          SPARSE  NULL,
datetime115             datetime          SPARSE  NULL,
```

```
    datetime116             datetime          SPARSE  NULL,
    datetime117             datetime          SPARSE  NULL,
    datetime118             datetime          SPARSE  NULL,
    datetime119             datetime          SPARSE  NULL,
    datetime120             datetime          SPARSE  NULL,
    datetime121             datetime          SPARSE  NULL,
    datetime122             datetime          SPARSE  NULL,
    datetime123             datetime          SPARSE  NULL,
    datetime124             datetime          SPARSE  NULL,
    datetime125             datetime          SPARSE  NULL,
    datetime126             datetime          SPARSE  NULL,
    datetime127             datetime          SPARSE  NULL,
    datetime128             datetime          SPARSE  NULL,
    datetime129             datetime          SPARSE  NULL,
    datetime130             datetime          SPARSE  NULL,
    datetime131             datetime          SPARSE  NULL,
    datetime132             datetime          SPARSE  NULL,
    datetime133             datetime          SPARSE  NULL,
    datetime134             datetime          SPARSE  NULL,
    datetime135             datetime          SPARSE  NULL,
    datetime136             datetime          SPARSE  NULL,
    datetime137             datetime          SPARSE  NULL,
    datetime138             datetime          SPARSE  NULL,
    datetime139             datetime          SPARSE  NULL,
    datetime140             datetime          SPARSE  NULL,
    datetime141             datetime          SPARSE  NULL,
    datetime142             datetime          SPARSE  NULL,
    datetime143             datetime          SPARSE  NULL,
    datetime144             datetime          SPARSE  NULL,
    datetime145             datetime          SPARSE  NULL,
    datetime146             datetime          SPARSE  NULL,
    datetime147             datetime          SPARSE  NULL,
    datetime148             datetime          SPARSE  NULL,
    datetime149             datetime          SPARSE  NULL,
    datetime150             datetime          SPARSE  NULL,
    datetime151             datetime          SPARSE  NULL,
    datetime152             datetime          SPARSE  NULL,
    datetime153             datetime          SPARSE  NULL,
    datetime154             datetime          SPARSE  NULL,
    datetime155             datetime          SPARSE  NULL,
    datetime156             datetime          SPARSE  NULL,
    datetime157             datetime          SPARSE  NULL,
    datetime158             datetime          SPARSE  NULL,
    datetime159             datetime          SPARSE  NULL,
    datetime160             datetime          SPARSE  NULL,
    datetime161             datetime          SPARSE  NULL,
    datetime162             datetime          SPARSE  NULL,
    datetime163             datetime          SPARSE  NULL,
    datetime164             datetime          SPARSE  NULL,
    datetime165             datetime          SPARSE  NULL,
    datetime166             datetime          SPARSE  NULL,
    datetime167             datetime          SPARSE  NULL,
    datetime168             datetime          SPARSE  NULL,
    datetime169             datetime          SPARSE  NULL,
    datetime170             datetime          SPARSE  NULL,
    datetime171             datetime          SPARSE  NULL,
    datetime172             datetime          SPARSE  NULL,
    datetime173             datetime          SPARSE  NULL,
    datetime174             datetime          SPARSE  NULL,
    datetime175             datetime          SPARSE  NULL,
    datetime176             datetime          SPARSE  NULL,
    datetime177             datetime          SPARSE  NULL,
    datetime178             datetime          SPARSE  NULL,
    datetime179             datetime          SPARSE  NULL,
    datetime180             datetime          SPARSE  NULL,
    datetime181             datetime          SPARSE  NULL,
    datetime182             datetime          SPARSE  NULL,
    datetime183             datetime          SPARSE  NULL,
    datetime184             datetime          SPARSE  NULL,
```

```
    datetime185             datetime          SPARSE  NULL,
    datetime186             datetime          SPARSE  NULL,
    datetime187             datetime          SPARSE  NULL,
    datetime188             datetime          SPARSE  NULL,
    datetime189             datetime          SPARSE  NULL,
    datetime190             datetime          SPARSE  NULL,
    datetime191             datetime          SPARSE  NULL,
    datetime192             datetime          SPARSE  NULL,
    datetime193             datetime          SPARSE  NULL,
    datetime194             datetime          SPARSE  NULL,
    datetime195             datetime          SPARSE  NULL,
    datetime196             datetime          SPARSE  NULL,
    datetime197             datetime          SPARSE  NULL,
    datetime198             datetime          SPARSE  NULL,
    datetime199             datetime          SPARSE  NULL,
    datetime200             datetime          SPARSE  NULL,
    datetime201             datetime          SPARSE  NULL,
    datetime202             datetime          SPARSE  NULL,
    datetime203             datetime          SPARSE  NULL,
    datetime204             datetime          SPARSE  NULL,
    datetime205             datetime          SPARSE  NULL,
    datetime206             datetime          SPARSE  NULL,
    datetime207             datetime          SPARSE  NULL,
    datetime208             datetime          SPARSE  NULL,
    datetime209             datetime          SPARSE  NULL,
    datetime210             datetime          SPARSE  NULL,
    datetime211             datetime          SPARSE  NULL,
    datetime212             datetime          SPARSE  NULL,
    datetime213             datetime          SPARSE  NULL,
    datetime214             datetime          SPARSE  NULL,
    datetime215             datetime          SPARSE  NULL,
    datetime216             datetime          SPARSE  NULL,
    datetime217             datetime          SPARSE  NULL,
    datetime218             datetime          SPARSE  NULL,
    datetime219             datetime          SPARSE  NULL,
    datetime220             datetime          SPARSE  NULL,
    datetime221             datetime          SPARSE  NULL,
    datetime222             datetime          SPARSE  NULL,
    datetime223             datetime          SPARSE  NULL,
    datetime224             datetime          SPARSE  NULL,
    datetime225             datetime          SPARSE  NULL,
    datetime226             datetime          SPARSE  NULL,
    datetime227             datetime          SPARSE  NULL,
    datetime228             datetime          SPARSE  NULL,
    datetime229             datetime          SPARSE  NULL,
    datetime230             datetime          SPARSE  NULL,
    datetime231             datetime          SPARSE  NULL,
    datetime232             datetime          SPARSE  NULL,
    datetime233             datetime          SPARSE  NULL,
    datetime234             datetime          SPARSE  NULL,
    datetime235             datetime          SPARSE  NULL,
    datetime236             datetime          SPARSE  NULL,
    datetime237             datetime          SPARSE  NULL,
    datetime238             datetime          SPARSE  NULL,
    datetime239             datetime          SPARSE  NULL,
    datetime240             datetime          SPARSE  NULL,
    datetime241             datetime          SPARSE  NULL,
    datetime242             datetime          SPARSE  NULL,
    datetime243             datetime          SPARSE  NULL,
    datetime244             datetime          SPARSE  NULL,
    datetime245             datetime          SPARSE  NULL,
    datetime246             datetime          SPARSE  NULL,
    datetime247             datetime          SPARSE  NULL,
    datetime248             datetime          SPARSE  NULL,
    datetime249             datetime          SPARSE  NULL,
    datetime250             datetime          SPARSE  NULL,
    datetime251             datetime          SPARSE  NULL,
    datetime252             datetime          SPARSE  NULL,
    datetime253             datetime          SPARSE  NULL,
```

```
datetime254             datetime          SPARSE  NULL,
datetime255             datetime          SPARSE  NULL,
datetime256             datetime          SPARSE  NULL,
datetime257             datetime          SPARSE  NULL,
datetime258             datetime          SPARSE  NULL,
datetime259             datetime          SPARSE  NULL,
datetime260             datetime          SPARSE  NULL,
datetime261             datetime          SPARSE  NULL,
datetime262             datetime          SPARSE  NULL,
datetime263             datetime          SPARSE  NULL,
datetime264             datetime          SPARSE  NULL,
datetime265             datetime          SPARSE  NULL,
datetime266             datetime          SPARSE  NULL,
datetime267             datetime          SPARSE  NULL,
datetime268             datetime          SPARSE  NULL,
datetime269             datetime          SPARSE  NULL,
datetime270             datetime          SPARSE  NULL,
datetime271             datetime          SPARSE  NULL,
datetime272             datetime          SPARSE  NULL,
datetime273             datetime          SPARSE  NULL,
datetime274             datetime          SPARSE  NULL,
datetime275             datetime          SPARSE  NULL,
datetime276             datetime          SPARSE  NULL,
datetime277             datetime          SPARSE  NULL,
datetime278             datetime          SPARSE  NULL,
datetime279             datetime          SPARSE  NULL,
datetime280             datetime          SPARSE  NULL,
datetime281             datetime          SPARSE  NULL,
datetime282             datetime          SPARSE  NULL,
datetime283             datetime          SPARSE  NULL,
datetime284             datetime          SPARSE  NULL,
datetime285             datetime          SPARSE  NULL,
datetime286             datetime          SPARSE  NULL,
datetime287             datetime          SPARSE  NULL,
datetime288             datetime          SPARSE  NULL,
datetime289             datetime          SPARSE  NULL,
datetime290             datetime          SPARSE  NULL,
datetime291             datetime          SPARSE  NULL,
datetime292             datetime          SPARSE  NULL,
datetime293             datetime          SPARSE  NULL,
datetime294             datetime          SPARSE  NULL,
datetime295             datetime          SPARSE  NULL,
datetime296             datetime          SPARSE  NULL,
datetime297             datetime          SPARSE  NULL,
datetime298             datetime          SPARSE  NULL,
datetime299             datetime          SPARSE  NULL,
datetime300             datetime          SPARSE  NULL,
datetime301             datetime          SPARSE  NULL,
datetime302             datetime          SPARSE  NULL,
datetime303             datetime          SPARSE  NULL,
datetime304             datetime          SPARSE  NULL,
datetime305             datetime          SPARSE  NULL,
datetime306             datetime          SPARSE  NULL,
datetime307             datetime          SPARSE  NULL,
datetime308             datetime          SPARSE  NULL,
datetime309             datetime          SPARSE  NULL,
datetime310             datetime          SPARSE  NULL,
datetime311             datetime          SPARSE  NULL,
datetime312             datetime          SPARSE  NULL,
datetime313             datetime          SPARSE  NULL,
datetime314             datetime          SPARSE  NULL,
datetime315             datetime          SPARSE  NULL,
datetime316             datetime          SPARSE  NULL,
datetime317             datetime          SPARSE  NULL,
datetime318             datetime          SPARSE  NULL,
datetime319             datetime          SPARSE  NULL,
datetime320             datetime          SPARSE  NULL,
datetime321             datetime          SPARSE  NULL,
datetime322             datetime          SPARSE  NULL,
```

```
datetime323             datetime          SPARSE  NULL,
datetime324             datetime          SPARSE  NULL,
datetime325             datetime          SPARSE  NULL,
datetime326             datetime          SPARSE  NULL,
datetime327             datetime          SPARSE  NULL,
datetime328             datetime          SPARSE  NULL,
datetime329             datetime          SPARSE  NULL,
datetime330             datetime          SPARSE  NULL,
datetime331             datetime          SPARSE  NULL,
datetime332             datetime          SPARSE  NULL,
datetime333             datetime          SPARSE  NULL,
datetime334             datetime          SPARSE  NULL,
datetime335             datetime          SPARSE  NULL,
datetime336             datetime          SPARSE  NULL,
datetime337             datetime          SPARSE  NULL,
datetime338             datetime          SPARSE  NULL,
datetime339             datetime          SPARSE  NULL,
datetime340             datetime          SPARSE  NULL,
datetime341             datetime          SPARSE  NULL,
datetime342             datetime          SPARSE  NULL,
datetime343             datetime          SPARSE  NULL,
datetime344             datetime          SPARSE  NULL,
datetime345             datetime          SPARSE  NULL,
datetime346             datetime          SPARSE  NULL,
datetime347             datetime          SPARSE  NULL,
datetime348             datetime          SPARSE  NULL,
datetime349             datetime          SPARSE  NULL,
datetime350             datetime          SPARSE  NULL,
datetime351             datetime          SPARSE  NULL,
datetime352             datetime          SPARSE  NULL,
datetime353             datetime          SPARSE  NULL,
datetime354             datetime          SPARSE  NULL,
datetime355             datetime          SPARSE  NULL,
datetime356             datetime          SPARSE  NULL,
datetime357             datetime          SPARSE  NULL,
datetime358             datetime          SPARSE  NULL,
datetime359             datetime          SPARSE  NULL,
datetime360             datetime          SPARSE  NULL,
datetime361             datetime          SPARSE  NULL,
datetime362             datetime          SPARSE  NULL,
datetime363             datetime          SPARSE  NULL,
datetime364             datetime          SPARSE  NULL,
datetime365             datetime          SPARSE  NULL,
datetime366             datetime          SPARSE  NULL,
datetime367             datetime          SPARSE  NULL,
datetime368             datetime          SPARSE  NULL,
datetime369             datetime          SPARSE  NULL,
datetime370             datetime          SPARSE  NULL,
datetime371             datetime          SPARSE  NULL,
datetime372             datetime          SPARSE  NULL,
datetime373             datetime          SPARSE  NULL,
datetime374             datetime          SPARSE  NULL,
datetime375             datetime          SPARSE  NULL,
datetime376             datetime          SPARSE  NULL,
datetime377             datetime          SPARSE  NULL,
datetime378             datetime          SPARSE  NULL,
datetime379             datetime          SPARSE  NULL,
datetime380             datetime          SPARSE  NULL,
datetime381             datetime          SPARSE  NULL,
datetime382             datetime          SPARSE  NULL,
datetime383             datetime          SPARSE  NULL,
datetime384             datetime          SPARSE  NULL,
datetime385             datetime          SPARSE  NULL,
datetime386             datetime          SPARSE  NULL,
datetime387             datetime          SPARSE  NULL,
datetime388             datetime          SPARSE  NULL,
datetime389             datetime          SPARSE  NULL,
datetime390             datetime          SPARSE  NULL,
datetime391             datetime          SPARSE  NULL,
```

```
datetime392             datetime          SPARSE  NULL,
datetime393             datetime          SPARSE  NULL,
datetime394             datetime          SPARSE  NULL,
datetime395             datetime          SPARSE  NULL,
datetime396             datetime          SPARSE  NULL,
datetime397             datetime          SPARSE  NULL,
datetime398             datetime          SPARSE  NULL,
datetime399             datetime          SPARSE  NULL,
datetime400             datetime          SPARSE  NULL,
datetime401             datetime          SPARSE  NULL,
datetime402             datetime          SPARSE  NULL,
datetime403             datetime          SPARSE  NULL,
datetime404             datetime          SPARSE  NULL,
datetime405             datetime          SPARSE  NULL,
datetime406             datetime          SPARSE  NULL,
datetime407             datetime          SPARSE  NULL,
datetime408             datetime          SPARSE  NULL,
datetime409             datetime          SPARSE  NULL,
datetime410             datetime          SPARSE  NULL,
datetime411             datetime          SPARSE  NULL,
datetime412             datetime          SPARSE  NULL,
datetime413             datetime          SPARSE  NULL,
datetime414             datetime          SPARSE  NULL,
datetime415             datetime          SPARSE  NULL,
datetime416             datetime          SPARSE  NULL,
datetime417             datetime          SPARSE  NULL,
datetime418             datetime          SPARSE  NULL,
datetime419             datetime          SPARSE  NULL,
datetime420             datetime          SPARSE  NULL,
datetime421             datetime          SPARSE  NULL,
datetime422             datetime          SPARSE  NULL,
datetime423             datetime          SPARSE  NULL,
datetime424             datetime          SPARSE  NULL,
datetime425             datetime          SPARSE  NULL,
datetime426             datetime          SPARSE  NULL,
datetime427             datetime          SPARSE  NULL,
datetime428             datetime          SPARSE  NULL,
datetime429             datetime          SPARSE  NULL,
datetime430             datetime          SPARSE  NULL,
datetime431             datetime          SPARSE  NULL,
datetime432             datetime          SPARSE  NULL,
datetime433             datetime          SPARSE  NULL,
datetime434             datetime          SPARSE  NULL,
datetime435             datetime          SPARSE  NULL,
datetime436             datetime          SPARSE  NULL,
datetime437             datetime          SPARSE  NULL,
datetime438             datetime          SPARSE  NULL,
datetime439             datetime          SPARSE  NULL,
datetime440             datetime          SPARSE  NULL,
datetime441             datetime          SPARSE  NULL,
datetime442             datetime          SPARSE  NULL,
datetime443             datetime          SPARSE  NULL,
datetime444             datetime          SPARSE  NULL,
datetime445             datetime          SPARSE  NULL,
datetime446             datetime          SPARSE  NULL,
datetime447             datetime          SPARSE  NULL,
datetime448             datetime          SPARSE  NULL,
datetime449             datetime          SPARSE  NULL,
datetime450             datetime          SPARSE  NULL,
datetime451             datetime          SPARSE  NULL,
datetime452             datetime          SPARSE  NULL,
datetime453             datetime          SPARSE  NULL,
datetime454             datetime          SPARSE  NULL,
datetime455             datetime          SPARSE  NULL,
datetime456             datetime          SPARSE  NULL,
datetime457             datetime          SPARSE  NULL,
datetime458             datetime          SPARSE  NULL,
datetime459             datetime          SPARSE  NULL,
datetime460             datetime          SPARSE  NULL,
```

```
    datetime461             datetime          SPARSE  NULL,
    datetime462             datetime          SPARSE  NULL,
    datetime463             datetime          SPARSE  NULL,
    datetime464             datetime          SPARSE  NULL,
    datetime465             datetime          SPARSE  NULL,
    datetime466             datetime          SPARSE  NULL,
    datetime467             datetime          SPARSE  NULL,
    datetime468             datetime          SPARSE  NULL,
    datetime469             datetime          SPARSE  NULL,
    datetime470             datetime          SPARSE  NULL,
    datetime471             datetime          SPARSE  NULL,
    datetime472             datetime          SPARSE  NULL,
    datetime473             datetime          SPARSE  NULL,
    datetime474             datetime          SPARSE  NULL,
    datetime475             datetime          SPARSE  NULL,
    datetime476             datetime          SPARSE  NULL,
    datetime477             datetime          SPARSE  NULL,
    datetime478             datetime          SPARSE  NULL,
    datetime479             datetime          SPARSE  NULL,
    datetime480             datetime          SPARSE  NULL,
    datetime481             datetime          SPARSE  NULL,
    datetime482             datetime          SPARSE  NULL,
    datetime483             datetime          SPARSE  NULL,
    datetime484             datetime          SPARSE  NULL,
    datetime485             datetime          SPARSE  NULL,
    datetime486             datetime          SPARSE  NULL,
    datetime487             datetime          SPARSE  NULL,
    datetime488             datetime          SPARSE  NULL,
    datetime489             datetime          SPARSE  NULL,
    datetime490             datetime          SPARSE  NULL,
    datetime491             datetime          SPARSE  NULL,
    datetime492             datetime          SPARSE  NULL,
    datetime493             datetime          SPARSE  NULL,
    datetime494             datetime          SPARSE  NULL,
    datetime495             datetime          SPARSE  NULL,
    datetime496             datetime          SPARSE  NULL,
    datetime497             datetime          SPARSE  NULL,
    datetime498             datetime          SPARSE  NULL,
    datetime499             datetime          SPARSE  NULL,
    datetime500             datetime          SPARSE  NULL,
    datetime501             datetime          SPARSE  NULL,
    datetime502             datetime          SPARSE  NULL,
    datetime503             datetime          SPARSE  NULL,
    datetime504             datetime          SPARSE  NULL,
    datetime505             datetime          SPARSE  NULL,
    datetime506             datetime          SPARSE  NULL,
    datetime507             datetime          SPARSE  NULL,
    datetime508             datetime          SPARSE  NULL,
    datetime509             datetime          SPARSE  NULL,
    datetime510             datetime          SPARSE  NULL,
    datetime511             datetime          SPARSE  NULL,
    datetime512             datetime          SPARSE  NULL,
    datetime513             datetime          SPARSE  NULL,
    datetime514             datetime          SPARSE  NULL,
    datetime515             datetime          SPARSE  NULL,
    datetime516             datetime          SPARSE  NULL,
    datetime517             datetime          SPARSE  NULL,
    datetime518             datetime          SPARSE  NULL,
    datetime519             datetime          SPARSE  NULL,
    datetime520             datetime          SPARSE  NULL,
    datetime521             datetime          SPARSE  NULL,
    datetime522             datetime          SPARSE  NULL,
    datetime523             datetime          SPARSE  NULL,
    datetime524             datetime          SPARSE  NULL,
    datetime525             datetime          SPARSE  NULL,
    datetime526             datetime          SPARSE  NULL,
    datetime527             datetime          SPARSE  NULL,
    datetime528             datetime          SPARSE  NULL,
    datetime529             datetime          SPARSE  NULL,
```

```
datetime530             datetime          SPARSE  NULL,
datetime531             datetime          SPARSE  NULL,
datetime532             datetime          SPARSE  NULL,
datetime533             datetime          SPARSE  NULL,
datetime534             datetime          SPARSE  NULL,
datetime535             datetime          SPARSE  NULL,
datetime536             datetime          SPARSE  NULL,
datetime537             datetime          SPARSE  NULL,
datetime538             datetime          SPARSE  NULL,
datetime539             datetime          SPARSE  NULL,
datetime540             datetime          SPARSE  NULL,
datetime541             datetime          SPARSE  NULL,
datetime542             datetime          SPARSE  NULL,
datetime543             datetime          SPARSE  NULL,
datetime544             datetime          SPARSE  NULL,
datetime545             datetime          SPARSE  NULL,
datetime546             datetime          SPARSE  NULL,
datetime547             datetime          SPARSE  NULL,
datetime548             datetime          SPARSE  NULL,
datetime549             datetime          SPARSE  NULL,
datetime550             datetime          SPARSE  NULL,
float1                  float             SPARSE  NULL,
float2                  float             SPARSE  NULL,
float3                  float             SPARSE  NULL,
float4                  float             SPARSE  NULL,
float5                  float             SPARSE  NULL,
float6                  float             SPARSE  NULL,
float7                  float             SPARSE  NULL,
float8                  float             SPARSE  NULL,
float9                  float             SPARSE  NULL,
float10                 float             SPARSE  NULL,
float11                 float             SPARSE  NULL,
float12                 float             SPARSE  NULL,
float13                 float             SPARSE  NULL,
float14                 float             SPARSE  NULL,
float15                 float             SPARSE  NULL,
float16                 float             SPARSE  NULL,
float17                 float             SPARSE  NULL,
float18                 float             SPARSE  NULL,
float19                 float             SPARSE  NULL,
float20                 float             SPARSE  NULL,
float21                 float             SPARSE  NULL,
float22                 float             SPARSE  NULL,
float23                 float             SPARSE  NULL,
float24                 float             SPARSE  NULL,
float25                 float             SPARSE  NULL,
float26                 float             SPARSE  NULL,
float27                 float             SPARSE  NULL,
float28                 float             SPARSE  NULL,
float29                 float             SPARSE  NULL,
float30                 float             SPARSE  NULL,
float31                 float             SPARSE  NULL,
float32                 float             SPARSE  NULL,
float33                 float             SPARSE  NULL,
float34                 float             SPARSE  NULL,
float35                 float             SPARSE  NULL,
float36                 float             SPARSE  NULL,
float37                 float             SPARSE  NULL,
float38                 float             SPARSE  NULL,
float39                 float             SPARSE  NULL,
float40                 float             SPARSE  NULL,
float41                 float             SPARSE  NULL,
float42                 float             SPARSE  NULL,
float43                 float             SPARSE  NULL,
float44                 float             SPARSE  NULL,
float45                 float             SPARSE  NULL,
float46                 float             SPARSE  NULL,
float47                 float             SPARSE  NULL,
float48                 float             SPARSE  NULL,
```

```
float49                 float             SPARSE  NULL,
float50                 float             SPARSE  NULL,
float51                 float             SPARSE  NULL,
float52                 float             SPARSE  NULL,
float53                 float             SPARSE  NULL,
float54                 float             SPARSE  NULL,
float55                 float             SPARSE  NULL,
float56                 float             SPARSE  NULL,
float57                 float             SPARSE  NULL,
float58                 float             SPARSE  NULL,
float59                 float             SPARSE  NULL,
float60                 float             SPARSE  NULL,
float61                 float             SPARSE  NULL,
float62                 float             SPARSE  NULL,
float63                 float             SPARSE  NULL,
float64                 float             SPARSE  NULL,
float65                 float             SPARSE  NULL,
float66                 float             SPARSE  NULL,
float67                 float             SPARSE  NULL,
float68                 float             SPARSE  NULL,
float69                 float             SPARSE  NULL,
float70                 float             SPARSE  NULL,
float71                 float             SPARSE  NULL,
float72                 float             SPARSE  NULL,
float73                 float             SPARSE  NULL,
float74                 float             SPARSE  NULL,
float75                 float             SPARSE  NULL,
float76                 float             SPARSE  NULL,
float77                 float             SPARSE  NULL,
float78                 float             SPARSE  NULL,
float79                 float             SPARSE  NULL,
float80                 float             SPARSE  NULL,
float81                 float             SPARSE  NULL,
float82                 float             SPARSE  NULL,
float83                 float             SPARSE  NULL,
float84                 float             SPARSE  NULL,
float85                 float             SPARSE  NULL,
float86                 float             SPARSE  NULL,
float87                 float             SPARSE  NULL,
float88                 float             SPARSE  NULL,
float89                 float             SPARSE  NULL,
float90                 float             SPARSE  NULL,
float91                 float             SPARSE  NULL,
float92                 float             SPARSE  NULL,
float93                 float             SPARSE  NULL,
float94                 float             SPARSE  NULL,
float95                 float             SPARSE  NULL,
float96                 float             SPARSE  NULL,
float97                 float             SPARSE  NULL,
float98                 float             SPARSE  NULL,
float99                 float             SPARSE  NULL,
float100                float             SPARSE  NULL,
float101                float             SPARSE  NULL,
float102                float             SPARSE  NULL,
float103                float             SPARSE  NULL,
float104                float             SPARSE  NULL,
float105                float             SPARSE  NULL,
float106                float             SPARSE  NULL,
float107                float             SPARSE  NULL,
float108                float             SPARSE  NULL,
float109                float             SPARSE  NULL,
float110                float             SPARSE  NULL,
float111                float             SPARSE  NULL,
float112                float             SPARSE  NULL,
float113                float             SPARSE  NULL,
float114                float             SPARSE  NULL,
float115                float             SPARSE  NULL,
float116                float             SPARSE  NULL,
float117                float             SPARSE  NULL,
```

```
float118                float             SPARSE  NULL,
float119                float             SPARSE  NULL,
float120                float             SPARSE  NULL,
float121                float             SPARSE  NULL,
float122                float             SPARSE  NULL,
float123                float             SPARSE  NULL,
float124                float             SPARSE  NULL,
float125                float             SPARSE  NULL,
float126                float             SPARSE  NULL,
float127                float             SPARSE  NULL,
float128                float             SPARSE  NULL,
float129                float             SPARSE  NULL,
float130                float             SPARSE  NULL,
float131                float             SPARSE  NULL,
float132                float             SPARSE  NULL,
float133                float             SPARSE  NULL,
float134                float             SPARSE  NULL,
float135                float             SPARSE  NULL,
float136                float             SPARSE  NULL,
float137                float             SPARSE  NULL,
float138                float             SPARSE  NULL,
float139                float             SPARSE  NULL,
float140                float             SPARSE  NULL,
float141                float             SPARSE  NULL,
float142                float             SPARSE  NULL,
float143                float             SPARSE  NULL,
float144                float             SPARSE  NULL,
float145                float             SPARSE  NULL,
float146                float             SPARSE  NULL,
float147                float             SPARSE  NULL,
float148                float             SPARSE  NULL,
float149                float             SPARSE  NULL,
float150                float             SPARSE  NULL,
float151                float             SPARSE  NULL,
float152                float             SPARSE  NULL,
float153                float             SPARSE  NULL,
float154                float             SPARSE  NULL,
float155                float             SPARSE  NULL,
float156                float             SPARSE  NULL,
float157                float             SPARSE  NULL,
float158                float             SPARSE  NULL,
float159                float             SPARSE  NULL,
float160                float             SPARSE  NULL,
float161                float             SPARSE  NULL,
float162                float             SPARSE  NULL,
float163                float             SPARSE  NULL,
float164                float             SPARSE  NULL,
float165                float             SPARSE  NULL,
float166                float             SPARSE  NULL,
float167                float             SPARSE  NULL,
float168                float             SPARSE  NULL,
float169                float             SPARSE  NULL,
float170                float             SPARSE  NULL,
float171                float             SPARSE  NULL,
float172                float             SPARSE  NULL,
float173                float             SPARSE  NULL,
float174                float             SPARSE  NULL,
float175                float             SPARSE  NULL,
float176                float             SPARSE  NULL,
float177                float             SPARSE  NULL,
float178                float             SPARSE  NULL,
float179                float             SPARSE  NULL,
float180                float             SPARSE  NULL,
float181                float             SPARSE  NULL,
float182                float             SPARSE  NULL,
float183                float             SPARSE  NULL,
float184                float             SPARSE  NULL,
float185                float             SPARSE  NULL,
float186                float             SPARSE  NULL,
```

```
float187                float             SPARSE  NULL,
float188                float             SPARSE  NULL,
float189                float             SPARSE  NULL,
float190                float             SPARSE  NULL,
float191                float             SPARSE  NULL,
float192                float             SPARSE  NULL,
float193                float             SPARSE  NULL,
float194                float             SPARSE  NULL,
float195                float             SPARSE  NULL,
float196                float             SPARSE  NULL,
float197                float             SPARSE  NULL,
float198                float             SPARSE  NULL,
float199                float             SPARSE  NULL,
float200                float             SPARSE  NULL,
float201                float             SPARSE  NULL,
float202                float             SPARSE  NULL,
float203                float             SPARSE  NULL,
float204                float             SPARSE  NULL,
float205                float             SPARSE  NULL,
float206                float             SPARSE  NULL,
float207                float             SPARSE  NULL,
float208                float             SPARSE  NULL,
float209                float             SPARSE  NULL,
float210                float             SPARSE  NULL,
float211                float             SPARSE  NULL,
float212                float             SPARSE  NULL,
float213                float             SPARSE  NULL,
float214                float             SPARSE  NULL,
float215                float             SPARSE  NULL,
float216                float             SPARSE  NULL,
float217                float             SPARSE  NULL,
float218                float             SPARSE  NULL,
float219                float             SPARSE  NULL,
float220                float             SPARSE  NULL,
float221                float             SPARSE  NULL,
float222                float             SPARSE  NULL,
float223                float             SPARSE  NULL,
float224                float             SPARSE  NULL,
float225                float             SPARSE  NULL,
float226                float             SPARSE  NULL,
float227                float             SPARSE  NULL,
float228                float             SPARSE  NULL,
float229                float             SPARSE  NULL,
float230                float             SPARSE  NULL,
float231                float             SPARSE  NULL,
float232                float             SPARSE  NULL,
float233                float             SPARSE  NULL,
float234                float             SPARSE  NULL,
float235                float             SPARSE  NULL,
float236                float             SPARSE  NULL,
float237                float             SPARSE  NULL,
float238                float             SPARSE  NULL,
float239                float             SPARSE  NULL,
float240                float             SPARSE  NULL,
float241                float             SPARSE  NULL,
float242                float             SPARSE  NULL,
float243                float             SPARSE  NULL,
float244                float             SPARSE  NULL,
float245                float             SPARSE  NULL,
float246                float             SPARSE  NULL,
float247                float             SPARSE  NULL,
float248                float             SPARSE  NULL,
float249                float             SPARSE  NULL,
float250                float             SPARSE  NULL,
float251                float             SPARSE  NULL,
float252                float             SPARSE  NULL,
float253                float             SPARSE  NULL,
float254                float             SPARSE  NULL,
float255                float             SPARSE  NULL,
```

```
float256                float             SPARSE  NULL,
float257                float             SPARSE  NULL,
float258                float             SPARSE  NULL,
float259                float             SPARSE  NULL,
float260                float             SPARSE  NULL,
float261                float             SPARSE  NULL,
float262                float             SPARSE  NULL,
float263                float             SPARSE  NULL,
float264                float             SPARSE  NULL,
float265                float             SPARSE  NULL,
float266                float             SPARSE  NULL,
float267                float             SPARSE  NULL,
float268                float             SPARSE  NULL,
float269                float             SPARSE  NULL,
float270                float             SPARSE  NULL,
float271                float             SPARSE  NULL,
float272                float             SPARSE  NULL,
float273                float             SPARSE  NULL,
float274                float             SPARSE  NULL,
float275                float             SPARSE  NULL,
float276                float             SPARSE  NULL,
float277                float             SPARSE  NULL,
float278                float             SPARSE  NULL,
float279                float             SPARSE  NULL,
float280                float             SPARSE  NULL,
float281                float             SPARSE  NULL,
float282                float             SPARSE  NULL,
float283                float             SPARSE  NULL,
float284                float             SPARSE  NULL,
float285                float             SPARSE  NULL,
float286                float             SPARSE  NULL,
float287                float             SPARSE  NULL,
float288                float             SPARSE  NULL,
float289                float             SPARSE  NULL,
float290                float             SPARSE  NULL,
float291                float             SPARSE  NULL,
float292                float             SPARSE  NULL,
float293                float             SPARSE  NULL,
float294                float             SPARSE  NULL,
float295                float             SPARSE  NULL,
float296                float             SPARSE  NULL,
float297                float             SPARSE  NULL,
float298                float             SPARSE  NULL,
float299                float             SPARSE  NULL,
float300                float             SPARSE  NULL,
float301                float             SPARSE  NULL,
float302                float             SPARSE  NULL,
float303                float             SPARSE  NULL,
float304                float             SPARSE  NULL,
float305                float             SPARSE  NULL,
float306                float             SPARSE  NULL,
float307                float             SPARSE  NULL,
float308                float             SPARSE  NULL,
float309                float             SPARSE  NULL,
float310                float             SPARSE  NULL,
float311                float             SPARSE  NULL,
float312                float             SPARSE  NULL,
float313                float             SPARSE  NULL,
float314                float             SPARSE  NULL,
float315                float             SPARSE  NULL,
float316                float             SPARSE  NULL,
float317                float             SPARSE  NULL,
float318                float             SPARSE  NULL,
float319                float             SPARSE  NULL,
float320                float             SPARSE  NULL,
float321                float             SPARSE  NULL,
float322                float             SPARSE  NULL,
float323                float             SPARSE  NULL,
float324                float             SPARSE  NULL,
```

```
float325                float             SPARSE  NULL,
float326                float             SPARSE  NULL,
float327                float             SPARSE  NULL,
float328                float             SPARSE  NULL,
float329                float             SPARSE  NULL,
float330                float             SPARSE  NULL,
float331                float             SPARSE  NULL,
float332                float             SPARSE  NULL,
float333                float             SPARSE  NULL,
float334                float             SPARSE  NULL,
float335                float             SPARSE  NULL,
float336                float             SPARSE  NULL,
float337                float             SPARSE  NULL,
float338                float             SPARSE  NULL,
float339                float             SPARSE  NULL,
float340                float             SPARSE  NULL,
float341                float             SPARSE  NULL,
float342                float             SPARSE  NULL,
float343                float             SPARSE  NULL,
float344                float             SPARSE  NULL,
float345                float             SPARSE  NULL,
float346                float             SPARSE  NULL,
float347                float             SPARSE  NULL,
float348                float             SPARSE  NULL,
float349                float             SPARSE  NULL,
float350                float             SPARSE  NULL,
float351                float             SPARSE  NULL,
float352                float             SPARSE  NULL,
float353                float             SPARSE  NULL,
float354                float             SPARSE  NULL,
float355                float             SPARSE  NULL,
float356                float             SPARSE  NULL,
float357                float             SPARSE  NULL,
float358                float             SPARSE  NULL,
float359                float             SPARSE  NULL,
float360                float             SPARSE  NULL,
float361                float             SPARSE  NULL,
float362                float             SPARSE  NULL,
float363                float             SPARSE  NULL,
float364                float             SPARSE  NULL,
float365                float             SPARSE  NULL,
float366                float             SPARSE  NULL,
float367                float             SPARSE  NULL,
float368                float             SPARSE  NULL,
float369                float             SPARSE  NULL,
float370                float             SPARSE  NULL,
float371                float             SPARSE  NULL,
float372                float             SPARSE  NULL,
float373                float             SPARSE  NULL,
float374                float             SPARSE  NULL,
float375                float             SPARSE  NULL,
float376                float             SPARSE  NULL,
float377                float             SPARSE  NULL,
float378                float             SPARSE  NULL,
float379                float             SPARSE  NULL,
float380                float             SPARSE  NULL,
float381                float             SPARSE  NULL,
float382                float             SPARSE  NULL,
float383                float             SPARSE  NULL,
float384                float             SPARSE  NULL,
float385                float             SPARSE  NULL,
float386                float             SPARSE  NULL,
float387                float             SPARSE  NULL,
float388                float             SPARSE  NULL,
float389                float             SPARSE  NULL,
float390                float             SPARSE  NULL,
float391                float             SPARSE  NULL,
float392                float             SPARSE  NULL,
float393                float             SPARSE  NULL,
```

```
float394                float             SPARSE  NULL,
float395                float             SPARSE  NULL,
float396                float             SPARSE  NULL,
float397                float             SPARSE  NULL,
float398                float             SPARSE  NULL,
float399                float             SPARSE  NULL,
float400                float             SPARSE  NULL,
float401                float             SPARSE  NULL,
float402                float             SPARSE  NULL,
float403                float             SPARSE  NULL,
float404                float             SPARSE  NULL,
float405                float             SPARSE  NULL,
float406                float             SPARSE  NULL,
float407                float             SPARSE  NULL,
float408                float             SPARSE  NULL,
float409                float             SPARSE  NULL,
float410                float             SPARSE  NULL,
float411                float             SPARSE  NULL,
float412                float             SPARSE  NULL,
float413                float             SPARSE  NULL,
float414                float             SPARSE  NULL,
float415                float             SPARSE  NULL,
float416                float             SPARSE  NULL,
float417                float             SPARSE  NULL,
float418                float             SPARSE  NULL,
float419                float             SPARSE  NULL,
float420                float             SPARSE  NULL,
float421                float             SPARSE  NULL,
float422                float             SPARSE  NULL,
float423                float             SPARSE  NULL,
float424                float             SPARSE  NULL,
float425                float             SPARSE  NULL,
float426                float             SPARSE  NULL,
float427                float             SPARSE  NULL,
float428                float             SPARSE  NULL,
float429                float             SPARSE  NULL,
float430                float             SPARSE  NULL,
float431                float             SPARSE  NULL,
float432                float             SPARSE  NULL,
float433                float             SPARSE  NULL,
float434                float             SPARSE  NULL,
float435                float             SPARSE  NULL,
float436                float             SPARSE  NULL,
float437                float             SPARSE  NULL,
float438                float             SPARSE  NULL,
float439                float             SPARSE  NULL,
float440                float             SPARSE  NULL,
float441                float             SPARSE  NULL,
float442                float             SPARSE  NULL,
float443                float             SPARSE  NULL,
float444                float             SPARSE  NULL,
float445                float             SPARSE  NULL,
float446                float             SPARSE  NULL,
float447                float             SPARSE  NULL,
float448                float             SPARSE  NULL,
float449                float             SPARSE  NULL,
float450                float             SPARSE  NULL,
float451                float             SPARSE  NULL,
float452                float             SPARSE  NULL,
float453                float             SPARSE  NULL,
float454                float             SPARSE  NULL,
float455                float             SPARSE  NULL,
float456                float             SPARSE  NULL,
float457                float             SPARSE  NULL,
float458                float             SPARSE  NULL,
float459                float             SPARSE  NULL,
float460                float             SPARSE  NULL,
float461                float             SPARSE  NULL,
float462                float             SPARSE  NULL,
```

```
float463                float             SPARSE  NULL,
float464                float             SPARSE  NULL,
float465                float             SPARSE  NULL,
float466                float             SPARSE  NULL,
float467                float             SPARSE  NULL,
float468                float             SPARSE  NULL,
float469                float             SPARSE  NULL,
float470                float             SPARSE  NULL,
float471                float             SPARSE  NULL,
float472                float             SPARSE  NULL,
float473                float             SPARSE  NULL,
float474                float             SPARSE  NULL,
float475                float             SPARSE  NULL,
float476                float             SPARSE  NULL,
float477                float             SPARSE  NULL,
float478                float             SPARSE  NULL,
float479                float             SPARSE  NULL,
float480                float             SPARSE  NULL,
float481                float             SPARSE  NULL,
float482                float             SPARSE  NULL,
float483                float             SPARSE  NULL,
float484                float             SPARSE  NULL,
float485                float             SPARSE  NULL,
float486                float             SPARSE  NULL,
float487                float             SPARSE  NULL,
float488                float             SPARSE  NULL,
float489                float             SPARSE  NULL,
float490                float             SPARSE  NULL,
float491                float             SPARSE  NULL,
float492                float             SPARSE  NULL,
float493                float             SPARSE  NULL,
float494                float             SPARSE  NULL,
float495                float             SPARSE  NULL,
float496                float             SPARSE  NULL,
float497                float             SPARSE  NULL,
float498                float             SPARSE  NULL,
float499                float             SPARSE  NULL,
float500                float             SPARSE  NULL,
float501                float             SPARSE  NULL,
float502                float             SPARSE  NULL,
float503                float             SPARSE  NULL,
float504                float             SPARSE  NULL,
float505                float             SPARSE  NULL,
float506                float             SPARSE  NULL,
float507                float             SPARSE  NULL,
float508                float             SPARSE  NULL,
float509                float             SPARSE  NULL,
float510                float             SPARSE  NULL,
float511                float             SPARSE  NULL,
float512                float             SPARSE  NULL,
float513                float             SPARSE  NULL,
float514                float             SPARSE  NULL,
float515                float             SPARSE  NULL,
float516                float             SPARSE  NULL,
float517                float             SPARSE  NULL,
float518                float             SPARSE  NULL,
float519                float             SPARSE  NULL,
float520                float             SPARSE  NULL,
float521                float             SPARSE  NULL,
float522                float             SPARSE  NULL,
float523                float             SPARSE  NULL,
float524                float             SPARSE  NULL,
float525                float             SPARSE  NULL,
float526                float             SPARSE  NULL,
float527                float             SPARSE  NULL,
float528                float             SPARSE  NULL,
float529                float             SPARSE  NULL,
float530                float             SPARSE  NULL,
float531                float             SPARSE  NULL,
```

```
float532                float             SPARSE  NULL,
float533                float             SPARSE  NULL,
float534                float             SPARSE  NULL,
float535                float             SPARSE  NULL,
float536                float             SPARSE  NULL,
float537                float             SPARSE  NULL,
float538                float             SPARSE  NULL,
float539                float             SPARSE  NULL,
float540                float             SPARSE  NULL,
float541                float             SPARSE  NULL,
float542                float             SPARSE  NULL,
float543                float             SPARSE  NULL,
float544                float             SPARSE  NULL,
float545                float             SPARSE  NULL,
float546                float             SPARSE  NULL,
float547                float             SPARSE  NULL,
float548                float             SPARSE  NULL,
float549                float             SPARSE  NULL,
float550                float             SPARSE  NULL,
int1                    int               SPARSE  NULL,
int2                    int               SPARSE  NULL,
int3                    int               SPARSE  NULL,
int4                    int               SPARSE  NULL,
int5                    int               SPARSE  NULL,
int6                    int               SPARSE  NULL,
int7                    int               SPARSE  NULL,
int8                    int               SPARSE  NULL,
int9                    int               SPARSE  NULL,
int10                   int               SPARSE  NULL,
int11                   int               SPARSE  NULL,
int12                   int               SPARSE  NULL,
int13                   int               SPARSE  NULL,
int14                   int               SPARSE  NULL,
int15                   int               SPARSE  NULL,
int16                   int               SPARSE  NULL,
int17                   int               SPARSE  NULL,
int18                   int               SPARSE  NULL,
int19                   int               SPARSE  NULL,
int20                   int               SPARSE  NULL,
int21                   int               SPARSE  NULL,
int22                   int               SPARSE  NULL,
int23                   int               SPARSE  NULL,
int24                   int               SPARSE  NULL,
int25                   int               SPARSE  NULL,
int26                   int               SPARSE  NULL,
int27                   int               SPARSE  NULL,
int28                   int               SPARSE  NULL,
int29                   int               SPARSE  NULL,
int30                   int               SPARSE  NULL,
int31                   int               SPARSE  NULL,
int32                   int               SPARSE  NULL,
int33                   int               SPARSE  NULL,
int34                   int               SPARSE  NULL,
int35                   int               SPARSE  NULL,
int36                   int               SPARSE  NULL,
int37                   int               SPARSE  NULL,
int38                   int               SPARSE  NULL,
int39                   int               SPARSE  NULL,
int40                   int               SPARSE  NULL,
int41                   int               SPARSE  NULL,
int42                   int               SPARSE  NULL,
int43                   int               SPARSE  NULL,
int44                   int               SPARSE  NULL,
int45                   int               SPARSE  NULL,
int46                   int               SPARSE  NULL,
int47                   int               SPARSE  NULL,
int48                   int               SPARSE  NULL,
int49                   int               SPARSE  NULL,
int50                   int               SPARSE  NULL,
```

```
    int51                int             SPARSE  NULL,
    int52                int             SPARSE  NULL,
    int53                int             SPARSE  NULL,
    int54                int             SPARSE  NULL,
    int55                int             SPARSE  NULL,
    int56                int             SPARSE  NULL,
    int57                int             SPARSE  NULL,
    int58                int             SPARSE  NULL,
    int59                int             SPARSE  NULL,
    int60                int             SPARSE  NULL,
    int61                int             SPARSE  NULL,
    int62                int             SPARSE  NULL,
    int63                int             SPARSE  NULL,
    int64                int             SPARSE  NULL,
    int65                int             SPARSE  NULL,
    int66                int             SPARSE  NULL,
    int67                int             SPARSE  NULL,
    int68                int             SPARSE  NULL,
    int69                int             SPARSE  NULL,
    int70                int             SPARSE  NULL,
    int71                int             SPARSE  NULL,
    int72                int             SPARSE  NULL,
    int73                int             SPARSE  NULL,
    int74                int             SPARSE  NULL,
    int75                int             SPARSE  NULL,
    int76                int             SPARSE  NULL,
    int77                int             SPARSE  NULL,
    int78                int             SPARSE  NULL,
    int79                int             SPARSE  NULL,
    int80                int             SPARSE  NULL,
    int81                int             SPARSE  NULL,
    int82                int             SPARSE  NULL,
    int83                int             SPARSE  NULL,
    int84                int             SPARSE  NULL,
    int85                int             SPARSE  NULL,
    int86                int             SPARSE  NULL,
    int87                int             SPARSE  NULL,
    int88                int             SPARSE  NULL,
    int89                int             SPARSE  NULL,
    int90                int             SPARSE  NULL,
    int91                int             SPARSE  NULL,
    int92                int             SPARSE  NULL,
    int93                int             SPARSE  NULL,
    int94                int             SPARSE  NULL,
    int95                int             SPARSE  NULL,
    int96                int             SPARSE  NULL,
    int97                int             SPARSE  NULL,
    int98                int             SPARSE  NULL,
    int99                int             SPARSE  NULL,
    int100               int             SPARSE  NULL,
    int101               int             SPARSE  NULL,
    int102               int             SPARSE  NULL,
    int103               int             SPARSE  NULL,
    int104               int             SPARSE  NULL,
    int105               int             SPARSE  NULL,
    int106               int             SPARSE  NULL,
    int107               int             SPARSE  NULL,
    int108               int             SPARSE  NULL,
    int109               int             SPARSE  NULL,
    int110               int             SPARSE  NULL,
    int111               int             SPARSE  NULL,
    int112               int             SPARSE  NULL,
    int113               int             SPARSE  NULL,
    int114               int             SPARSE  NULL,
    int115               int             SPARSE  NULL,
    int116               int             SPARSE  NULL,
    int117               int             SPARSE  NULL,
    int118               int             SPARSE  NULL,
    int119               int             SPARSE  NULL,
```

```
    int120                int             SPARSE  NULL,
    int121                int             SPARSE  NULL,
    int122                int             SPARSE  NULL,
    int123                int             SPARSE  NULL,
    int124                int             SPARSE  NULL,
    int125                int             SPARSE  NULL,
    int126                int             SPARSE  NULL,
    int127                int             SPARSE  NULL,
    int128                int             SPARSE  NULL,
    int129                int             SPARSE  NULL,
    int130                int             SPARSE  NULL,
    int131                int             SPARSE  NULL,
    int132                int             SPARSE  NULL,
    int133                int             SPARSE  NULL,
    int134                int             SPARSE  NULL,
    int135                int             SPARSE  NULL,
    int136                int             SPARSE  NULL,
    int137                int             SPARSE  NULL,
    int138                int             SPARSE  NULL,
    int139                int             SPARSE  NULL,
    int140                int             SPARSE  NULL,
    int141                int             SPARSE  NULL,
    int142                int             SPARSE  NULL,
    int143                int             SPARSE  NULL,
    int144                int             SPARSE  NULL,
    int145                int             SPARSE  NULL,
    int146                int             SPARSE  NULL,
    int147                int             SPARSE  NULL,
    int148                int             SPARSE  NULL,
    int149                int             SPARSE  NULL,
    int150                int             SPARSE  NULL,
    int151                int             SPARSE  NULL,
    int152                int             SPARSE  NULL,
    int153                int             SPARSE  NULL,
    int154                int             SPARSE  NULL,
    int155                int             SPARSE  NULL,
    int156                int             SPARSE  NULL,
    int157                int             SPARSE  NULL,
    int158                int             SPARSE  NULL,
    int159                int             SPARSE  NULL,
    int160                int             SPARSE  NULL,
    int161                int             SPARSE  NULL,
    int162                int             SPARSE  NULL,
    int163                int             SPARSE  NULL,
    int164                int             SPARSE  NULL,
    int165                int             SPARSE  NULL,
    int166                int             SPARSE  NULL,
    int167                int             SPARSE  NULL,
    int168                int             SPARSE  NULL,
    int169                int             SPARSE  NULL,
    int170                int             SPARSE  NULL,
    int171                int             SPARSE  NULL,
    int172                int             SPARSE  NULL,
    int173                int             SPARSE  NULL,
    int174                int             SPARSE  NULL,
    int175                int             SPARSE  NULL,
    int176                int             SPARSE  NULL,
    int177                int             SPARSE  NULL,
    int178                int             SPARSE  NULL,
    int179                int             SPARSE  NULL,
    int180                int             SPARSE  NULL,
    int181                int             SPARSE  NULL,
    int182                int             SPARSE  NULL,
    int183                int             SPARSE  NULL,
    int184                int             SPARSE  NULL,
    int185                int             SPARSE  NULL,
    int186                int             SPARSE  NULL,
    int187                int             SPARSE  NULL,
    int188                int             SPARSE  NULL,
```

```
    int189                int             SPARSE  NULL,
    int190                int             SPARSE  NULL,
    int191                int             SPARSE  NULL,
    int192                int             SPARSE  NULL,
    int193                int             SPARSE  NULL,
    int194                int             SPARSE  NULL,
    int195                int             SPARSE  NULL,
    int196                int             SPARSE  NULL,
    int197                int             SPARSE  NULL,
    int198                int             SPARSE  NULL,
    int199                int             SPARSE  NULL,
    int200                int             SPARSE  NULL,
    int201                int             SPARSE  NULL,
    int202                int             SPARSE  NULL,
    int203                int             SPARSE  NULL,
    int204                int             SPARSE  NULL,
    int205                int             SPARSE  NULL,
    int206                int             SPARSE  NULL,
    int207                int             SPARSE  NULL,
    int208                int             SPARSE  NULL,
    int209                int             SPARSE  NULL,
    int210                int             SPARSE  NULL,
    int211                int             SPARSE  NULL,
    int212                int             SPARSE  NULL,
    int213                int             SPARSE  NULL,
    int214                int             SPARSE  NULL,
    int215                int             SPARSE  NULL,
    int216                int             SPARSE  NULL,
    int217                int             SPARSE  NULL,
    int218                int             SPARSE  NULL,
    int219                int             SPARSE  NULL,
    int220                int             SPARSE  NULL,
    int221                int             SPARSE  NULL,
    int222                int             SPARSE  NULL,
    int223                int             SPARSE  NULL,
    int224                int             SPARSE  NULL,
    int225                int             SPARSE  NULL,
    int226                int             SPARSE  NULL,
    int227                int             SPARSE  NULL,
    int228                int             SPARSE  NULL,
    int229                int             SPARSE  NULL,
    int230                int             SPARSE  NULL,
    int231                int             SPARSE  NULL,
    int232                int             SPARSE  NULL,
    int233                int             SPARSE  NULL,
    int234                int             SPARSE  NULL,
    int235                int             SPARSE  NULL,
    int236                int             SPARSE  NULL,
    int237                int             SPARSE  NULL,
    int238                int             SPARSE  NULL,
    int239                int             SPARSE  NULL,
    int240                int             SPARSE  NULL,
    int241                int             SPARSE  NULL,
    int242                int             SPARSE  NULL,
    int243                int             SPARSE  NULL,
    int244                int             SPARSE  NULL,
    int245                int             SPARSE  NULL,
    int246                int             SPARSE  NULL,
    int247                int             SPARSE  NULL,
    int248                int             SPARSE  NULL,
    int249                int             SPARSE  NULL,
    int250                int             SPARSE  NULL,
    int251                int             SPARSE  NULL,
    int252                int             SPARSE  NULL,
    int253                int             SPARSE  NULL,
    int254                int             SPARSE  NULL,
    int255                int             SPARSE  NULL,
    int256                int             SPARSE  NULL,
    int257                int             SPARSE  NULL,
```

```
int258                int             SPARSE  NULL,
int259                int             SPARSE  NULL,
int260                int             SPARSE  NULL,
int261                int             SPARSE  NULL,
int262                int             SPARSE  NULL,
int263                int             SPARSE  NULL,
int264                int             SPARSE  NULL,
int265                int             SPARSE  NULL,
int266                int             SPARSE  NULL,
int267                int             SPARSE  NULL,
int268                int             SPARSE  NULL,
int269                int             SPARSE  NULL,
int270                int             SPARSE  NULL,
int271                int             SPARSE  NULL,
int272                int             SPARSE  NULL,
int273                int             SPARSE  NULL,
int274                int             SPARSE  NULL,
int275                int             SPARSE  NULL,
int276                int             SPARSE  NULL,
int277                int             SPARSE  NULL,
int278                int             SPARSE  NULL,
int279                int             SPARSE  NULL,
int280                int             SPARSE  NULL,
int281                int             SPARSE  NULL,
int282                int             SPARSE  NULL,
int283                int             SPARSE  NULL,
int284                int             SPARSE  NULL,
int285                int             SPARSE  NULL,
int286                int             SPARSE  NULL,
int287                int             SPARSE  NULL,
int288                int             SPARSE  NULL,
int289                int             SPARSE  NULL,
int290                int             SPARSE  NULL,
int291                int             SPARSE  NULL,
int292                int             SPARSE  NULL,
int293                int             SPARSE  NULL,
int294                int             SPARSE  NULL,
int295                int             SPARSE  NULL,
int296                int             SPARSE  NULL,
int297                int             SPARSE  NULL,
int298                int             SPARSE  NULL,
int299                int             SPARSE  NULL,
int300                int             SPARSE  NULL,
int301                int             SPARSE  NULL,
int302                int             SPARSE  NULL,
int303                int             SPARSE  NULL,
int304                int             SPARSE  NULL,
int305                int             SPARSE  NULL,
int306                int             SPARSE  NULL,
int307                int             SPARSE  NULL,
int308                int             SPARSE  NULL,
int309                int             SPARSE  NULL,
int310                int             SPARSE  NULL,
int311                int             SPARSE  NULL,
int312                int             SPARSE  NULL,
int313                int             SPARSE  NULL,
int314                int             SPARSE  NULL,
int315                int             SPARSE  NULL,
int316                int             SPARSE  NULL,
int317                int             SPARSE  NULL,
int318                int             SPARSE  NULL,
int319                int             SPARSE  NULL,
int320                int             SPARSE  NULL,
int321                int             SPARSE  NULL,
int322                int             SPARSE  NULL,
int323                int             SPARSE  NULL,
int324                int             SPARSE  NULL,
int325                int             SPARSE  NULL,
int326                int             SPARSE  NULL,
```

```
    int327                int             SPARSE  NULL,
    int328                int             SPARSE  NULL,
    int329                int             SPARSE  NULL,
    int330                int             SPARSE  NULL,
    int331                int             SPARSE  NULL,
    int332                int             SPARSE  NULL,
    int333                int             SPARSE  NULL,
    int334                int             SPARSE  NULL,
    int335                int             SPARSE  NULL,
    int336                int             SPARSE  NULL,
    int337                int             SPARSE  NULL,
    int338                int             SPARSE  NULL,
    int339                int             SPARSE  NULL,
    int340                int             SPARSE  NULL,
    int341                int             SPARSE  NULL,
    int342                int             SPARSE  NULL,
    int343                int             SPARSE  NULL,
    int344                int             SPARSE  NULL,
    int345                int             SPARSE  NULL,
    int346                int             SPARSE  NULL,
    int347                int             SPARSE  NULL,
    int348                int             SPARSE  NULL,
    int349                int             SPARSE  NULL,
    int350                int             SPARSE  NULL,
    int351                int             SPARSE  NULL,
    int352                int             SPARSE  NULL,
    int353                int             SPARSE  NULL,
    int354                int             SPARSE  NULL,
    int355                int             SPARSE  NULL,
    int356                int             SPARSE  NULL,
    int357                int             SPARSE  NULL,
    int358                int             SPARSE  NULL,
    int359                int             SPARSE  NULL,
    int360                int             SPARSE  NULL,
    int361                int             SPARSE  NULL,
    int362                int             SPARSE  NULL,
    int363                int             SPARSE  NULL,
    int364                int             SPARSE  NULL,
    int365                int             SPARSE  NULL,
    int366                int             SPARSE  NULL,
    int367                int             SPARSE  NULL,
    int368                int             SPARSE  NULL,
    int369                int             SPARSE  NULL,
    int370                int             SPARSE  NULL,
    int371                int             SPARSE  NULL,
    int372                int             SPARSE  NULL,
    int373                int             SPARSE  NULL,
    int374                int             SPARSE  NULL,
    int375                int             SPARSE  NULL,
    int376                int             SPARSE  NULL,
    int377                int             SPARSE  NULL,
    int378                int             SPARSE  NULL,
    int379                int             SPARSE  NULL,
    int380                int             SPARSE  NULL,
    int381                int             SPARSE  NULL,
    int382                int             SPARSE  NULL,
    int383                int             SPARSE  NULL,
    int384                int             SPARSE  NULL,
    int385                int             SPARSE  NULL,
    int386                int             SPARSE  NULL,
    int387                int             SPARSE  NULL,
    int388                int             SPARSE  NULL,
    int389                int             SPARSE  NULL,
    int390                int             SPARSE  NULL,
    int391                int             SPARSE  NULL,
    int392                int             SPARSE  NULL,
    int393                int             SPARSE  NULL,
    int394                int             SPARSE  NULL,
    int395                int             SPARSE  NULL,
```

```
    int396                  int                 SPARSE  NULL,
    int397                  int                 SPARSE  NULL,
    int398                  int                 SPARSE  NULL,
    int399                  int                 SPARSE  NULL,
    int400                  int                 SPARSE  NULL,
    int401                  int                 SPARSE  NULL,
    int402                  int                 SPARSE  NULL,
    int403                  int                 SPARSE  NULL,
    int404                  int                 SPARSE  NULL,
    int405                  int                 SPARSE  NULL,
    int406                  int                 SPARSE  NULL,
    int407                  int                 SPARSE  NULL,
    int408                  int                 SPARSE  NULL,
    int409                  int                 SPARSE  NULL,
    int410                  int                 SPARSE  NULL,
    int411                  int                 SPARSE  NULL,
    int412                  int                 SPARSE  NULL,
    int413                  int                 SPARSE  NULL,
    int414                  int                 SPARSE  NULL,
    int415                  int                 SPARSE  NULL,
    int416                  int                 SPARSE  NULL,
    int417                  int                 SPARSE  NULL,
    int418                  int                 SPARSE  NULL,
    int419                  int                 SPARSE  NULL,
    int420                  int                 SPARSE  NULL,
    int421                  int                 SPARSE  NULL,
    int422                  int                 SPARSE  NULL,
    int423                  int                 SPARSE  NULL,
    int424                  int                 SPARSE  NULL,
    int425                  int                 SPARSE  NULL,
    int426                  int                 SPARSE  NULL,
    int427                  int                 SPARSE  NULL,
    int428                  int                 SPARSE  NULL,
    int429                  int                 SPARSE  NULL,
    int430                  int                 SPARSE  NULL,
    int431                  int                 SPARSE  NULL,
    int432                  int                 SPARSE  NULL,
    int433                  int                 SPARSE  NULL,
    int434                  int                 SPARSE  NULL,
    int435                  int                 SPARSE  NULL,
    int436                  int                 SPARSE  NULL,
    int437                  int                 SPARSE  NULL,
    int438                  int                 SPARSE  NULL,
    int439                  int                 SPARSE  NULL,
    int440                  int                 SPARSE  NULL,
    int441                  int                 SPARSE  NULL,
    int442                  int                 SPARSE  NULL,
    int443                  int                 SPARSE  NULL,
    int444                  int                 SPARSE  NULL,
    int445                  int                 SPARSE  NULL,
    int446                  int                 SPARSE  NULL,
    int447                  int                 SPARSE  NULL,
    int448                  int                 SPARSE  NULL,
    int449                  int                 SPARSE  NULL,
    int450                  int                 SPARSE  NULL,
    int451                  int                 SPARSE  NULL,
    int452                  int                 SPARSE  NULL,
    int453                  int                 SPARSE  NULL,
    int454                  int                 SPARSE  NULL,
    int455                  int                 SPARSE  NULL,
    int456                  int                 SPARSE  NULL,
    int457                  int                 SPARSE  NULL,
    int458                  int                 SPARSE  NULL,
    int459                  int                 SPARSE  NULL,
    int460                  int                 SPARSE  NULL,
    int461                  int                 SPARSE  NULL,
    int462                  int                 SPARSE  NULL,
    int463                  int                 SPARSE  NULL,
    int464                  int                 SPARSE  NULL,
```

```
    int465                int             SPARSE  NULL,
    int466                int             SPARSE  NULL,
    int467                int             SPARSE  NULL,
    int468                int             SPARSE  NULL,
    int469                int             SPARSE  NULL,
    int470                int             SPARSE  NULL,
    int471                int             SPARSE  NULL,
    int472                int             SPARSE  NULL,
    int473                int             SPARSE  NULL,
    int474                int             SPARSE  NULL,
    int475                int             SPARSE  NULL,
    int476                int             SPARSE  NULL,
    int477                int             SPARSE  NULL,
    int478                int             SPARSE  NULL,
    int479                int             SPARSE  NULL,
    int480                int             SPARSE  NULL,
    int481                int             SPARSE  NULL,
    int482                int             SPARSE  NULL,
    int483                int             SPARSE  NULL,
    int484                int             SPARSE  NULL,
    int485                int             SPARSE  NULL,
    int486                int             SPARSE  NULL,
    int487                int             SPARSE  NULL,
    int488                int             SPARSE  NULL,
    int489                int             SPARSE  NULL,
    int490                int             SPARSE  NULL,
    int491                int             SPARSE  NULL,
    int492                int             SPARSE  NULL,
    int493                int             SPARSE  NULL,
    int494                int             SPARSE  NULL,
    int495                int             SPARSE  NULL,
    int496                int             SPARSE  NULL,
    int497                int             SPARSE  NULL,
    int498                int             SPARSE  NULL,
    int499                int             SPARSE  NULL,
    int500                int             SPARSE  NULL,
    int501                int             SPARSE  NULL,
    int502                int             SPARSE  NULL,
    int503                int             SPARSE  NULL,
    int504                int             SPARSE  NULL,
    int505                int             SPARSE  NULL,
    int506                int             SPARSE  NULL,
    int507                int             SPARSE  NULL,
    int508                int             SPARSE  NULL,
    int509                int             SPARSE  NULL,
    int510                int             SPARSE  NULL,
    int511                int             SPARSE  NULL,
    int512                int             SPARSE  NULL,
    int513                int             SPARSE  NULL,
    int514                int             SPARSE  NULL,
    int515                int             SPARSE  NULL,
    int516                int             SPARSE  NULL,
    int517                int             SPARSE  NULL,
    int518                int             SPARSE  NULL,
    int519                int             SPARSE  NULL,
    int520                int             SPARSE  NULL,
    int521                int             SPARSE  NULL,
    int522                int             SPARSE  NULL,
    int523                int             SPARSE  NULL,
    int524                int             SPARSE  NULL,
    int525                int             SPARSE  NULL,
    int526                int             SPARSE  NULL,
    int527                int             SPARSE  NULL,
    int528                int             SPARSE  NULL,
    int529                int             SPARSE  NULL,
    int530                int             SPARSE  NULL,
    int531                int             SPARSE  NULL,
    int532                int             SPARSE  NULL,
    int533                int             SPARSE  NULL,
```

```
    int534                  int                 SPARSE  NULL,
    int535                  int                 SPARSE  NULL,
    int536                  int                 SPARSE  NULL,
    int537                  int                 SPARSE  NULL,
    int538                  int                 SPARSE  NULL,
    int539                  int                 SPARSE  NULL,
    int540                  int                 SPARSE  NULL,
    int541                  int                 SPARSE  NULL,
    int542                  int                 SPARSE  NULL,
    int543                  int                 SPARSE  NULL,
    int544                  int                 SPARSE  NULL,
    int545                  int                 SPARSE  NULL,
    int546                  int                 SPARSE  NULL,
    int547                  int                 SPARSE  NULL,
    int548                  int                 SPARSE  NULL,
    int549                  int                 SPARSE  NULL,
    int550                  int                 SPARSE  NULL,
    int551                  int                 SPARSE  NULL,
    int552                  int                 SPARSE  NULL,
    int553                  int                 SPARSE  NULL,
    int554                  int                 SPARSE  NULL,
    int555                  int                 SPARSE  NULL,
    int556                  int                 SPARSE  NULL,
    int557                  int                 SPARSE  NULL,
    int558                  int                 SPARSE  NULL,
    int559                  int                 SPARSE  NULL,
    int560                  int                 SPARSE  NULL,
    int561                  int                 SPARSE  NULL,
    int562                  int                 SPARSE  NULL,
    int563                  int                 SPARSE  NULL,
    int564                  int                 SPARSE  NULL,
    int565                  int                 SPARSE  NULL,
    int566                  int                 SPARSE  NULL,
    int567                  int                 SPARSE  NULL,
    int568                  int                 SPARSE  NULL,
    int569                  int                 SPARSE  NULL,
    int570                  int                 SPARSE  NULL,
    int571                  int                 SPARSE  NULL,
    int572                  int                 SPARSE  NULL,
    int573                  int                 SPARSE  NULL,
    int574                  int                 SPARSE  NULL,
    int575                  int                 SPARSE  NULL,
    int576                  int                 SPARSE  NULL,
    int577                  int                 SPARSE  NULL,
    int578                  int                 SPARSE  NULL,
    int579                  int                 SPARSE  NULL,
    int580                  int                 SPARSE  NULL,
    int581                  int                 SPARSE  NULL,
    int582                  int                 SPARSE  NULL,
    int583                  int                 SPARSE  NULL,
    int584                  int                 SPARSE  NULL,
    int585                  int                 SPARSE  NULL,
    int586                  int                 SPARSE  NULL,
    int587                  int                 SPARSE  NULL,
    int588                  int                 SPARSE  NULL,
    int589                  int                 SPARSE  NULL,
    int590                  int                 SPARSE  NULL,
    int591                  int                 SPARSE  NULL,
    int592                  int                 SPARSE  NULL,
    int593                  int                 SPARSE  NULL,
    int594                  int                 SPARSE  NULL,
    int595                  int                 SPARSE  NULL,
    int596                  int                 SPARSE  NULL,
    int597                  int                 SPARSE  NULL,
    int598                  int                 SPARSE  NULL,
    int599                  int                 SPARSE  NULL,
    int600                  int                 SPARSE  NULL,
    int601                  int                 SPARSE  NULL,
    int602                  int                 SPARSE  NULL,
```

```
int603                int             SPARSE  NULL,
int604                int             SPARSE  NULL,
int605                int             SPARSE  NULL,
int606                int             SPARSE  NULL,
int607                int             SPARSE  NULL,
int608                int             SPARSE  NULL,
int609                int             SPARSE  NULL,
int610                int             SPARSE  NULL,
int611                int             SPARSE  NULL,
int612                int             SPARSE  NULL,
int613                int             SPARSE  NULL,
int614                int             SPARSE  NULL,
int615                int             SPARSE  NULL,
int616                int             SPARSE  NULL,
int617                int             SPARSE  NULL,
int618                int             SPARSE  NULL,
int619                int             SPARSE  NULL,
int620                int             SPARSE  NULL,
int621                int             SPARSE  NULL,
int622                int             SPARSE  NULL,
int623                int             SPARSE  NULL,
int624                int             SPARSE  NULL,
int625                int             SPARSE  NULL,
int626                int             SPARSE  NULL,
int627                int             SPARSE  NULL,
int628                int             SPARSE  NULL,
int629                int             SPARSE  NULL,
int630                int             SPARSE  NULL,
int631                int             SPARSE  NULL,
int632                int             SPARSE  NULL,
int633                int             SPARSE  NULL,
int634                int             SPARSE  NULL,
int635                int             SPARSE  NULL,
int636                int             SPARSE  NULL,
int637                int             SPARSE  NULL,
int638                int             SPARSE  NULL,
int639                int             SPARSE  NULL,
int640                int             SPARSE  NULL,
int641                int             SPARSE  NULL,
int642                int             SPARSE  NULL,
int643                int             SPARSE  NULL,
int644                int             SPARSE  NULL,
int645                int             SPARSE  NULL,
int646                int             SPARSE  NULL,
int647                int             SPARSE  NULL,
int648                int             SPARSE  NULL,
int649                int             SPARSE  NULL,
int650                int             SPARSE  NULL,
int651                int             SPARSE  NULL,
int652                int             SPARSE  NULL,
int653                int             SPARSE  NULL,
int654                int             SPARSE  NULL,
int655                int             SPARSE  NULL,
int656                int             SPARSE  NULL,
int657                int             SPARSE  NULL,
int658                int             SPARSE  NULL,
int659                int             SPARSE  NULL,
int660                int             SPARSE  NULL,
int661                int             SPARSE  NULL,
int662                int             SPARSE  NULL,
int663                int             SPARSE  NULL,
int664                int             SPARSE  NULL,
int665                int             SPARSE  NULL,
int666                int             SPARSE  NULL,
int667                int             SPARSE  NULL,
int668                int             SPARSE  NULL,
int669                int             SPARSE  NULL,
int670                int             SPARSE  NULL,
int671                int             SPARSE  NULL,
```

```
    int672                int             SPARSE  NULL,
    int673                int             SPARSE  NULL,
    int674                int             SPARSE  NULL,
    int675                int             SPARSE  NULL,
    int676                int             SPARSE  NULL,
    int677                int             SPARSE  NULL,
    int678                int             SPARSE  NULL,
    int679                int             SPARSE  NULL,
    int680                int             SPARSE  NULL,
    int681                int             SPARSE  NULL,
    int682                int             SPARSE  NULL,
    int683                int             SPARSE  NULL,
    int684                int             SPARSE  NULL,
    int685                int             SPARSE  NULL,
    int686                int             SPARSE  NULL,
    int687                int             SPARSE  NULL,
    int688                int             SPARSE  NULL,
    int689                int             SPARSE  NULL,
    int690                int             SPARSE  NULL,
    int691                int             SPARSE  NULL,
    int692                int             SPARSE  NULL,
    int693                int             SPARSE  NULL,
    int694                int             SPARSE  NULL,
    int695                int             SPARSE  NULL,
    int696                int             SPARSE  NULL,
    int697                int             SPARSE  NULL,
    int698                int             SPARSE  NULL,
    int699                int             SPARSE  NULL,
    int700                int             SPARSE  NULL,
    int701                int             SPARSE  NULL,
    int702                int             SPARSE  NULL,
    int703                int             SPARSE  NULL,
    int704                int             SPARSE  NULL,
    int705                int             SPARSE  NULL,
    int706                int             SPARSE  NULL,
    int707                int             SPARSE  NULL,
    int708                int             SPARSE  NULL,
    int709                int             SPARSE  NULL,
    int710                int             SPARSE  NULL,
    int711                int             SPARSE  NULL,
    int712                int             SPARSE  NULL,
    int713                int             SPARSE  NULL,
    int714                int             SPARSE  NULL,
    int715                int             SPARSE  NULL,
    int716                int             SPARSE  NULL,
    int717                int             SPARSE  NULL,
    int718                int             SPARSE  NULL,
    int719                int             SPARSE  NULL,
    int720                int             SPARSE  NULL,
    int721                int             SPARSE  NULL,
    int722                int             SPARSE  NULL,
    int723                int             SPARSE  NULL,
    int724                int             SPARSE  NULL,
    int725                int             SPARSE  NULL,
    int726                int             SPARSE  NULL,
    int727                int             SPARSE  NULL,
    int728                int             SPARSE  NULL,
    int729                int             SPARSE  NULL,
    int730                int             SPARSE  NULL,
    int731                int             SPARSE  NULL,
    int732                int             SPARSE  NULL,
    int733                int             SPARSE  NULL,
    int734                int             SPARSE  NULL,
    int735                int             SPARSE  NULL,
    int736                int             SPARSE  NULL,
    int737                int             SPARSE  NULL,
    int738                int             SPARSE  NULL,
    int739                int             SPARSE  NULL,
    int740                int             SPARSE  NULL,
```

```
int741                int              SPARSE  NULL,
int742                int              SPARSE  NULL,
int743                int              SPARSE  NULL,
int744                int              SPARSE  NULL,
int745                int              SPARSE  NULL,
int746                int              SPARSE  NULL,
int747                int              SPARSE  NULL,
int748                int              SPARSE  NULL,
int749                int              SPARSE  NULL,
int750                int              SPARSE  NULL,
ntext1                nvarchar(max)    SPARSE  NULL,
ntext2                nvarchar(max)    SPARSE  NULL,
ntext3                nvarchar(max)    SPARSE  NULL,
ntext4                nvarchar(max)    SPARSE  NULL,
ntext5                nvarchar(max)    SPARSE  NULL,
ntext6                nvarchar(max)    SPARSE  NULL,
ntext7                nvarchar(max)    SPARSE  NULL,
ntext8                nvarchar(max)    SPARSE  NULL,
ntext9                nvarchar(max)    SPARSE  NULL,
ntext10               nvarchar(max)    SPARSE  NULL,
ntext11               nvarchar(max)    SPARSE  NULL,
ntext12               nvarchar(max)    SPARSE  NULL,
ntext13               nvarchar(max)    SPARSE  NULL,
ntext14               nvarchar(max)    SPARSE  NULL,
ntext15               nvarchar(max)    SPARSE  NULL,
ntext16               nvarchar(max)    SPARSE  NULL,
ntext17               nvarchar(max)    SPARSE  NULL,
ntext18               nvarchar(max)    SPARSE  NULL,
ntext19               nvarchar(max)    SPARSE  NULL,
ntext20               nvarchar(max)    SPARSE  NULL,
ntext21               nvarchar(max)    SPARSE  NULL,
ntext22               nvarchar(max)    SPARSE  NULL,
ntext23               nvarchar(max)    SPARSE  NULL,
ntext24               nvarchar(max)    SPARSE  NULL,
ntext25               nvarchar(max)    SPARSE  NULL,
ntext26               nvarchar(max)    SPARSE  NULL,
ntext27               nvarchar(max)    SPARSE  NULL,
ntext28               nvarchar(max)    SPARSE  NULL,
ntext29               nvarchar(max)    SPARSE  NULL,
ntext30               nvarchar(max)    SPARSE  NULL,
ntext31               nvarchar(max)    SPARSE  NULL,
ntext32               nvarchar(max)    SPARSE  NULL,
ntext33               nvarchar(max)    SPARSE  NULL,
ntext34               nvarchar(max)    SPARSE  NULL,
ntext35               nvarchar(max)    SPARSE  NULL,
ntext36               nvarchar(max)    SPARSE  NULL,
ntext37               nvarchar(max)    SPARSE  NULL,
ntext38               nvarchar(max)    SPARSE  NULL,
ntext39               nvarchar(max)    SPARSE  NULL,
ntext40               nvarchar(max)    SPARSE  NULL,
ntext41               nvarchar(max)    SPARSE  NULL,
ntext42               nvarchar(max)    SPARSE  NULL,
ntext43               nvarchar(max)    SPARSE  NULL,
ntext44               nvarchar(max)    SPARSE  NULL,
ntext45               nvarchar(max)    SPARSE  NULL,
ntext46               nvarchar(max)    SPARSE  NULL,
ntext47               nvarchar(max)    SPARSE  NULL,
ntext48               nvarchar(max)    SPARSE  NULL,
ntext49               nvarchar(max)    SPARSE  NULL,
ntext50               nvarchar(max)    SPARSE  NULL,
ntext51               nvarchar(max)    SPARSE  NULL,
ntext52               nvarchar(max)    SPARSE  NULL,
ntext53               nvarchar(max)    SPARSE  NULL,
ntext54               nvarchar(max)    SPARSE  NULL,
ntext55               nvarchar(max)    SPARSE  NULL,
ntext56               nvarchar(max)    SPARSE  NULL,
ntext57               nvarchar(max)    SPARSE  NULL,
ntext58               nvarchar(max)    SPARSE  NULL,
ntext59               nvarchar(max)    SPARSE  NULL,
```

```
ntext60                nvarchar(max)     SPARSE  NULL,
ntext61                nvarchar(max)     SPARSE  NULL,
ntext62                nvarchar(max)     SPARSE  NULL,
ntext63                nvarchar(max)     SPARSE  NULL,
ntext64                nvarchar(max)     SPARSE  NULL,
ntext65                nvarchar(max)     SPARSE  NULL,
ntext66                nvarchar(max)     SPARSE  NULL,
ntext67                nvarchar(max)     SPARSE  NULL,
ntext68                nvarchar(max)     SPARSE  NULL,
ntext69                nvarchar(max)     SPARSE  NULL,
ntext70                nvarchar(max)     SPARSE  NULL,
ntext71                nvarchar(max)     SPARSE  NULL,
ntext72                nvarchar(max)     SPARSE  NULL,
ntext73                nvarchar(max)     SPARSE  NULL,
ntext74                nvarchar(max)     SPARSE  NULL,
ntext75                nvarchar(max)     SPARSE  NULL,
ntext76                nvarchar(max)     SPARSE  NULL,
ntext77                nvarchar(max)     SPARSE  NULL,
ntext78                nvarchar(max)     SPARSE  NULL,
ntext79                nvarchar(max)     SPARSE  NULL,
ntext80                nvarchar(max)     SPARSE  NULL,
ntext81                nvarchar(max)     SPARSE  NULL,
ntext82                nvarchar(max)     SPARSE  NULL,
ntext83                nvarchar(max)     SPARSE  NULL,
ntext84                nvarchar(max)     SPARSE  NULL,
ntext85                nvarchar(max)     SPARSE  NULL,
ntext86                nvarchar(max)     SPARSE  NULL,
ntext87                nvarchar(max)     SPARSE  NULL,
ntext88                nvarchar(max)     SPARSE  NULL,
ntext89                nvarchar(max)     SPARSE  NULL,
ntext90                nvarchar(max)     SPARSE  NULL,
ntext91                nvarchar(max)     SPARSE  NULL,
ntext92                nvarchar(max)     SPARSE  NULL,
ntext93                nvarchar(max)     SPARSE  NULL,
ntext94                nvarchar(max)     SPARSE  NULL,
ntext95                nvarchar(max)     SPARSE  NULL,
ntext96                nvarchar(max)     SPARSE  NULL,
ntext97                nvarchar(max)     SPARSE  NULL,
ntext98                nvarchar(max)     SPARSE  NULL,
ntext99                nvarchar(max)     SPARSE  NULL,
ntext100               nvarchar(max)     SPARSE  NULL,
ntext101               nvarchar(max)     SPARSE  NULL,
ntext102               nvarchar(max)     SPARSE  NULL,
ntext103               nvarchar(max)     SPARSE  NULL,
ntext104               nvarchar(max)     SPARSE  NULL,
ntext105               nvarchar(max)     SPARSE  NULL,
ntext106               nvarchar(max)     SPARSE  NULL,
ntext107               nvarchar(max)     SPARSE  NULL,
ntext108               nvarchar(max)     SPARSE  NULL,
ntext109               nvarchar(max)     SPARSE  NULL,
ntext110               nvarchar(max)     SPARSE  NULL,
ntext111               nvarchar(max)     SPARSE  NULL,
ntext112               nvarchar(max)     SPARSE  NULL,
ntext113               nvarchar(max)     SPARSE  NULL,
ntext114               nvarchar(max)     SPARSE  NULL,
ntext115               nvarchar(max)     SPARSE  NULL,
ntext116               nvarchar(max)     SPARSE  NULL,
ntext117               nvarchar(max)     SPARSE  NULL,
ntext118               nvarchar(max)     SPARSE  NULL,
ntext119               nvarchar(max)     SPARSE  NULL,
ntext120               nvarchar(max)     SPARSE  NULL,
ntext121               nvarchar(max)     SPARSE  NULL,
ntext122               nvarchar(max)     SPARSE  NULL,
ntext123               nvarchar(max)     SPARSE  NULL,
ntext124               nvarchar(max)     SPARSE  NULL,
ntext125               nvarchar(max)     SPARSE  NULL,
ntext126               nvarchar(max)     SPARSE  NULL,
ntext127               nvarchar(max)     SPARSE  NULL,
ntext128               nvarchar(max)     SPARSE  NULL,
```

```
ntext129                nvarchar(max)     SPARSE  NULL,
ntext130                nvarchar(max)     SPARSE  NULL,
ntext131                nvarchar(max)     SPARSE  NULL,
ntext132                nvarchar(max)     SPARSE  NULL,
ntext133                nvarchar(max)     SPARSE  NULL,
ntext134                nvarchar(max)     SPARSE  NULL,
ntext135                nvarchar(max)     SPARSE  NULL,
ntext136                nvarchar(max)     SPARSE  NULL,
ntext137                nvarchar(max)     SPARSE  NULL,
ntext138                nvarchar(max)     SPARSE  NULL,
ntext139                nvarchar(max)     SPARSE  NULL,
ntext140                nvarchar(max)     SPARSE  NULL,
ntext141                nvarchar(max)     SPARSE  NULL,
ntext142                nvarchar(max)     SPARSE  NULL,
ntext143                nvarchar(max)     SPARSE  NULL,
ntext144                nvarchar(max)     SPARSE  NULL,
ntext145                nvarchar(max)     SPARSE  NULL,
ntext146                nvarchar(max)     SPARSE  NULL,
ntext147                nvarchar(max)     SPARSE  NULL,
ntext148                nvarchar(max)     SPARSE  NULL,
ntext149                nvarchar(max)     SPARSE  NULL,
ntext150                nvarchar(max)     SPARSE  NULL,
ntext151                nvarchar(max)     SPARSE  NULL,
ntext152                nvarchar(max)     SPARSE  NULL,
ntext153                nvarchar(max)     SPARSE  NULL,
ntext154                nvarchar(max)     SPARSE  NULL,
ntext155                nvarchar(max)     SPARSE  NULL,
ntext156                nvarchar(max)     SPARSE  NULL,
ntext157                nvarchar(max)     SPARSE  NULL,
ntext158                nvarchar(max)     SPARSE  NULL,
ntext159                nvarchar(max)     SPARSE  NULL,
ntext160                nvarchar(max)     SPARSE  NULL,
ntext161                nvarchar(max)     SPARSE  NULL,
ntext162                nvarchar(max)     SPARSE  NULL,
ntext163                nvarchar(max)     SPARSE  NULL,
ntext164                nvarchar(max)     SPARSE  NULL,
ntext165                nvarchar(max)     SPARSE  NULL,
ntext166                nvarchar(max)     SPARSE  NULL,
ntext167                nvarchar(max)     SPARSE  NULL,
ntext168                nvarchar(max)     SPARSE  NULL,
ntext169                nvarchar(max)     SPARSE  NULL,
ntext170                nvarchar(max)     SPARSE  NULL,
ntext171                nvarchar(max)     SPARSE  NULL,
ntext172                nvarchar(max)     SPARSE  NULL,
ntext173                nvarchar(max)     SPARSE  NULL,
ntext174                nvarchar(max)     SPARSE  NULL,
ntext175                nvarchar(max)     SPARSE  NULL,
ntext176                nvarchar(max)     SPARSE  NULL,
ntext177                nvarchar(max)     SPARSE  NULL,
ntext178                nvarchar(max)     SPARSE  NULL,
ntext179                nvarchar(max)     SPARSE  NULL,
ntext180                nvarchar(max)     SPARSE  NULL,
ntext181                nvarchar(max)     SPARSE  NULL,
ntext182                nvarchar(max)     SPARSE  NULL,
ntext183                nvarchar(max)     SPARSE  NULL,
ntext184                nvarchar(max)     SPARSE  NULL,
ntext185                nvarchar(max)     SPARSE  NULL,
ntext186                nvarchar(max)     SPARSE  NULL,
ntext187                nvarchar(max)     SPARSE  NULL,
ntext188                nvarchar(max)     SPARSE  NULL,
ntext189                nvarchar(max)     SPARSE  NULL,
ntext190                nvarchar(max)     SPARSE  NULL,
ntext191                nvarchar(max)     SPARSE  NULL,
ntext192                nvarchar(max)     SPARSE  NULL,
ntext193                nvarchar(max)     SPARSE  NULL,
ntext194                nvarchar(max)     SPARSE  NULL,
ntext195                nvarchar(max)     SPARSE  NULL,
ntext196                nvarchar(max)     SPARSE  NULL,
ntext197                nvarchar(max)     SPARSE  NULL,
```

```
ntext198                nvarchar(max)     SPARSE  NULL,
ntext199                nvarchar(max)     SPARSE  NULL,
ntext200                nvarchar(max)     SPARSE  NULL,
ntext201                nvarchar(max)     SPARSE  NULL,
ntext202                nvarchar(max)     SPARSE  NULL,
ntext203                nvarchar(max)     SPARSE  NULL,
ntext204                nvarchar(max)     SPARSE  NULL,
ntext205                nvarchar(max)     SPARSE  NULL,
ntext206                nvarchar(max)     SPARSE  NULL,
ntext207                nvarchar(max)     SPARSE  NULL,
ntext208                nvarchar(max)     SPARSE  NULL,
ntext209                nvarchar(max)     SPARSE  NULL,
ntext210                nvarchar(max)     SPARSE  NULL,
ntext211                nvarchar(max)     SPARSE  NULL,
ntext212                nvarchar(max)     SPARSE  NULL,
ntext213                nvarchar(max)     SPARSE  NULL,
ntext214                nvarchar(max)     SPARSE  NULL,
ntext215                nvarchar(max)     SPARSE  NULL,
ntext216                nvarchar(max)     SPARSE  NULL,
ntext217                nvarchar(max)     SPARSE  NULL,
ntext218                nvarchar(max)     SPARSE  NULL,
ntext219                nvarchar(max)     SPARSE  NULL,
ntext220                nvarchar(max)     SPARSE  NULL,
ntext221                nvarchar(max)     SPARSE  NULL,
ntext222                nvarchar(max)     SPARSE  NULL,
ntext223                nvarchar(max)     SPARSE  NULL,
ntext224                nvarchar(max)     SPARSE  NULL,
ntext225                nvarchar(max)     SPARSE  NULL,
ntext226                nvarchar(max)     SPARSE  NULL,
ntext227                nvarchar(max)     SPARSE  NULL,
ntext228                nvarchar(max)     SPARSE  NULL,
ntext229                nvarchar(max)     SPARSE  NULL,
ntext230                nvarchar(max)     SPARSE  NULL,
ntext231                nvarchar(max)     SPARSE  NULL,
ntext232                nvarchar(max)     SPARSE  NULL,
ntext233                nvarchar(max)     SPARSE  NULL,
ntext234                nvarchar(max)     SPARSE  NULL,
ntext235                nvarchar(max)     SPARSE  NULL,
ntext236                nvarchar(max)     SPARSE  NULL,
ntext237                nvarchar(max)     SPARSE  NULL,
ntext238                nvarchar(max)     SPARSE  NULL,
ntext239                nvarchar(max)     SPARSE  NULL,
ntext240                nvarchar(max)     SPARSE  NULL,
ntext241                nvarchar(max)     SPARSE  NULL,
ntext242                nvarchar(max)     SPARSE  NULL,
ntext243                nvarchar(max)     SPARSE  NULL,
ntext244                nvarchar(max)     SPARSE  NULL,
ntext245                nvarchar(max)     SPARSE  NULL,
ntext246                nvarchar(max)     SPARSE  NULL,
ntext247                nvarchar(max)     SPARSE  NULL,
ntext248                nvarchar(max)     SPARSE  NULL,
ntext249                nvarchar(max)     SPARSE  NULL,
ntext250                nvarchar(max)     SPARSE  NULL,
ntext251                nvarchar(max)     SPARSE  NULL,
ntext252                nvarchar(max)     SPARSE  NULL,
ntext253                nvarchar(max)     SPARSE  NULL,
ntext254                nvarchar(max)     SPARSE  NULL,
ntext255                nvarchar(max)     SPARSE  NULL,
ntext256                nvarchar(max)     SPARSE  NULL,
ntext257                nvarchar(max)     SPARSE  NULL,
ntext258                nvarchar(max)     SPARSE  NULL,
ntext259                nvarchar(max)     SPARSE  NULL,
ntext260                nvarchar(max)     SPARSE  NULL,
ntext261                nvarchar(max)     SPARSE  NULL,
ntext262                nvarchar(max)     SPARSE  NULL,
nvarchar1               nvarchar(255)     SPARSE  NULL,
nvarchar2               nvarchar(255)     SPARSE  NULL,
nvarchar3               nvarchar(255)     SPARSE  NULL,
nvarchar4               nvarchar(255)     SPARSE  NULL,
```

```
    nvarchar5               nvarchar(255)     SPARSE  NULL,
    nvarchar6               nvarchar(255)     SPARSE  NULL,
    nvarchar7               nvarchar(255)     SPARSE  NULL,
    nvarchar8               nvarchar(255)     SPARSE  NULL,
    nvarchar9               nvarchar(255)     SPARSE  NULL,
    nvarchar10              nvarchar(255)     SPARSE  NULL,
    nvarchar11              nvarchar(255)     SPARSE  NULL,
    nvarchar12              nvarchar(255)     SPARSE  NULL,
    nvarchar13              nvarchar(255)     SPARSE  NULL,
    nvarchar14              nvarchar(255)     SPARSE  NULL,
    nvarchar15              nvarchar(255)     SPARSE  NULL,
    nvarchar16              nvarchar(255)     SPARSE  NULL,
    nvarchar17              nvarchar(255)     SPARSE  NULL,
    nvarchar18              nvarchar(255)     SPARSE  NULL,
    nvarchar19              nvarchar(255)     SPARSE  NULL,
    nvarchar20              nvarchar(255)     SPARSE  NULL,
    nvarchar21              nvarchar(255)     SPARSE  NULL,
    nvarchar22              nvarchar(255)     SPARSE  NULL,
    nvarchar23              nvarchar(255)     SPARSE  NULL,
    nvarchar24              nvarchar(255)     SPARSE  NULL,
    nvarchar25              nvarchar(255)     SPARSE  NULL,
    nvarchar26              nvarchar(255)     SPARSE  NULL,
    nvarchar27              nvarchar(255)     SPARSE  NULL,
    nvarchar28              nvarchar(255)     SPARSE  NULL,
    nvarchar29              nvarchar(255)     SPARSE  NULL,
    nvarchar30              nvarchar(255)     SPARSE  NULL,
    nvarchar31              nvarchar(255)     SPARSE  NULL,
    nvarchar32              nvarchar(255)     SPARSE  NULL,
    nvarchar33              nvarchar(255)     SPARSE  NULL,
    nvarchar34              nvarchar(255)     SPARSE  NULL,
    nvarchar35              nvarchar(255)     SPARSE  NULL,
    nvarchar36              nvarchar(255)     SPARSE  NULL,
    nvarchar37              nvarchar(255)     SPARSE  NULL,
    nvarchar38              nvarchar(255)     SPARSE  NULL,
    nvarchar39              nvarchar(255)     SPARSE  NULL,
    nvarchar40              nvarchar(255)     SPARSE  NULL,
    nvarchar41              nvarchar(255)     SPARSE  NULL,
    nvarchar42              nvarchar(255)     SPARSE  NULL,
    nvarchar43              nvarchar(255)     SPARSE  NULL,
    nvarchar44              nvarchar(255)     SPARSE  NULL,
    nvarchar45              nvarchar(255)     SPARSE  NULL,
    nvarchar46              nvarchar(255)     SPARSE  NULL,
    nvarchar47              nvarchar(255)     SPARSE  NULL,
    nvarchar48              nvarchar(255)     SPARSE  NULL,
    nvarchar49              nvarchar(255)     SPARSE  NULL,
    nvarchar50              nvarchar(255)     SPARSE  NULL,
    nvarchar51              nvarchar(255)     SPARSE  NULL,
    nvarchar52              nvarchar(255)     SPARSE  NULL,
    nvarchar53              nvarchar(255)     SPARSE  NULL,
    nvarchar54              nvarchar(255)     SPARSE  NULL,
    nvarchar55              nvarchar(255)     SPARSE  NULL,
    nvarchar56              nvarchar(255)     SPARSE  NULL,
    nvarchar57              nvarchar(255)     SPARSE  NULL,
    nvarchar58              nvarchar(255)     SPARSE  NULL,
    nvarchar59              nvarchar(255)     SPARSE  NULL,
    nvarchar60              nvarchar(255)     SPARSE  NULL,
    nvarchar61              nvarchar(255)     SPARSE  NULL,
    nvarchar62              nvarchar(255)     SPARSE  NULL,
    nvarchar63              nvarchar(255)     SPARSE  NULL,
    nvarchar64              nvarchar(255)     SPARSE  NULL,
    nvarchar65              nvarchar(255)     SPARSE  NULL,
    nvarchar66              nvarchar(255)     SPARSE  NULL,
    nvarchar67              nvarchar(255)     SPARSE  NULL,
    nvarchar68              nvarchar(255)     SPARSE  NULL,
    nvarchar69              nvarchar(255)     SPARSE  NULL,
    nvarchar70              nvarchar(255)     SPARSE  NULL,
    nvarchar71              nvarchar(255)     SPARSE  NULL,
    nvarchar72              nvarchar(255)     SPARSE  NULL,
    nvarchar73              nvarchar(255)     SPARSE  NULL,
```

```
    nvarchar74              nvarchar(255)     SPARSE  NULL,
    nvarchar75              nvarchar(255)     SPARSE  NULL,
    nvarchar76              nvarchar(255)     SPARSE  NULL,
    nvarchar77              nvarchar(255)     SPARSE  NULL,
    nvarchar78              nvarchar(255)     SPARSE  NULL,
    nvarchar79              nvarchar(255)     SPARSE  NULL,
    nvarchar80              nvarchar(255)     SPARSE  NULL,
    nvarchar81              nvarchar(255)     SPARSE  NULL,
    nvarchar82              nvarchar(255)     SPARSE  NULL,
    nvarchar83              nvarchar(255)     SPARSE  NULL,
    nvarchar84              nvarchar(255)     SPARSE  NULL,
    nvarchar85              nvarchar(255)     SPARSE  NULL,
    nvarchar86              nvarchar(255)     SPARSE  NULL,
    nvarchar87              nvarchar(255)     SPARSE  NULL,
    nvarchar88              nvarchar(255)     SPARSE  NULL,
    nvarchar89              nvarchar(255)     SPARSE  NULL,
    nvarchar90              nvarchar(255)     SPARSE  NULL,
    nvarchar91              nvarchar(255)     SPARSE  NULL,
    nvarchar92              nvarchar(255)     SPARSE  NULL,
    nvarchar93              nvarchar(255)     SPARSE  NULL,
    nvarchar94              nvarchar(255)     SPARSE  NULL,
    nvarchar95              nvarchar(255)     SPARSE  NULL,
    nvarchar96              nvarchar(255)     SPARSE  NULL,
    nvarchar97              nvarchar(255)     SPARSE  NULL,
    nvarchar98              nvarchar(255)     SPARSE  NULL,
    nvarchar99              nvarchar(255)     SPARSE  NULL,
    nvarchar100             nvarchar(255)     SPARSE  NULL,
    nvarchar101             nvarchar(255)     SPARSE  NULL,
    nvarchar102             nvarchar(255)     SPARSE  NULL,
    nvarchar103             nvarchar(255)     SPARSE  NULL,
    nvarchar104             nvarchar(255)     SPARSE  NULL,
    nvarchar105             nvarchar(255)     SPARSE  NULL,
    nvarchar106             nvarchar(255)     SPARSE  NULL,
    nvarchar107             nvarchar(255)     SPARSE  NULL,
    nvarchar108             nvarchar(255)     SPARSE  NULL,
    nvarchar109             nvarchar(255)     SPARSE  NULL,
    nvarchar110             nvarchar(255)     SPARSE  NULL,
    nvarchar111             nvarchar(255)     SPARSE  NULL,
    nvarchar112             nvarchar(255)     SPARSE  NULL,
    nvarchar113             nvarchar(255)     SPARSE  NULL,
    nvarchar114             nvarchar(255)     SPARSE  NULL,
    nvarchar115             nvarchar(255)     SPARSE  NULL,
    nvarchar116             nvarchar(255)     SPARSE  NULL,
    nvarchar117             nvarchar(255)     SPARSE  NULL,
    nvarchar118             nvarchar(255)     SPARSE  NULL,
    nvarchar119             nvarchar(255)     SPARSE  NULL,
    nvarchar120             nvarchar(255)     SPARSE  NULL,
    nvarchar121             nvarchar(255)     SPARSE  NULL,
    nvarchar122             nvarchar(255)     SPARSE  NULL,
    nvarchar123             nvarchar(255)     SPARSE  NULL,
    nvarchar124             nvarchar(255)     SPARSE  NULL,
    nvarchar125             nvarchar(255)     SPARSE  NULL,
    nvarchar126             nvarchar(255)     SPARSE  NULL,
    nvarchar127             nvarchar(255)     SPARSE  NULL,
    nvarchar128             nvarchar(255)     SPARSE  NULL,
    nvarchar129             nvarchar(255)     SPARSE  NULL,
    nvarchar130             nvarchar(255)     SPARSE  NULL,
    nvarchar131             nvarchar(255)     SPARSE  NULL,
    nvarchar132             nvarchar(255)     SPARSE  NULL,
    nvarchar133             nvarchar(255)     SPARSE  NULL,
    nvarchar134             nvarchar(255)     SPARSE  NULL,
    nvarchar135             nvarchar(255)     SPARSE  NULL,
    nvarchar136             nvarchar(255)     SPARSE  NULL,
    nvarchar137             nvarchar(255)     SPARSE  NULL,
    nvarchar138             nvarchar(255)     SPARSE  NULL,
    nvarchar139             nvarchar(255)     SPARSE  NULL,
    nvarchar140             nvarchar(255)     SPARSE  NULL,
    nvarchar141             nvarchar(255)     SPARSE  NULL,
    nvarchar142             nvarchar(255)     SPARSE  NULL,
```

```
    nvarchar143             nvarchar(255)     SPARSE  NULL,
    nvarchar144             nvarchar(255)     SPARSE  NULL,
    nvarchar145             nvarchar(255)     SPARSE  NULL,
    nvarchar146             nvarchar(255)     SPARSE  NULL,
    nvarchar147             nvarchar(255)     SPARSE  NULL,
    nvarchar148             nvarchar(255)     SPARSE  NULL,
    nvarchar149             nvarchar(255)     SPARSE  NULL,
    nvarchar150             nvarchar(255)     SPARSE  NULL,
    nvarchar151             nvarchar(255)     SPARSE  NULL,
    nvarchar152             nvarchar(255)     SPARSE  NULL,
    nvarchar153             nvarchar(255)     SPARSE  NULL,
    nvarchar154             nvarchar(255)     SPARSE  NULL,
    nvarchar155             nvarchar(255)     SPARSE  NULL,
    nvarchar156             nvarchar(255)     SPARSE  NULL,
    nvarchar157             nvarchar(255)     SPARSE  NULL,
    nvarchar158             nvarchar(255)     SPARSE  NULL,
    nvarchar159             nvarchar(255)     SPARSE  NULL,
    nvarchar160             nvarchar(255)     SPARSE  NULL,
    nvarchar161             nvarchar(255)     SPARSE  NULL,
    nvarchar162             nvarchar(255)     SPARSE  NULL,
    nvarchar163             nvarchar(255)     SPARSE  NULL,
    nvarchar164             nvarchar(255)     SPARSE  NULL,
    nvarchar165             nvarchar(255)     SPARSE  NULL,
    nvarchar166             nvarchar(255)     SPARSE  NULL,
    nvarchar167             nvarchar(255)     SPARSE  NULL,
    nvarchar168             nvarchar(255)     SPARSE  NULL,
    nvarchar169             nvarchar(255)     SPARSE  NULL,
    nvarchar170             nvarchar(255)     SPARSE  NULL,
    nvarchar171             nvarchar(255)     SPARSE  NULL,
    nvarchar172             nvarchar(255)     SPARSE  NULL,
    nvarchar173             nvarchar(255)     SPARSE  NULL,
    nvarchar174             nvarchar(255)     SPARSE  NULL,
    nvarchar175             nvarchar(255)     SPARSE  NULL,
    nvarchar176             nvarchar(255)     SPARSE  NULL,
    nvarchar177             nvarchar(255)     SPARSE  NULL,
    nvarchar178             nvarchar(255)     SPARSE  NULL,
    nvarchar179             nvarchar(255)     SPARSE  NULL,
    nvarchar180             nvarchar(255)     SPARSE  NULL,
    nvarchar181             nvarchar(255)     SPARSE  NULL,
    nvarchar182             nvarchar(255)     SPARSE  NULL,
    nvarchar183             nvarchar(255)     SPARSE  NULL,
    nvarchar184             nvarchar(255)     SPARSE  NULL,
    nvarchar185             nvarchar(255)     SPARSE  NULL,
    nvarchar186             nvarchar(255)     SPARSE  NULL,
    nvarchar187             nvarchar(255)     SPARSE  NULL,
    nvarchar188             nvarchar(255)     SPARSE  NULL,
    nvarchar189             nvarchar(255)     SPARSE  NULL,
    nvarchar190             nvarchar(255)     SPARSE  NULL,
    nvarchar191             nvarchar(255)     SPARSE  NULL,
    nvarchar192             nvarchar(255)     SPARSE  NULL,
    nvarchar193             nvarchar(255)     SPARSE  NULL,
    nvarchar194             nvarchar(255)     SPARSE  NULL,
    nvarchar195             nvarchar(255)     SPARSE  NULL,
    nvarchar196             nvarchar(255)     SPARSE  NULL,
    nvarchar197             nvarchar(255)     SPARSE  NULL,
    nvarchar198             nvarchar(255)     SPARSE  NULL,
    nvarchar199             nvarchar(255)     SPARSE  NULL,
    nvarchar200             nvarchar(255)     SPARSE  NULL,
    nvarchar201             nvarchar(255)     SPARSE  NULL,
    nvarchar202             nvarchar(255)     SPARSE  NULL,
    nvarchar203             nvarchar(255)     SPARSE  NULL,
    nvarchar204             nvarchar(255)     SPARSE  NULL,
    nvarchar205             nvarchar(255)     SPARSE  NULL,
    nvarchar206             nvarchar(255)     SPARSE  NULL,
    nvarchar207             nvarchar(255)     SPARSE  NULL,
    nvarchar208             nvarchar(255)     SPARSE  NULL,
    nvarchar209             nvarchar(255)     SPARSE  NULL,
    nvarchar210             nvarchar(255)     SPARSE  NULL,
    nvarchar211             nvarchar(255)     SPARSE  NULL,
```

```
nvarchar212             nvarchar(255)     SPARSE  NULL,
nvarchar213             nvarchar(255)     SPARSE  NULL,
nvarchar214             nvarchar(255)     SPARSE  NULL,
nvarchar215             nvarchar(255)     SPARSE  NULL,
nvarchar216             nvarchar(255)     SPARSE  NULL,
nvarchar217             nvarchar(255)     SPARSE  NULL,
nvarchar218             nvarchar(255)     SPARSE  NULL,
nvarchar219             nvarchar(255)     SPARSE  NULL,
nvarchar220             nvarchar(255)     SPARSE  NULL,
nvarchar221             nvarchar(255)     SPARSE  NULL,
nvarchar222             nvarchar(255)     SPARSE  NULL,
nvarchar223             nvarchar(255)     SPARSE  NULL,
nvarchar224             nvarchar(255)     SPARSE  NULL,
nvarchar225             nvarchar(255)     SPARSE  NULL,
nvarchar226             nvarchar(255)     SPARSE  NULL,
nvarchar227             nvarchar(255)     SPARSE  NULL,
nvarchar228             nvarchar(255)     SPARSE  NULL,
nvarchar229             nvarchar(255)     SPARSE  NULL,
nvarchar230             nvarchar(255)     SPARSE  NULL,
nvarchar231             nvarchar(255)     SPARSE  NULL,
nvarchar232             nvarchar(255)     SPARSE  NULL,
nvarchar233             nvarchar(255)     SPARSE  NULL,
nvarchar234             nvarchar(255)     SPARSE  NULL,
nvarchar235             nvarchar(255)     SPARSE  NULL,
nvarchar236             nvarchar(255)     SPARSE  NULL,
nvarchar237             nvarchar(255)     SPARSE  NULL,
nvarchar238             nvarchar(255)     SPARSE  NULL,
nvarchar239             nvarchar(255)     SPARSE  NULL,
nvarchar240             nvarchar(255)     SPARSE  NULL,
nvarchar241             nvarchar(255)     SPARSE  NULL,
nvarchar242             nvarchar(255)     SPARSE  NULL,
nvarchar243             nvarchar(255)     SPARSE  NULL,
nvarchar244             nvarchar(255)     SPARSE  NULL,
nvarchar245             nvarchar(255)     SPARSE  NULL,
nvarchar246             nvarchar(255)     SPARSE  NULL,
nvarchar247             nvarchar(255)     SPARSE  NULL,
nvarchar248             nvarchar(255)     SPARSE  NULL,
nvarchar249             nvarchar(255)     SPARSE  NULL,
nvarchar250             nvarchar(255)     SPARSE  NULL,
nvarchar251             nvarchar(255)     SPARSE  NULL,
nvarchar252             nvarchar(255)     SPARSE  NULL,
nvarchar253             nvarchar(255)     SPARSE  NULL,
nvarchar254             nvarchar(255)     SPARSE  NULL,
nvarchar255             nvarchar(255)     SPARSE  NULL,
nvarchar256             nvarchar(255)     SPARSE  NULL,
nvarchar257             nvarchar(255)     SPARSE  NULL,
nvarchar258             nvarchar(255)     SPARSE  NULL,
nvarchar259             nvarchar(255)     SPARSE  NULL,
nvarchar260             nvarchar(255)     SPARSE  NULL,
nvarchar261             nvarchar(255)     SPARSE  NULL,
nvarchar262             nvarchar(255)     SPARSE  NULL,
sql_variant1            sql_variant       SPARSE  NULL,
sql_variant2            sql_variant       SPARSE  NULL,
sql_variant3            sql_variant       SPARSE  NULL,
sql_variant4            sql_variant       SPARSE  NULL,
sql_variant5            sql_variant       SPARSE  NULL,
sql_variant6            sql_variant       SPARSE  NULL,
sql_variant7            sql_variant       SPARSE  NULL,
sql_variant8            sql_variant       SPARSE  NULL,
sql_variant9            sql_variant       SPARSE  NULL,
sql_variant10           sql_variant       SPARSE  NULL,
sql_variant11           sql_variant       SPARSE  NULL,
sql_variant12           sql_variant       SPARSE  NULL,
sql_variant13           sql_variant       SPARSE  NULL,
sql_variant14           sql_variant       SPARSE  NULL,
sql_variant15           sql_variant       SPARSE  NULL,
sql_variant16           sql_variant       SPARSE  NULL,
sql_variant17           sql_variant       SPARSE  NULL,
sql_variant18           sql_variant       SPARSE  NULL,
```

```
sql_variant19          sql_variant       SPARSE  NULL,
sql_variant20          sql_variant       SPARSE  NULL,
sql_variant21          sql_variant       SPARSE  NULL,
sql_variant22          sql_variant       SPARSE  NULL,
sql_variant23          sql_variant       SPARSE  NULL,
sql_variant24          sql_variant       SPARSE  NULL,
sql_variant25          sql_variant       SPARSE  NULL,
sql_variant26          sql_variant       SPARSE  NULL,
sql_variant27          sql_variant       SPARSE  NULL,
sql_variant28          sql_variant       SPARSE  NULL,
sql_variant29          sql_variant       SPARSE  NULL,
sql_variant30          sql_variant       SPARSE  NULL,
sql_variant31          sql_variant       SPARSE  NULL,
sql_variant32          sql_variant       SPARSE  NULL,
sql_variant33          sql_variant       SPARSE  NULL,
sql_variant34          sql_variant       SPARSE  NULL,
sql_variant35          sql_variant       SPARSE  NULL,
sql_variant36          sql_variant       SPARSE  NULL,
sql_variant37          sql_variant       SPARSE  NULL,
sql_variant38          sql_variant       SPARSE  NULL,
sql_variant39          sql_variant       SPARSE  NULL,
sql_variant40          sql_variant       SPARSE  NULL,
sql_variant41          sql_variant       SPARSE  NULL,
sql_variant42          sql_variant       SPARSE  NULL,
sql_variant43          sql_variant       SPARSE  NULL,
sql_variant44          sql_variant       SPARSE  NULL,
sql_variant45          sql_variant       SPARSE  NULL,
sql_variant46          sql_variant       SPARSE  NULL,
sql_variant47          sql_variant       SPARSE  NULL,
sql_variant48          sql_variant       SPARSE  NULL,
sql_variant49          sql_variant       SPARSE  NULL,
sql_variant50          sql_variant       SPARSE  NULL,
sql_variant51          sql_variant       SPARSE  NULL,
sql_variant52          sql_variant       SPARSE  NULL,
sql_variant53          sql_variant       SPARSE  NULL,
sql_variant54          sql_variant       SPARSE  NULL,
sql_variant55          sql_variant       SPARSE  NULL,
sql_variant56          sql_variant       SPARSE  NULL,
sql_variant57          sql_variant       SPARSE  NULL,
sql_variant58          sql_variant       SPARSE  NULL,
sql_variant59          sql_variant       SPARSE  NULL,
sql_variant60          sql_variant       SPARSE  NULL,
sql_variant61          sql_variant       SPARSE  NULL,
sql_variant62          sql_variant       SPARSE  NULL,
sql_variant63          sql_variant       SPARSE  NULL,
sql_variant64          sql_variant       SPARSE  NULL,
sql_variant65          sql_variant       SPARSE  NULL,
sql_variant66          sql_variant       SPARSE  NULL,
sql_variant67          sql_variant       SPARSE  NULL,
sql_variant68          sql_variant       SPARSE  NULL,
sql_variant69          sql_variant       SPARSE  NULL,
sql_variant70          sql_variant       SPARSE  NULL,
sql_variant71          sql_variant       SPARSE  NULL,
sql_variant72          sql_variant       SPARSE  NULL,
sql_variant73          sql_variant       SPARSE  NULL,
sql_variant74          sql_variant       SPARSE  NULL,
sql_variant75          sql_variant       SPARSE  NULL,
sql_variant76          sql_variant       SPARSE  NULL,
sql_variant77          sql_variant       SPARSE  NULL,
sql_variant78          sql_variant       SPARSE  NULL,
sql_variant79          sql_variant       SPARSE  NULL,
sql_variant80          sql_variant       SPARSE  NULL,
sql_variant81          sql_variant       SPARSE  NULL,
sql_variant82          sql_variant       SPARSE  NULL,
sql_variant83          sql_variant       SPARSE  NULL,
sql_variant84          sql_variant       SPARSE  NULL,
sql_variant85          sql_variant       SPARSE  NULL,
sql_variant86          sql_variant       SPARSE  NULL,
sql_variant87          sql_variant       SPARSE  NULL,
```

```
sql_variant88          sql_variant       SPARSE  NULL,
sql_variant89          sql_variant       SPARSE  NULL,
sql_variant90          sql_variant       SPARSE  NULL,
sql_variant91          sql_variant       SPARSE  NULL,
sql_variant92          sql_variant       SPARSE  NULL,
sql_variant93          sql_variant       SPARSE  NULL,
sql_variant94          sql_variant       SPARSE  NULL,
sql_variant95          sql_variant       SPARSE  NULL,
sql_variant96          sql_variant       SPARSE  NULL,
sql_variant97          sql_variant       SPARSE  NULL,
sql_variant98          sql_variant       SPARSE  NULL,
sql_variant99          sql_variant       SPARSE  NULL,
sql_variant100         sql_variant       SPARSE  NULL,
sql_variant101         sql_variant       SPARSE  NULL,
sql_variant102         sql_variant       SPARSE  NULL,
sql_variant103         sql_variant       SPARSE  NULL,
sql_variant104         sql_variant       SPARSE  NULL,
sql_variant105         sql_variant       SPARSE  NULL,
sql_variant106         sql_variant       SPARSE  NULL,
sql_variant107         sql_variant       SPARSE  NULL,
sql_variant108         sql_variant       SPARSE  NULL,
sql_variant109         sql_variant       SPARSE  NULL,
sql_variant110         sql_variant       SPARSE  NULL,
sql_variant111         sql_variant       SPARSE  NULL,
sql_variant112         sql_variant       SPARSE  NULL,
sql_variant113         sql_variant       SPARSE  NULL,
sql_variant114         sql_variant       SPARSE  NULL,
sql_variant115         sql_variant       SPARSE  NULL,
sql_variant116         sql_variant       SPARSE  NULL,
sql_variant117         sql_variant       SPARSE  NULL,
sql_variant118         sql_variant       SPARSE  NULL,
sql_variant119         sql_variant       SPARSE  NULL,
sql_variant120         sql_variant       SPARSE  NULL,
sql_variant121         sql_variant       SPARSE  NULL,
sql_variant122         sql_variant       SPARSE  NULL,
sql_variant123         sql_variant       SPARSE  NULL,
sql_variant124         sql_variant       SPARSE  NULL,
sql_variant125         sql_variant       SPARSE  NULL,
sql_variant126         sql_variant       SPARSE  NULL,
sql_variant127         sql_variant       SPARSE  NULL,
sql_variant128         sql_variant       SPARSE  NULL,
sql_variant129         sql_variant       SPARSE  NULL,
sql_variant130         sql_variant       SPARSE  NULL,
sql_variant131         sql_variant       SPARSE  NULL,
sql_variant132         sql_variant       SPARSE  NULL,
sql_variant133         sql_variant       SPARSE  NULL,
sql_variant134         sql_variant       SPARSE  NULL,
sql_variant135         sql_variant       SPARSE  NULL,
sql_variant136         sql_variant       SPARSE  NULL,
sql_variant137         sql_variant       SPARSE  NULL,
sql_variant138         sql_variant       SPARSE  NULL,
sql_variant139         sql_variant       SPARSE  NULL,
sql_variant140         sql_variant       SPARSE  NULL,
sql_variant141         sql_variant       SPARSE  NULL,
sql_variant142         sql_variant       SPARSE  NULL,
sql_variant143         sql_variant       SPARSE  NULL,
sql_variant144         sql_variant       SPARSE  NULL,
sql_variant145         sql_variant       SPARSE  NULL,
sql_variant146         sql_variant       SPARSE  NULL,
sql_variant147         sql_variant       SPARSE  NULL,
sql_variant148         sql_variant       SPARSE  NULL,
sql_variant149         sql_variant       SPARSE  NULL,
sql_variant150         sql_variant       SPARSE  NULL,
sql_variant151         sql_variant       SPARSE  NULL,
sql_variant152         sql_variant       SPARSE  NULL,
sql_variant153         sql_variant       SPARSE  NULL,
sql_variant154         sql_variant       SPARSE  NULL,
sql_variant155         sql_variant       SPARSE  NULL,
sql_variant156         sql_variant       SPARSE  NULL,
```

```
sql_variant157          sql_variant       SPARSE  NULL,
sql_variant158          sql_variant       SPARSE  NULL,
sql_variant159          sql_variant       SPARSE  NULL,
sql_variant160          sql_variant       SPARSE  NULL,
sql_variant161          sql_variant       SPARSE  NULL,
sql_variant162          sql_variant       SPARSE  NULL,
sql_variant163          sql_variant       SPARSE  NULL,
sql_variant164          sql_variant       SPARSE  NULL,
sql_variant165          sql_variant       SPARSE  NULL,
sql_variant166          sql_variant       SPARSE  NULL,
sql_variant167          sql_variant       SPARSE  NULL,
sql_variant168          sql_variant       SPARSE  NULL,
sql_variant169          sql_variant       SPARSE  NULL,
sql_variant170          sql_variant       SPARSE  NULL,
sql_variant171          sql_variant       SPARSE  NULL,
sql_variant172          sql_variant       SPARSE  NULL,
sql_variant173          sql_variant       SPARSE  NULL,
sql_variant174          sql_variant       SPARSE  NULL,
sql_variant175          sql_variant       SPARSE  NULL,
sql_variant176          sql_variant       SPARSE  NULL,
sql_variant177          sql_variant       SPARSE  NULL,
sql_variant178          sql_variant       SPARSE  NULL,
sql_variant179          sql_variant       SPARSE  NULL,
sql_variant180          sql_variant       SPARSE  NULL,
sql_variant181          sql_variant       SPARSE  NULL,
sql_variant182          sql_variant       SPARSE  NULL,
sql_variant183          sql_variant       SPARSE  NULL,
sql_variant184          sql_variant       SPARSE  NULL,
sql_variant185          sql_variant       SPARSE  NULL,
sql_variant186          sql_variant       SPARSE  NULL,
sql_variant187          sql_variant       SPARSE  NULL,
sql_variant188          sql_variant       SPARSE  NULL,
sql_variant189          sql_variant       SPARSE  NULL,
sql_variant190          sql_variant       SPARSE  NULL,
sql_variant191          sql_variant       SPARSE  NULL,
sql_variant192          sql_variant       SPARSE  NULL,
sql_variant193          sql_variant       SPARSE  NULL,
sql_variant194          sql_variant       SPARSE  NULL,
sql_variant195          sql_variant       SPARSE  NULL,
sql_variant196          sql_variant       SPARSE  NULL,
sql_variant197          sql_variant       SPARSE  NULL,
sql_variant198          sql_variant       SPARSE  NULL,
sql_variant199          sql_variant       SPARSE  NULL,
sql_variant200          sql_variant       SPARSE  NULL,
sql_variant201          sql_variant       SPARSE  NULL,
sql_variant202          sql_variant       SPARSE  NULL,
sql_variant203          sql_variant       SPARSE  NULL,
sql_variant204          sql_variant       SPARSE  NULL,
sql_variant205          sql_variant       SPARSE  NULL,
sql_variant206          sql_variant       SPARSE  NULL,
sql_variant207          sql_variant       SPARSE  NULL,
sql_variant208          sql_variant       SPARSE  NULL,
sql_variant209          sql_variant       SPARSE  NULL,
sql_variant210          sql_variant       SPARSE  NULL,
sql_variant211          sql_variant       SPARSE  NULL,
sql_variant212          sql_variant       SPARSE  NULL,
sql_variant213          sql_variant       SPARSE  NULL,
sql_variant214          sql_variant       SPARSE  NULL,
sql_variant215          sql_variant       SPARSE  NULL,
sql_variant216          sql_variant       SPARSE  NULL,
sql_variant217          sql_variant       SPARSE  NULL,
sql_variant218          sql_variant       SPARSE  NULL,
sql_variant219          sql_variant       SPARSE  NULL,
sql_variant220          sql_variant       SPARSE  NULL,
sql_variant221          sql_variant       SPARSE  NULL,
sql_variant222          sql_variant       SPARSE  NULL,
sql_variant223          sql_variant       SPARSE  NULL,
sql_variant224          sql_variant       SPARSE  NULL,
sql_variant225          sql_variant       SPARSE  NULL,
```

```
sql_variant226          sql_variant       SPARSE  NULL,
sql_variant227          sql_variant       SPARSE  NULL,
sql_variant228          sql_variant       SPARSE  NULL,
sql_variant229          sql_variant       SPARSE  NULL,
sql_variant230          sql_variant       SPARSE  NULL,
sql_variant231          sql_variant       SPARSE  NULL,
sql_variant232          sql_variant       SPARSE  NULL,
sql_variant233          sql_variant       SPARSE  NULL,
sql_variant234          sql_variant       SPARSE  NULL,
sql_variant235          sql_variant       SPARSE  NULL,
sql_variant236          sql_variant       SPARSE  NULL,
sql_variant237          sql_variant       SPARSE  NULL,
sql_variant238          sql_variant       SPARSE  NULL,
sql_variant239          sql_variant       SPARSE  NULL,
sql_variant240          sql_variant       SPARSE  NULL,
sql_variant241          sql_variant       SPARSE  NULL,
sql_variant242          sql_variant       SPARSE  NULL,
sql_variant243          sql_variant       SPARSE  NULL,
sql_variant244          sql_variant       SPARSE  NULL,
sql_variant245          sql_variant       SPARSE  NULL,
sql_variant246          sql_variant       SPARSE  NULL,
sql_variant247          sql_variant       SPARSE  NULL,
sql_variant248          sql_variant       SPARSE  NULL,
sql_variant249          sql_variant       SPARSE  NULL,
sql_variant250          sql_variant       SPARSE  NULL,
sql_variant251          sql_variant       SPARSE  NULL,
sql_variant252          sql_variant       SPARSE  NULL,
sql_variant253          sql_variant       SPARSE  NULL,
sql_variant254          sql_variant       SPARSE  NULL,
sql_variant255          sql_variant       SPARSE  NULL,
sql_variant256          sql_variant       SPARSE  NULL,
sql_variant257          sql_variant       SPARSE  NULL,
sql_variant258          sql_variant       SPARSE  NULL,
sql_variant259          sql_variant       SPARSE  NULL,
sql_variant260          sql_variant       SPARSE  NULL,
sql_variant261          sql_variant       SPARSE  NULL,
sql_variant262          sql_variant       SPARSE  NULL,
uniqueidentifier1       uniqueidentifier  SPARSE  NULL,
uniqueidentifier2       uniqueidentifier  SPARSE  NULL,
uniqueidentifier3       uniqueidentifier  SPARSE  NULL,
uniqueidentifier4       uniqueidentifier  SPARSE  NULL,
uniqueidentifier5       uniqueidentifier  SPARSE  NULL,
uniqueidentifier6       uniqueidentifier  SPARSE  NULL,
uniqueidentifier7       uniqueidentifier  SPARSE  NULL,
uniqueidentifier8       uniqueidentifier  SPARSE  NULL,
uniqueidentifier9       uniqueidentifier  SPARSE  NULL,
uniqueidentifier10      uniqueidentifier  SPARSE  NULL,
uniqueidentifier11      uniqueidentifier  SPARSE  NULL,
uniqueidentifier12      uniqueidentifier  SPARSE  NULL,
uniqueidentifier13      uniqueidentifier  SPARSE  NULL,
uniqueidentifier14      uniqueidentifier  SPARSE  NULL,
uniqueidentifier15      uniqueidentifier  SPARSE  NULL,
uniqueidentifier16      uniqueidentifier  SPARSE  NULL,
uniqueidentifier17      uniqueidentifier  SPARSE  NULL,
uniqueidentifier18      uniqueidentifier  SPARSE  NULL,
uniqueidentifier19      uniqueidentifier  SPARSE  NULL,
uniqueidentifier20      uniqueidentifier  SPARSE  NULL,
uniqueidentifier21      uniqueidentifier  SPARSE  NULL,
uniqueidentifier22      uniqueidentifier  SPARSE  NULL,
uniqueidentifier23      uniqueidentifier  SPARSE  NULL,
uniqueidentifier24      uniqueidentifier  SPARSE  NULL,
uniqueidentifier25      uniqueidentifier  SPARSE  NULL,
uniqueidentifier26      uniqueidentifier  SPARSE  NULL,
uniqueidentifier27      uniqueidentifier  SPARSE  NULL,
uniqueidentifier28      uniqueidentifier  SPARSE  NULL,
uniqueidentifier29      uniqueidentifier  SPARSE  NULL,
uniqueidentifier30      uniqueidentifier  SPARSE  NULL,
uniqueidentifier31      uniqueidentifier  SPARSE  NULL,
uniqueidentifier32      uniqueidentifier  SPARSE  NULL,
```

```
uniqueidentifier33      uniqueidentifier  SPARSE  NULL,
uniqueidentifier34      uniqueidentifier  SPARSE  NULL,
uniqueidentifier35      uniqueidentifier  SPARSE  NULL,
uniqueidentifier36      uniqueidentifier  SPARSE  NULL,
uniqueidentifier37      uniqueidentifier  SPARSE  NULL,
uniqueidentifier38      uniqueidentifier  SPARSE  NULL,
uniqueidentifier39      uniqueidentifier  SPARSE  NULL,
uniqueidentifier40      uniqueidentifier  SPARSE  NULL,
uniqueidentifier41      uniqueidentifier  SPARSE  NULL,
uniqueidentifier42      uniqueidentifier  SPARSE  NULL,
uniqueidentifier43      uniqueidentifier  SPARSE  NULL,
uniqueidentifier44      uniqueidentifier  SPARSE  NULL,
uniqueidentifier45      uniqueidentifier  SPARSE  NULL,
uniqueidentifier46      uniqueidentifier  SPARSE  NULL,
uniqueidentifier47      uniqueidentifier  SPARSE  NULL,
uniqueidentifier48      uniqueidentifier  SPARSE  NULL,
uniqueidentifier49      uniqueidentifier  SPARSE  NULL,
uniqueidentifier50      uniqueidentifier  SPARSE  NULL,
uniqueidentifier51      uniqueidentifier  SPARSE  NULL,
uniqueidentifier52      uniqueidentifier  SPARSE  NULL,
uniqueidentifier53      uniqueidentifier  SPARSE  NULL,
uniqueidentifier54      uniqueidentifier  SPARSE  NULL,
uniqueidentifier55      uniqueidentifier  SPARSE  NULL,
uniqueidentifier56      uniqueidentifier  SPARSE  NULL,
uniqueidentifier57      uniqueidentifier  SPARSE  NULL,
uniqueidentifier58      uniqueidentifier  SPARSE  NULL,
uniqueidentifier59      uniqueidentifier  SPARSE  NULL,
uniqueidentifier60      uniqueidentifier  SPARSE  NULL,
uniqueidentifier61      uniqueidentifier  SPARSE  NULL,
uniqueidentifier62      uniqueidentifier  SPARSE  NULL,
uniqueidentifier63      uniqueidentifier  SPARSE  NULL,
uniqueidentifier64      uniqueidentifier  SPARSE  NULL,
uniqueidentifier65      uniqueidentifier  SPARSE  NULL,
uniqueidentifier66      uniqueidentifier  SPARSE  NULL,
uniqueidentifier67      uniqueidentifier  SPARSE  NULL,
uniqueidentifier68      uniqueidentifier  SPARSE  NULL,
uniqueidentifier69      uniqueidentifier  SPARSE  NULL,
uniqueidentifier70      uniqueidentifier  SPARSE  NULL,
uniqueidentifier71      uniqueidentifier  SPARSE  NULL,
uniqueidentifier72      uniqueidentifier  SPARSE  NULL,
uniqueidentifier73      uniqueidentifier  SPARSE  NULL,
uniqueidentifier74      uniqueidentifier  SPARSE  NULL,
uniqueidentifier75      uniqueidentifier  SPARSE  NULL,
uniqueidentifier76      uniqueidentifier  SPARSE  NULL,
uniqueidentifier77      uniqueidentifier  SPARSE  NULL,
uniqueidentifier78      uniqueidentifier  SPARSE  NULL,
uniqueidentifier79      uniqueidentifier  SPARSE  NULL,
uniqueidentifier80      uniqueidentifier  SPARSE  NULL,
uniqueidentifier81      uniqueidentifier  SPARSE  NULL,
uniqueidentifier82      uniqueidentifier  SPARSE  NULL,
uniqueidentifier83      uniqueidentifier  SPARSE  NULL,
uniqueidentifier84      uniqueidentifier  SPARSE  NULL,
uniqueidentifier85      uniqueidentifier  SPARSE  NULL,
uniqueidentifier86      uniqueidentifier  SPARSE  NULL,
uniqueidentifier87      uniqueidentifier  SPARSE  NULL,
uniqueidentifier88      uniqueidentifier  SPARSE  NULL,
uniqueidentifier89      uniqueidentifier  SPARSE  NULL,
uniqueidentifier90      uniqueidentifier  SPARSE  NULL,
uniqueidentifier91      uniqueidentifier  SPARSE  NULL,
uniqueidentifier92      uniqueidentifier  SPARSE  NULL,
uniqueidentifier93      uniqueidentifier  SPARSE  NULL,
uniqueidentifier94      uniqueidentifier  SPARSE  NULL,
uniqueidentifier95      uniqueidentifier  SPARSE  NULL,
uniqueidentifier96      uniqueidentifier  SPARSE  NULL,
uniqueidentifier97      uniqueidentifier  SPARSE  NULL,
uniqueidentifier98      uniqueidentifier  SPARSE  NULL,
uniqueidentifier99      uniqueidentifier  SPARSE  NULL,
uniqueidentifier100     uniqueidentifier  SPARSE  NULL,
uniqueidentifier101     uniqueidentifier  SPARSE  NULL,
```

```
uniqueidentifier102     uniqueidentifier  SPARSE  NULL,
uniqueidentifier103     uniqueidentifier  SPARSE  NULL,
uniqueidentifier104     uniqueidentifier  SPARSE  NULL,
uniqueidentifier105     uniqueidentifier  SPARSE  NULL,
uniqueidentifier106     uniqueidentifier  SPARSE  NULL,
uniqueidentifier107     uniqueidentifier  SPARSE  NULL,
uniqueidentifier108     uniqueidentifier  SPARSE  NULL,
uniqueidentifier109     uniqueidentifier  SPARSE  NULL,
uniqueidentifier110     uniqueidentifier  SPARSE  NULL,
uniqueidentifier111     uniqueidentifier  SPARSE  NULL,
uniqueidentifier112     uniqueidentifier  SPARSE  NULL,
uniqueidentifier113     uniqueidentifier  SPARSE  NULL,
uniqueidentifier114     uniqueidentifier  SPARSE  NULL,
uniqueidentifier115     uniqueidentifier  SPARSE  NULL,
uniqueidentifier116     uniqueidentifier  SPARSE  NULL,
uniqueidentifier117     uniqueidentifier  SPARSE  NULL,
uniqueidentifier118     uniqueidentifier  SPARSE  NULL,
uniqueidentifier119     uniqueidentifier  SPARSE  NULL,
uniqueidentifier120     uniqueidentifier  SPARSE  NULL,
uniqueidentifier121     uniqueidentifier  SPARSE  NULL,
uniqueidentifier122     uniqueidentifier  SPARSE  NULL,
uniqueidentifier123     uniqueidentifier  SPARSE  NULL,
uniqueidentifier124     uniqueidentifier  SPARSE  NULL,
uniqueidentifier125     uniqueidentifier  SPARSE  NULL,
uniqueidentifier126     uniqueidentifier  SPARSE  NULL,
uniqueidentifier127     uniqueidentifier  SPARSE  NULL,
uniqueidentifier128     uniqueidentifier  SPARSE  NULL,
uniqueidentifier129     uniqueidentifier  SPARSE  NULL,
uniqueidentifier130     uniqueidentifier  SPARSE  NULL,
uniqueidentifier131     uniqueidentifier  SPARSE  NULL,
uniqueidentifier132     uniqueidentifier  SPARSE  NULL,
uniqueidentifier133     uniqueidentifier  SPARSE  NULL,
uniqueidentifier134     uniqueidentifier  SPARSE  NULL,
uniqueidentifier135     uniqueidentifier  SPARSE  NULL,
uniqueidentifier136     uniqueidentifier  SPARSE  NULL,
uniqueidentifier137     uniqueidentifier  SPARSE  NULL,
uniqueidentifier138     uniqueidentifier  SPARSE  NULL,
uniqueidentifier139     uniqueidentifier  SPARSE  NULL,
uniqueidentifier140     uniqueidentifier  SPARSE  NULL,
uniqueidentifier141     uniqueidentifier  SPARSE  NULL,
uniqueidentifier142     uniqueidentifier  SPARSE  NULL,
uniqueidentifier143     uniqueidentifier  SPARSE  NULL,
uniqueidentifier144     uniqueidentifier  SPARSE  NULL,
uniqueidentifier145     uniqueidentifier  SPARSE  NULL,
uniqueidentifier146     uniqueidentifier  SPARSE  NULL,
uniqueidentifier147     uniqueidentifier  SPARSE  NULL,
uniqueidentifier148     uniqueidentifier  SPARSE  NULL,
uniqueidentifier149     uniqueidentifier  SPARSE  NULL,
uniqueidentifier150     uniqueidentifier  SPARSE  NULL,
uniqueidentifier151     uniqueidentifier  SPARSE  NULL,
uniqueidentifier152     uniqueidentifier  SPARSE  NULL,
uniqueidentifier153     uniqueidentifier  SPARSE  NULL,
uniqueidentifier154     uniqueidentifier  SPARSE  NULL,
uniqueidentifier155     uniqueidentifier  SPARSE  NULL,
uniqueidentifier156     uniqueidentifier  SPARSE  NULL,
uniqueidentifier157     uniqueidentifier  SPARSE  NULL,
uniqueidentifier158     uniqueidentifier  SPARSE  NULL,
uniqueidentifier159     uniqueidentifier  SPARSE  NULL,
uniqueidentifier160     uniqueidentifier  SPARSE  NULL,
uniqueidentifier161     uniqueidentifier  SPARSE  NULL,
uniqueidentifier162     uniqueidentifier  SPARSE  NULL,
uniqueidentifier163     uniqueidentifier  SPARSE  NULL,
uniqueidentifier164     uniqueidentifier  SPARSE  NULL,
uniqueidentifier165     uniqueidentifier  SPARSE  NULL,
uniqueidentifier166     uniqueidentifier  SPARSE  NULL,
uniqueidentifier167     uniqueidentifier  SPARSE  NULL,
uniqueidentifier168     uniqueidentifier  SPARSE  NULL,
uniqueidentifier169     uniqueidentifier  SPARSE  NULL,
uniqueidentifier170     uniqueidentifier  SPARSE  NULL,
```

```
    uniqueidentifier171     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier172     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier173     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier174     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier175     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier176     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier177     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier178     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier179     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier180     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier181     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier182     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier183     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier184     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier185     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier186     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier187     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier188     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier189     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier190     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier191     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier192     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier193     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier194     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier195     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier196     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier197     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier198     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier199     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier200     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier201     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier202     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier203     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier204     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier205     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier206     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier207     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier208     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier209     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier210     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier211     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier212     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier213     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier214     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier215     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier216     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier217     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier218     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier219     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier220     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier221     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier222     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier223     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier224     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier225     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier226     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier227     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier228     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier229     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier230     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier231     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier232     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier233     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier234     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier235     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier236     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier237     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier238     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier239     uniqueidentifier  SPARSE  NULL,
```

```
    uniqueidentifier240     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier241     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier242     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier243     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier244     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier245     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier246     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier247     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier248     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier249     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier250     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier251     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier252     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier253     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier254     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier255     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier256     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier257     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier258     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier259     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier260     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier261     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier262     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier263     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier264     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier265     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier266     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier267     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier268     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier269     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier270     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier271     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier272     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier273     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier274     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier275     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier276     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier277     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier278     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier279     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier280     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier281     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier282     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier283     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier284     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier285     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier286     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier287     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier288     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier289     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier290     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier291     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier292     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier293     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier294     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier295     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier296     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier297     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier298     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier299     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier300     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier301     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier302     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier303     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier304     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier305     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier306     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier307     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier308     uniqueidentifier  SPARSE  NULL,
```

```
    uniqueidentifier309     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier310     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier311     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier312     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier313     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier314     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier315     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier316     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier317     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier318     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier319     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier320     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier321     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier322     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier323     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier324     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier325     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier326     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier327     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier328     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier329     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier330     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier331     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier332     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier333     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier334     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier335     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier336     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier337     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier338     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier339     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier340     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier341     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier342     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier343     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier344     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier345     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier346     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier347     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier348     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier349     uniqueidentifier  SPARSE  NULL,
    uniqueidentifier350     uniqueidentifier  SPARSE  NULL,
    geography1              varbinary(8000)   SPARSE  NULL,
    geography2              varbinary(8000)   SPARSE  NULL,
    tp_ColumnSet            xml COLUMN_SET FOR ALL_SPARSE_COLUMNS NULL
);
```

tp_Id: The identifier for the list item, uniquely identifying it within the **AllUserData** table.

tp_ListId: The **list identifier** of the list (1) or document library containing the list item.

tp_SiteId: The **site collection identifier** 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 used for internal conflict detection that is incremented any time a change is made to the list item. Because of 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** 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 (1) (a list (1) with a **List Base Type** (section 2.2.1.2.11) of three). For list items in all other types of lists (1), this parameter MUST be NULL.

tp_ThreadIndex: A binary structure specifying the list item's position within a legacy discussion board list (1) (a list (1) with a **List Base Type** of three). For list items in all other types of lists (1), this parameter MUST be NULL.

tp_HasAttachment: A bit set to one if the list item has an attachment associated with it; otherwise, it is set to zero.

tp_ModerationStatus: A **Moderation Status** value (section 2.2.1.2.13) indicating the current moderation approval status of the list item.

tp_IsCurrent: A bit set to one if this is a current version of this list item; otherwise, it is set to zero.

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 (1).

tp_InstanceID: If this list item is associated with a particular instance of a recurring meeting, this is the integer identifier 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 one 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 zero.

tp_AuditFlags: An **Audit Flags** value (section 2.2.2.1) 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 **document stream** for list items that have an associated document 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** value (section 2.2.1.1.16) 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** 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 (2)**) 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.

tContentTypeId is defined in section 2.2.1.1.1.

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 a historical version of the list item. This value MUST be one if this row contains a current version. Otherwise, it MUST be zero.

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 a historical version of the list item. This MUST be zero 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.

tp_AppAuthor: The **app principal** identifier of the app principal who created the list item.

tp_AppEditor: The app principal identifier of the app principal who last edited the list item.

The next nine 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.

bit#: Columns for application-defined fields that hold values of type **bit**. The 1000 columns are named **bit1** to **bit1000**. 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 550 columns are named **datetime1** to **datetime550**. 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 550 columns are named **float1** to **float550**. 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 750 columns are named **int1** to **int750**. 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 262 columns are named **ntext1** to **ntext262**. If the column does not contain data, this value MUST be NULL.

nvarchar#: Columns for application-defined fields that hold values of type **nvarchar**. The 262 columns are named **nvarchar1** to **nvarchar262**. 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 262 columns are named **sql_variant1** to **sql_variant262**. If the column does not contain data, this value MUST be NULL.

uniqueidentifier#: Columns for application-defined fields that hold values of type **uniqueidentifier**. The 350 columns are named **uniqueidentifier1** to **uniqueidentifier350**. If the column does not contain data, this value MUST be NULL.

geography#: Columns for application-defined fields that hold values of type **varbinary(8000)**. The 2 columns are named **geography1** and **geography2**. If the column does not contain data, this value MUST be NULL.

tp_ColumnSet: An **XML** representation of the non-NULL values from the columns **bit#**, **datetime#**, **float#**, **int#**, **ntext#**, **nvarchar#**, **sql_variant#**, **uniqueidentifier#**, and **geography#**.

2.2.6.3 Docs View

The **Docs View** reflects all rows in the table **AllDocs** (section 2.2.6.1) 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** for the T-SQL syntax description of the structure of this view.

2.2.6.4 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 2.2.1.1.9) of the site collection containing the principals and site groups.

GroupId: The group identifier of the site group that the **principal (1)** specified by **MemberId** is a member of.

MemberId: The User Identifier (section 2.2.1.1.13) of a principal that is a member of the site group specified by **GroupId**.

2.2.6.5 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 one, 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 zero, and the **UserID**, **UserSID**, **UserName**, **UserLogin**, **UserEmail**, **UserNotes**, **UserSiteAdmin** and **UserDomainGroup** are set to NULL.

The **Sec_SiteGroupsView** 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 2.2.1.1.9) 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 2.2.1.1.13) or **Site Group Identifier** (section 2.2.1.1.10) 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 one. When the value in the **Owner field** is a **Site Group Identifier**, the **OwnerIsUser** flag MUST be set to zero. 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 one and the **GroupID**, **GroupName**, **GroupDescription**, **GroupSiteID**, **GroupOwner** and **GroupOwnerIsUser** fields MUST be populated with information from the site group owning this site group, while the **UserID**, **UserSID**, **UserName**, **UserLogin**, **UserEmail**, **UserNotes**, **UserSiteAdmin** and **UserDomainGroup** fields MUST be NULL.

When the owner is a user or domain group, the **OwnerGlobal** flag MUST be set to zero and the **UserID**, **UserSID**, **UserName**, **UserLogin**, **UserEmail**, **UserNotes**, **UserSiteAdmin** and **UserDomainGroup** fields MUST be populated with information from the user or domain group owning this site group, while the **GroupID**, **GroupName**, **GroupDescription**, **GroupSiteID**, **GroupOwner** and **GroupOwnerIsUser** fields MUST be NULL.

Type: This value MUST be zero.

Global: This value MUST be one.

SiteID: The **Site Collection Identifier** 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** 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 e-mail 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 one. If the user or domain group owner of the site group is not a site collection administrator, the **UserSiteAdmin** flag MUST be set to zero. 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 one. When the site group owner is a user or domain group, the flag MUST be set to zero. This flag MUST be NULL when the site group owner is a site group.

UserFlags: Contains the **User Information Flags** value (section 2.2.2.12) 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** 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** 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** 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 one. This flag MUST be set to zero 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 e-mail distribution list address for the site group. This value MUST be NULL if the **group (2)** has no e-mail distribution list address.

DLErrorMessage: Contains the most recent error message returned by an asynchronous e-mail distribution list operation, if any.

DLFlags: Contains a bit field of status flags for the e-mail 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 e-mail distribution list associated with this site group currently has a pending asynchronous operation.
0x00000002	Dirty - If this bit is set, the e-mail distribution list associated with this group (2) is **dirty**. The e-mail 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 e-mail 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 e-mail 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 identifier of the currently pending asynchronous e-mail distribution list operation, or zero if there is no pending operation.

DLArchives: An array of bytes containing a list of pairs of **List Identifiers** (section 2.2.1.1.5) defining additional lists, which are recipients of e-mail sent to the e-mail 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 2.2.1.1.2) and the **List Identifier**, and with semicolons following each pair.

RequestEmail: The e-mail 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 e-mail 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.

2.2.6.6 AllSites Table

The **AllSites** table stores information about **site collections**. The table is defined using **T-SQL** syntax, as follows.

```
TABLE AllSites(
      FullUrl                        nvarchar(260) DEFAULT N'',
      Id                             uniqueidentifier,
      RootWebId                      uniqueidentifier,
      ClientTag                      smallint, DEFAULT 0 NOT NULL,
      NextUserOrGroupId              int,
      NextAppPrincipalId             int NOT NULL DEFAULT 1,
      OwnerID                        int,
      SecondaryContactID             int,
      Subscribed                     bit,
      TimeCreated                    datetime,
      Deleted                        bit,
      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,
      DenyPermMask                   tPermMask DEFAULT 0 NOT NULL,
      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(512),

      HashKey                        binary(16),
      DomainGroupMapVersion          bigint,
      DomainGroupMapCacheVersion     bigint        DEFAULT ((-1)),
      DomainGroupMapCache            varbinary(max),
      HostHeader                     nvarchar(128),
      SiteUrls                       varbinary(max),
      SubscriptionId                 varbinary(16),
      RbsCollectionId                int           DEFAULT 0 NOT NULL,
      LastSMRequest                  datetime      DEFAULT NULL,
      PlatformVersion              nvarchar(64) DEFAULT NULL,
      UpgradeFlags                   int  DEFAULT 1 NOT NULL,

      SourceSiteId                   uniqueidentifier SPARSE NULL,
      ExpirationDate                 datetime      SPARSE NULL,
      EvalSiteId                     uniqueidentifier SPARSE NULL,
      UpgradeReminderDate            datetime,
      SubscriptionName               nvarchar(48) DEFAULT NULL,
      AppSiteDomainId                varchar(6) DEFAULT NULL,
      NextAppWebDomainId             binary(4) DEFAULT NULL

);
```

FullUrl: The absolute **URL** of the site collection.

Id: The **Site Collection Identifier** of the site collection. Used only during upgrade.

RootWebId: The **site identifier** of the **site (2)** configured as the root site in the site collection.

ClientTag: A number that represents the state of the application files of the site (2).

NextUserOrGroupId: An integer value that is incremented when a new user or **permission level** is added to the site collection. Indicates the value of the next user or **Site Group Identifier** (section 2.2.1.1.10) to be used.

NextAppPrincipalId: An integer value that is incremented when a new **app principal** is added to the site collection. Indicates the value of the next app principal integer identifier in the site collection to be used.

OwnerID: The **user identifier** 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.

Deleted: A bit set to 1 indicates that a delete operation has been initiated on the site collection and it has been scheduled for deletion.

UsersCount: The number of users in the site collection.

BWUsed: The number of sites in the site collection actively used. Serves as an implementation-specific, usage-reporting feature.

DiskUsed: The size, in bytes, of disk space used to store content in the site collection.

SecondStageDiskUsed: The size, in bytes, of disk space used to store the second-stage trash bin items for the site collection.

QuotaTemplateID: The 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 that specifies the current resource usage warning setting. If the current resource usage exceeds it, a warning notification will be sent.

ResourceUsageMaximum: A float that specifies the maximum 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 that is incremented when changes are made to the site collection's permissions.

DenyPermMask: A **WSS Rights Mask** (section 2.2.2.15) specifying the rights to deny to the current user.

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 the **CertificationDate** column.)

PortalURL: The URL of an external **Web site** designated as the portal of the site (1). This is an implementation-specific navigation links **feature**.

PortalName: The name of the external Web site 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** (section 2.2.2.1) value specifying the operations to be audited on the specified object that are inherited.

UserInfoListId: The **List Identifier** (section 2.2.1.1.5) 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.

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 permission levels 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**.

SiteUrls: A string that contains a semi-colon separated list of encoded URLs of the site collection. Each URL is encoded as a comma separated set of properties:

<UrlZone:1 char encoding **tinyint** as ascii digit><comma><Scheme: string, its length is less or equal than 5><comma><HostHeader><comma><AppSiteDomainId>

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.

PlatformVersion: The string that represents the current version of the site collection from the platform compatibility perspective.

UpgradeFlags: The site collection upgrade flags (see section 2.2.2.10) for this site collection.

SourceSiteId: The **site collection identifier** of the site collection that was used to create the **upgrade evaluation site collection**. A non-NULL value for this column implies that the site collection is an upgrade evaluation site collection.

ExpirationDate: The date and time value in UTC format, after which the upgrade evaluation site collection will be automatically deleted.

EvalSiteId: The site collection identifier of the upgrade evaluation site collection that was created using the site collection.

UpgradeReminderDate: The upgrade reminder date of the site collection.

SubscriptionName: The name corresponding to the **site subscription identifier** of the implementation-specific subscription for the requested site collection.

AppSiteDomainId: The **app site domain identifier**.

NextAppWebDomainId: The next **app web domain identifier** that will be used for the next **subsite** created within the site collection.

2.2.6.7 Sites View

The **Sites View** reflects all rows in the table **AllSites** (section 2.2.6.6) that are not scheduled for deletion (that is, all table rows where Deleted is equal to 0). **Sites View** records information about **site collections**.

See **AllSites** for the **T-SQL** syntax description of the structure of this view.

2.2.6.8 UserData View

The **UserData View** reflects all list items in the **AllUserData** table (section 2.2.6.2) 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** for the T-SQL syntax description of the structure of this view. Columns defined with the keyword SPARSE do not appear in this view.

2.2.6.9 UserDataVersioned View

The **UserDataVersioned View** reflects all list items in the **AllUserData** table (section 2.2.6.2) that are not deleted (that is, all table rows where **tp_DeleteTransactionId** is 0x).

See **AllUserData** for the T-SQL syntax description of the structure of this view.

2.2.6.10 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              tinyint             NULL,
tp_CalendarViewOptions          tinyint             NULL,
tp_WorkDays                     smallint            NULL,
tp_WorkDayStartHour             smallint            NULL,
tp_WorkDayEndHour               smallint            NULL,
tp_Mobile                       nvarchar(127)       NULL,
tp_MUILanguages                 varchar(64)         NULL,
tp_ContentLanguages             varchar(64)         NULL,
tp_Flags                        int                 NOT NULL DEFAULT (0),
);
```

tp_SiteID: The Site Collection Identifier (section 2.2.1.1.9) of the site collection associated with the **principal (1)**.

tp_Id: The User Identifier (section 2.2.1.1.13) of the principal, which MUST be unique within both this table and the group identifiers for permission levels. Certain User Identifiers are predefined, such as the system account (1073741823), the site collection owner (1), and the secondary site collection contact (2). 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** 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 e-mail 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 2.2.3.10) value specifying the permission level membership of the principal. This can be NULL, indicating that the principal is not a member of a permission level.

tp_ExternalToken: An **External Group Token** (section 2.2.3.2) 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** 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** 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 workweek 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_MUILanguages: A string that contains the distinct culture(s) for the user's preferred display language(s), separated by semicolon. This string can be NULL.

tp_ContentLanguages: A string that contains the distinct culture(s) for the user's preferred content language(s), separated by semicolon. This string can be NULL.

tp_Flags: A 4-byte integer bit mask determining the user's options. For more details, see section 2.2.2.12.

2.2.6.11 WebMembers

The **WebMembers** table stores information used for determining the membership of users in sites. This table is defined using **T-SQL** syntax, as follows.

```
TABLE WebMembers (
      SiteId                        uniqueidentifier NOT NULL,
      WebId                         uniqueidentifier NOT NULL,
      UserId                        int NOT NULL
);
```

SiteId: The **site collection identifier** of the **site collection** that contains this **site (2)**.

WebId: The **site identifier** of the site (2) associated with the user.

UserId: The user identifier of the user associated with the site (2).

2.2.6.12 Versions

The **Versions Table** stores information used for object versioning during upgrade operations. This 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.

2.2.7 XML Structures

No common XML Structures are defined in this protocol.

2.2.7.1 Namespaces

This specification does not define any common XML schema namespaces.

2.2.7.2 Simple Types

2.2.7.2.1 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>
```

2.2.7.2.2 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

2.2.7.2.3 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.
sum	The sum of the nodes that match and whose values can be interpreted as a floating point numbers.

2.2.7.2.4 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	Description
Recurrence	Specifies the recurrence field.
EventType	Specifies the field that contains the type of the event.
UID	Specifies the field that contains the identifier for the event.
RecurrenceId	Specifies the field that contains the recurrence identifier for the event.
EventCancel	Specifies the field that identifies whether the event has been canceled.
StartDate	Specifies the field that contains the start date for the event.
EndDate	Specifies the field that contains the end date for the event.
RecurData	Specifies the field that contains recurrence information for the event.
Duration	Specifies the field that contains the duration of the event.
TimeZone	Specifies a reference to the effective time zone for the event.
XMLTZone	Specifies a reference to XML that describes the effective time zone for the event.
CPLink	Specifies a reference to another web site.
LinkURL	Specifies the field that contains a link to another web site.
MasterSeriesItemID	Specifies the field that contains the ID of the item that defines the recurring event.
AllDayEvent	Specifies the field that identifies whether the event is an all-day event.

2.2.7.2.5 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.

Value	Description
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.

2.2.7.2.6 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 reader MUST accept any value. The writer SHOULD write this attribute as one of the valid values for the **ime-mode** attribute (as described in [CSS3UI]) to be applied to an <input> tag (as specified in [HTML], section 17.4).

2.2.7.2.7 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>
```

2.2.7.2.8 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.

2.2.7.2.9 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.

2.2.7.2.10 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>
```

2.2.7.2.11 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>
```

2.2.7.2.12 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>
```

2.2.7.2.13 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>
```

2.2.7.2.14 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>
```

2.2.7.2.15 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.

2.2.7.3 Complex Types

2.2.7.3.1 CHOICEDEFINITION

The **CHOICEDEFINITION** type contains a choice for a **Choice** or **MultiChoice** field.

2.2.7.3.1.1 Schema

```
  <xs:complexType name="CHOICEDEFINITION" mixed="true">
      <xs:attribute name="JumpTo" type="xs:string" />
  </xs:complexType>
```

2.2.7.3.1.2 Attributes

JumpTo: Specifies the **Name** of the next field in a survey list to display when this choice is selected.

2.2.7.3.1.3 Child Elements

<Content>: Specifies a value for a choice in a **Choice** or **MultiChoice** field.

2.2.7.3.2 CHOICEDEFINITIONS

The **CHOICEDEFINITIONS** type contains a collection of choices for a **Choice** or **MultiChoice** field.

2.2.7.3.2.1 Schema

```
<xs:complexType name="CHOICEDEFINITIONS">
    <xs:sequence>
      <xs:element name="CHOICE" type="CHOICEDEFINITION" minOccurs="0" maxOccurs="unbounded"
/>
    </xs:sequence>
</xs:complexType>
```

2.2.7.3.2.2 Attributes

None.

2.2.7.3.2.3 Child Elements

CHOICE: Specifies a choice in a **Choice** or **MultiChoice** field.

2.2.7.3.3 FieldDefinition

A field definition describes the structure and format of a field that is used within a list or content type.

2.2.7.3.3.1 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="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>
```

2.2.7.3.3.2 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 (2)**.

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: If TRUE, the internal type of the field MUST be **Lookup**, the **List** and **FieldRef** attributes of the field MUST refer to a field of internal type **Lookup**, and for the referred field, its **List** attribute MUST refer to the list of which the source field is a member. 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 **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 (2)**.

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 set to 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 values 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 values 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 will be rendered 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.

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 **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.

RelationshipDeleteBehavior: 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 (2)** can be selected.

UserSelectionScope: When present on a **User** field, specifies the group (2) 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 (2) 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.

2.2.7.3.3.3 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**.

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 (2)** 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**.

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**.

FieldRefs: Specifies fields needed to render **DisplayPattern** or **DisplayBidiPattern**. **DisplayPattern** and **DisplayBidiPattern** MUST NOT use fields not listed in this element.

Formula: The formula used to calculate a value for a list item based on the value of other fields in the list. A reader MUST ignore **Formula** if the **Type** attribute is not **Calculated**. A writer SHOULD omit **Formula** if the **Type** attribute is not **Calculated**.

FormulaDisplayNames: The **Formula** element formatted according to the regional settings of the site (2).

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**.

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.

2.2.7.3.4 FieldDefinitionDatabase

The **FieldDefinitionDatabase** type provides an overall container for field definitions in a Content Database.

2.2.7.3.4.1 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>
```

2.2.7.3.4.2 Attributes

None.

2.2.7.3.4.3 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.

2.2.7.3.5 FieldDefinitionDatabaseWithVersion

The **FieldDefinitionDatabaseWithVersion** type provides an overall container for field definitions in **tp_Fields**, specified as follows.

2.2.7.3.5.1 Schema

```
  <xs:complexType name="FieldDefinitionDatabaseWithVersion" mixed="true">
    <xs:all>
      <xs:element name="tp_Fields" type="FieldDefinitionDatabase" minOccurs="1"
maxOccurs="1"/>
    </xs:all>
  </xs:complexType>
```

2.2.7.3.5.2 Attributes

None.

2.2.7.3.5.3 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.

2.2.7.3.6 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.

2.2.7.3.6.1 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>
```

2.2.7.3.6.2 Attributes

See **FieldDefinition** (section 2.2.7.3.3).

2.2.7.3.6.3 Child Elements

See **FieldDefinition** (section 2.2.7.3.3).

2.2.7.3.7 FieldParserRef

The **FieldParserRef** type allows schema authors to override the **Name** attribute of a **FieldDefinition** type (section 2.2.7.3.3) for a specific parser.

2.2.7.3.7.1 Schema

```
<xs:complexType name="FieldParserRef">
    <xs:attribute name="Name" type="xs:string" />
    <xs:attribute name="ProgId" type="xs:string" />
</xs:complexType>
```

2.2.7.3.7.2 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**.

2.2.7.3.7.3 Child Elements

None.

2.2.7.3.8 FieldParserRefs

The **FieldParserRefs** type allows schema authors to override the **Name** attribute of a **FieldDefinition** type (section 2.2.7.3.3) for a collection of parsers.

2.2.7.3.8.1 Schema

```
  <xs:complexType name="FieldParserRefs">
    <xs:sequence>
      <xs:element name="ParserRef" type="FieldParserRef" minOccurs="0" maxOccurs="unbounded"
/>
    </xs:sequence>
  </xs:complexType>
```

2.2.7.3.8.2 Attributes

None.

2.2.7.3.8.3 Child Elements

ParserRef: Specifies the overridden **Name** for a specific parser.

2.2.7.3.9 FieldRefDefinitionField

The **FieldRefDefinitionField** type specifies field definitions that are referenced within another field definition.

2.2.7.3.9.1 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>
```

2.2.7.3.9.2 Attributes

See attributes section of **FieldDefinition** (section 2.2.7.3.3).

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 2.2.7.2.4. 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.

2.2.7.3.9.3 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.

2.2.7.3.10 FieldRefDefinitionIndex

The **FieldRefDefinitionIndex** type specifies a field to use in a composite field index.

2.2.7.3.10.1 Schema

```
<xs:complexType name="FieldRefDefinitionIndex">
  <xs:attribute name="ID" type="UniqueIdentifierWithBracesUppercase" use="required" />
</xs:complexType>
```

2.2.7.3.10.2 Attributes

ID: Identifies a field for which the server can optimize queries.

2.2.7.3.10.3 Child Elements

None.

2.2.7.3.11 FieldRefDefinitionTP

The **FieldRefDefinitionTP** type specifies a reference to a field definition. The attributes specified here override existing values in the field definition.

2.2.7.3.11.1 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>
```

2.2.7.3.11.2 Attributes

See **FieldDefinition** (section 2.2.7.3.3).

2.2.7.3.11.3 Child Elements

None.

2.2.7.3.12 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.

2.2.7.3.12.1 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>
```

2.2.7.3.12.2 Attributes

ID: Identifier for the composite field index.

2.2.7.3.12.3 Child Elements

FieldRef: Specifies a reference to an existing field definition that is used in the current list.

2.2.7.3.13 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.

2.2.7.3.13.1 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>
```

2.2.7.3.13.2 Attributes

Value: String which contains a canonical, non-localizable value for a **CHOICE**.

2.2.7.3.13.3 Child Elements

<content>: Contains a string that MUST specify exactly the string of a corresponding **CHOICE**.

2.2.7.3.14 MAPPINGDEFINITIONS

The **MAPPINGDEFINITIONS** type is a container for one or more **MAPPINGS**. **MAPPINGS** MUST NOT be defined for field types 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.

2.2.7.3.14.1 Schema

```
<xs:complexType name="MAPPINGDEFINITIONS">
    <xs:sequence>
      <xs:element name="MAPPING" type="MAPPINGDEFINITION" minOccurs="0" maxOccurs="unbounded"
/>
    </xs:sequence>
</xs:complexType>
```

2.2.7.3.14.2 Attributes

None.

2.2.7.3.14.3 Child Elements

MAPPING: A canonical value mapping for a **CHOICE**.

2.2.7.4 Elements

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

2.2.7.5 Attributes

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

2.2.7.6 Groups

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

2.2.7.7 Attribute Groups

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

2.2.8 User-Defined Table Types

2.2.8.1 tvpBSNMetadata

The **tvpBSNMetadata** table type represents an array of values to identify a **stream binary piece**.

The **tvpBSNMetadata** table type is defined using T-SQL syntax, as follows.

```
TYPE tvpBSNMetadata AS TABLE (
     DocId uniqueidentifier NULL,
     Partition tinyint NULL,
     BSN bigint NULL,
     StreamId bigint NULL,
     Type tinyint NULL,
     Size int NULL
);
```

DocId: The **document identifier** for the document that is associated with the stream binary piece.

Partition: The **stream partition** that contains the stream binary piece.

BSN: The **BLOB** sequence number of the stream binary piece.

StreamId: The **stream identifier** of the stream binary piece.

Type: The **stream** type of the stream binary piece.

Size: The size, in bytes, of the stream binary piece.

2.2.8.2 tvpBSNMetadata2

The **tvpBSNMetadata2** table type represents an array of values to identify a **stream binary piece**.

The **tvpBSNMetadata2** table type is defined using T-SQL syntax, as follows.

```
TYPE tvpBSNMetadata2 AS TABLE (
     DocId uniqueidentifier NULL,
     Partition tinyint NULL,
     BSN bigint NULL,
     StreamId bigint NULL,
     Type tinyint NULL,
     Size int NULL,
     Expiration datetime NULL
);
```

DocId: The **document identifier** for the document that is associated with the stream binary piece.

Partition: The **stream partition** that contains the stream binary piece.

BSN: The **BLOB** sequence number of the stream binary piece.

StreamId: The **stream identifier** of the stream binary piece.

Type: The **stream** type of the stream binary piece.

Size: The size, in bytes, of the stream binary piece.

Expiration: The timestamp in **UTC** format of when the stream binary piece expires.

2.2.8.3 tvpStreamData

The **tvpStreamData** Table Type represents an array of values to identify a **stream binary piece** and its data. It is defined using T-SQL syntax, as follows.

```
TYPE tvpStreamData AS TABLE (
     BSN bigint NULL,
     Data varbinary(max) NULL,
     Offset bigint NULL,
     Length int NULL,
     RbsId varbinary(64) NULL
);
```

BSN: The **BLOB** sequence number of the stream binary piece.

Data: If the stream binary piece is not stored in remote BLOB storage, this value MUST contain a subset of the binary data of the stream binary piece, otherwise it MUST be NULL. See definition for format.

Offset: The offset into the stream binary piece where this subset data belongs.

Length: The size, in bytes, of this subset data of the stream binary piece.

RbsId: If this stream binary piece is stored in remote BLOB storage, this value MUST contain the remote BLOB storage identifier of a subset of the binary data of the stream binary piece. Otherwise it MUST be NULL. For additional information regarding remote BLOB storage, see [MS-WSSO] section 2.1.2.3.8.

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**, 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** and are also represented by store-relative form URLs.

Lists: A set of information about all **lists (1)** in a content database. List entries are identified by **list identifiers** 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** 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**.

Site Groups: A set of information about all permission levels in a content database. Site group entries are identified by site group identifiers.

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. The following table lists the functions, result sets, and stored procedures in this section.

Name	Description
fn_GetFullUrl	Constructs a **full URL** from two component parts.
proc_AddBuildDependency	Associates a build dependency with a specified document.
proc_AddDocument	Adds a document to the back-end database server with the specified parameters.
proc_AddListItem	Adds a list item to a list.
proc_ChangeLevelForDoc	Updates the **publishing level** of a **document**.
proc_CheckRbsInstalled	Tests whether the back-end database server can support remote **BLOB** storage.
proc_CheckoutDocument	Places a short-term lock on a document, refreshes, converts or releases an existing short-term lock, or checks out a document.
proc_ClearLinks	Deletes all link information associated with a

Name	Description
	particular **field** in a list item as part of a list item or document update.
proc_CreateDir	Creates a directory or folder with a specified name at a specified location.
proc_DeleteAllDocumentVersions	Deletes either the draft versions or older **published versions** of a document and optionally places them in the recycle bin.
proc_DeleteDocBuildDependencySet	Deletes a **build dependency set** for a specified document.
proc_DeleteDocumentVersion	Deletes a document version and optionally places it in the recycle bin.
proc_DeleteUrl	Deletes a document.
proc_DisableRbs	Prepares the back-end database server to stop storing content in remote BLOB storage, and reclaims any back-end database server resources allocated with **proc_EnableRbs**.
proc_DirtyDependents	Marks 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.
proc_EnableRbs	Prepares the back-end database server to store content in remote BLOB storage.
proc_EnumLists	Returns a list of the **lists (1)** in a **site (2)**, along with their associated metadata.
proc_ReadStream	Reads a specific range of data from a document's **stream binary piece**.
proc_FetchDocForHttpGet	Fetches a **document stream** and provides information necessary to render the document on a **front-end Web server**.
proc_FetchDocForRead	Requests the metadata information and document stream of a document.
proc_FetchDocForUpdate	Requests document content and metadata information. It also updates the **CacheParseId** flag for the requested document if the *@CacheParse* parameter is set to 1.
proc_FetchWelcomeNames	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.
proc_GenerateNextId	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 this procedure for the same List.
proc_GetAllRolesForUser	Retrieves the role identifiers for the roles assigned to a set of principals in a given security

Name	Description
	scope.
proc_GetAuditMask	Identifies **Audit Flags** (section 2.2.2.1) information for a specified object.
proc_GetAuditMaskOutput	Gets **Audit Flags** (section 2.2.2.1) information about an object.
proc_GetContainingList	Gets metadata and event receiver information about the list containing a specified URL.
proc_GetContainingListCore	Gets metadata and event receiver information about the list containing a specified URL.
proc_GetDocsMetaInfo	Requests document metadata information for up to ten documents within a specified site (2).
proc_GetLinkInfoSingleDoc	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.
proc_GetListCheckedOutFiles	Retrieves 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.
proc_GetListFields	Gets the mapping of fields in a list.
proc_GetListMetaDataAndEventReceivers	Retrieves information about the metadata, security scopes, Web Parts, and event receivers for a specified list.
proc_getObject	Retrieves a single Configuration Object (section 2.2.5.1) from the Configuration Database (section 3.1.1).
proc_getObjectsByBaseClass	Returns a list of GUIDs for child Configuration Objects (section 2.2.5.1) of the specified parent Configuration Object that inherit from the specified base Class (section 2.2.5.1.1).
proc_getObjectsByClass	Retrieves the GUIDs of Configuration Objects (section 2.2.5.1) based upon the Class type (section 2.2.5.1.1), the parent Configuration Object, and the **Name** (section 2.2.5.1.3) of the Configuration Object.
proc_GetSiteFlags	Returns the **Site Collection Flags** (section 2.2.2.9) set for a specified site collection.
proc_getSiteMap	Determines the **Site Map** (section 3.1.5.38.1) information for a **site collection**.
proc_getSiteMapById	Determines the complete **Site Map** (section

Name	Description
	3.1.5.38.1) information for a site collection.
proc_GetTpWebMetaDataAndListMetaData	Retrieves metadata for a particular site (2) or list (1).
proc_GetUniqueScopesInList	Gets the **WSS ACL Format** (section 2.2.3.6) information for all the unique security scopes of folders and list items contained in a specified list.
proc_GetVersion	Gets the component version number string associated with the specified version identifier GUID.
proc_GetWebMetaInfo	Requests metadata information for a site (2).
proc_GetWebMetainfoByUrl	Requests site metadata information for the site (2) containing a specified **Uniform Resource Locator (URL)**.
proc_ListDocumentVersions	Lists all available version history information for a specified document.
proc_ListRbsStores	Enumerates the remote BLOB storage stores with which the back-end database server has been configured, if any.
proc_ListUrls	Retrieves metadata for a document specified by a URL and the documents contained within it, if any.
proc_RenameUrl	Renames a document or folder within a specified site (2).
proc_SecAddPrincipalToRole	Adds a **security principal** to a role defined within a site collection.
proc_SecAddRoleDef	Creates a new role definition for a specified site (2) within a site collection.
proc_SecAddUser	Adds a new entry for a security principal to the **UserInfo Table** (section 2.2.6.10) that contains descriptive properties and security information about security principals (2), and is stored in the back-end database server.
proc_SecAddUserToSiteGroup	Adds a user to a permission level in the site collection.
proc_SecAddWebMembership	Associates an existing user within a site collection to a site (2) within the same site collection.

Name	Description
proc_SecChangeToInheritedList	Changes the scope of the specified list to the specified site (2) and remove any **role assignment**s and permission settings specific to the list.
proc_SecChangeToInheritedWeb	Changes a site (2) from having its own unique permissions to instead use the permissions inherited from its nearest ancestor with unique permissions.
proc_SecChangeToUniqueScope	Sets a **securable object** such as a site (2), list, or document library to use its own unique security scope, instead of inheriting its security scope from the first ancestor with uniquely permissions.
proc_SecCheckDeletedAccounts	Checks whether a login name exists in the site collection.
proc_SecCloneRoleDefinitions	Creates a copy of the current role definition for a site (2).
proc_SecCreateSiteGroup	Adds a new permission level to a site collection.
proc_SecDecCurrentUsersCount	Reduces by one the total number of users in the specified site collection when configured to use **Active Directory account creation mode**.
proc_SecGetAccountStatus	Provides status information for a site collection's users, which are not marked as deleted, and match the specified login name or email address.
proc_SecGetAclFromScope	Obtains the Access Control List and the anonymous User permissions for a given scope.
proc_SecGetAllAclsForSite	Lists all Scope Identifiers and their associated ACL and anonymous permission masks in a site collection.
proc_SecGetAllGroupsAndMembershipInfo	Returns information about all site groups and site group **members** within a site collection.
proc_SecGetApplicationPrincipalAndUserToken	Returns information about a **principal (1)** based on the principal's login name and user identifier.
proc_SecGetCompleteWebRoleMemberList	Lists all role assignments for all the principals with permissions on a specified Site.
proc_SecGetCurrentUsersCount	Returns a result set containing a count of Users in the specified site collection.
proc_SecGetDomainGroupMapData	Retrieves the domain group map cache

Name	Description
	information for a site collection.
proc_SecGetGroupById	Checks whether the specified Group (either a site group or a domain group) exists in the specified site collection.
proc_SecGetGroupOwner	Retrieves the User Identifier or Site Group Identifier for the owner of a site group.
proc_SecGetGroupSecurityScopes	Retrieves a site group's role assignments information on all security scopes within a given site collection.
proc_SecGetIndividualUrlSecurityCheckEventReceivers	Requests security information and event receiver information about a document at a specified location.
proc_SecGetItemsWithUniquePermissions	Returns information about list items with unique permissions.
proc_SecGetPrincipalByEmail	Returns user information about a user associated with a specified email address.
proc_SecGetPrincipalById	Returns information about a principal or collection of principals based on a specified site group identifier or user identifier.
proc_SecGetPrincipalByIdEx	Returns information about a principal based on a specified user identifier or a collection of principals based on a specified site group identifier.
proc_SecGetPrincipalByLogin	Returns security and attribute information for a principal (a user or domain group) identified by a specified login name.
proc_SecGetPrincipalByLogin20	Returns security principal (2) information based on up to 20 separate login names.
proc_SecGetPrincipalDisplayInformation20	Returns security principal (2) or site group information for up to 20 principal identifiers.
proc_SecGetRoleAssignments	Requests the **WSS ACE** (section 2.2.3.5) for a specified security scope in a site collection.
proc_SecGetRoleBindingsForAllPrincipals	Lists the role assignments for all principals in a security scope.
proc_SecGetRoleDefs	Retrieves role definition information for items in the specified site collection and scope.
proc_SecGetSecurityInfo	Retrieves security permissions information about

Name	Description
	a document.
proc_SecGetSiteAdmins	Returns a list of all site collection administrators not marked as deleted.
proc_SecGetSiteGroupById	Gets information about a site group given its ID.
proc_SecGetSiteGroupByTitle	Gets site group information for the site group with the specified user-friendly name.
proc_SecGetSiteGroupByTitle20	Provides information about site groups in bulk, as specified by a set of up to 20 site group titles.
proc_SecGetUserAccountDirectoryPath	Returns the User Account Directory Path of a specified site collection.
proc_SecGetUserPermissionOnGroup	Determines what permissions the requesting user has within a specified site group.
proc_SecGetWebsAndListWithUniquePermissions	Returns a list of sites and lists that have unique permissions.
proc_SecListAllSiteMembers	Provides information about all non-deleted security principals (2) in the specified site collection.
proc_SecListAllWebMembers	Lists all non-deleted user and domain group accounts registered with a specified site (2).
proc_SecListGroupsInRole	Lists all site groups that have the specified role.
proc_SecListScopeGroups	Lists all site groups that have role assignments in a specified scope.
proc_SecListScopeUsers	Lists 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.
proc_SecListSiteGroupMembership	Lists members in a specified site group.
proc_SecListSiteGroups	Lists site group information for a specified site collection.
proc_SecListSiteGroupsContainingUser	Lists the site groups in which the user is a member.

Name	Description
proc_SecListSiteGroupsWhichUserOwns	Returns site group information for the site groups owned by the specified principal.
proc_SecListUsersInRole	Lists the non-deleted principals assigned to a specified role in the specified scope.
proc_SecMigrateUser	Updates the **SystemID** and login name for a principal in the **UserInfo Table** (section 2.2.6.10) and **AllUserData Table** (section 2.2.6.2).
proc_SecReCalculateWebFGP	Updates the bit at 0x00000400 of the **Site Property Flags** (section 2.2.2.11) to indicate whether the site (2) has at least one uniquely secured object within it.
proc_SecRefreshToken	Updates the **UserInfo Table** (section 2.2.6.10) with information about a specified user's membership in external groups.
proc_SecRemoveGroup	Removes a site group from a site collection.
proc_SecMarkGroupForWebDelete	Marks a site group for later removal from a site collection.
proc_SecRemovePrincipalFromScope	Removes a principal from a **security role** associated with a site in a specified security scope.
proc_SecRemoveRoleDef	Removes a role definition, and any role assignments which use that role definition, from a specified site (2).
proc_SecRemoveUserFromScopeByLogin	Removes a user, specified by login name, from a role in a specified security scope.
proc_SecRemoveUserFromSite	Removes a user from a site collection.
proc_SecRemoveUserFromSiteGroup	Removes a user from a site group in a site collection.
proc_SecRemoveUserFromSiteGroupByLogin	Removes a user identified by login name from a specified site group.
proc_SecResetItemPerm	Causes a list item to revert its set of permissions so that it inherits permissions from its parent rather than has its own.
proc_SecResetWebToDefaultRoleDefinition	Replaces existing roles, role assignments, and Permissions for a specified site (2) and its **subsites** with a default set of role definitions, as specified by input parameters.
proc_SecResolvePrincipal	Retrieves information about a principal or site group in the specified site collection, given the

Name	Description
	principal's login name, e-mail address, or display name.
proc_SecSetSiteGroupProperties	Updates properties of a site group.
proc_SecSetUserAcountDirectoryPath	Sets or clears the **user account directory path** for a specified site collection.
proc_SecSetWebRequestAccess	Sets the request access email address for a specified site (2).
proc_SecUpdateAnonymousPermMask	Modifies the permissions for the anonymous user.
proc_SecUpdateDomainGroupMapData	Updates the domain group map cache of a specified site collection.
proc_SecUpdateRoleDef	Updates a role definition with a new title, description, display order, **Role Definition Type** (section 2.2.1.2.16), and **WSS Rights Mask** (section 2.2.2.15).
proc_SecUpdateUser	Updates the **display name**, **e-mail address**, notes, or administrative privileges listed in the user information associated with a principal.
proc_SecUpdateUserActiveStatus	Marks a user as an active user in a site collection.
proc_TakeOverCheckOut	Overrides checkout for a document that is checked out and has no checked-in draft or published version.
proc_UncheckoutDocument	Releases a checked out document.
proc_UpdateDocBuildDependencySet	Stores an updated version of a build dependency set for a specified document.
proc_UpdateDocument	Updates the metadata and **stream** of a document.
proc_UpdateListItem	Updates a list item in a document library or list.
proc_UpdateListSettings	Updates list (1) metadata.
proc_UpdateUserInfoInTableFromRowUpdater	Updates the user information data for the specified user.
proc_UrlToWebUrl	Returns the store-relative form URL of the site (2) that contains a specified store-relative form URL and is inside a specific site collection.
proc_WriteChunkToAllDocStreams	Establishes new document content associated with the document specified by *@DocId* or appends to already existing document content

Name	Description
	established by an earlier invocation of **proc_WriteChunkToAllDocStreams**.
proc_WriteStreamToRBS	Establishes the remote BLOB storage identifier representing a new stream binary piece for a document.
proc_WriteStreams	Creates a new stream binary piece for a document.
proc_WriteStreamToSQL	Creates a new stream binary piece for a document.
proc_SetStreamsToDoc	Associates stream binary pieces with a document.
proc_SetStreamsToDocNoTVP	Associates stream binary pieces with a document.
TVF_Docs_Url_Level	Returns a subset of rows from the **Docs View** (section 2.2.6.3).
TVF_UserData_ListItemLevelRow	Returns a subset of rows from the **UserData View** (section 2.2.6.8).
TVF_UserData_PId_DId_Level_Row	Returns a subset of rows from the **UserData View** (section 2.2.6.8).
proc_HasCurrentPublishVersion	Tests whether the specified list item has a current published version.

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 2.2.1.1.9) 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,
      @AppPrincipalId               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                       bigint,
        @CreateParentDir                bit,
        @UrlIsSuggestion                bit,
        @ThicketMainFile                bit,
        @CharSet                        int,
        @ProgId                         nvarchar(255),
        @AttachmentOp                   int,
        @VirusVendorID                  int,
        @VirusStatus                    int,
        @VirusInfo                      nvarchar(255),
        @VirusInfoEx                    varbinary(max),
        @LockTimeout                    int,
        @SharedLock                     bit,
        @Comment                        nvarchar(1023),
        @DocDTM                         datetime          OUTPUT,
        @fNoQuotaOrLockCheck            bit,
        @DocContentVerBump              int,
        @BSNBump                        bigint,
        @StreamSchema                   tinyint,
        @DocClientId                    varbinary(16)     = NULL,
        @RequestGuid                    uniqueidentifier  = NULL OUTPUT
);
```

@DocSiteId: The **site collection identifier** for the **site collection** that will contain the document to be stored. This MUST NOT be NULL.

@DocWebId: The **site identifier** for the **site (2)** that will contain the document to be stored. This MUST NOT be NULL.

@UserId: The **user identifier** 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.

@AppPrincipalId: The **app principal** identifier of the **app** that is requesting this operation. If the operation was not requested by an app then it MUST be set to 0.

@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, *@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", this name MUST be replaced with a unique name if the name is not unique and is returned in this output parameter as the actual document leaf name. This parameter MUST NOT be NULL.

@Level: The publishing level type value (section 2.2.2.6) 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 2.2.1.1.2) of the document to be stored. This MUST NOT be NULL, MUST be unique for a new document, and MUST be the same for an existing document adding a new publishing level.

@DoclibId: The list identifier (section 2.2.1.1.5) of the **list (1)** 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 (1) or document library.

@NewDoclibRowId: The identifier for a row in a document library for the document to be stored. This MUST be NULL if the document is not in a list (1) or document library.

@SendingContent: Specifies whether to store the document stream in the back-end database server. This field MUST be one if the document stream of the document is intended to be stored in the back-end database server; otherwise, it MUST be zero.

@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 zero if *@SendingContent* is zero.

@DocMetaInfoSize: Size in bytes of the document's metadata info. This MUST be zero 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 zero 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 (1) or document library into which the document will be stored has content approval enabled. The value MUST be one if the list (1) or document library into which the document will be stored has content approval enabled; otherwise, it MUST be zero. If the document is not to be stored in a list (1) or document library, the value MUST be zero.

@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 **Document Flags** value (section 2.2.2.3) 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", this procedure MUST return a list (1) 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** value (section 2.2.2.7) 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", this procedure 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", the document is a thicket main file. If this parameter is set to "0", the document is not a thicket main file or a **thicket supporting file**. If this parameter is NULL, this document is a thicket supporting file.

@CharSet: An optional parameter that specifies a **code page** 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 editors for **HTML** or **XML** files).

@AttachmentOp: An attachments flag value (section 2.2.1.2.1) 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 type (section 2.2.1.2.17) 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.

@VirusInfoEx: A binary value that specifies extensible virus information for this document, which is opaque to the back-end server.

@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 this procedure MUST set the short-term lock for this period. Otherwise, this 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 one to establish a shared lock; otherwise, it MUST be set to zero.

@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 (2) 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.

@BSNBump: Specifies the amount the **BLOB** sequence number of the document is increased by.

@StreamSchema: Specifies the **stream schema** of the document to be created.

@DocClientId: An optional parameter that specifies the client identifier of the document to be created.

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

Return Values: This procedure returns an integer return code, which MUST be one of the values listed in the following table.

Value	Description
0	Successful execution.
3	The path specified for the document was not found.
5	Access denied error while attempting to create a directory.
80	Attempted to create a directory when the specified directory exists. The document 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 following order: **Site List for Normalization Result Set** (section 3.1.5.3.1), **Checkout Information Result Set** (section 3.1.5.3.2).

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 (2)** 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 the **Common Result Sets URL Result Set** (section 2.2.4.27).

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 (1)** 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 (1)**.

```
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,
      @MtgEventType                 int                 = NULL,
      @EventUID                     nvarchar(255)       = NULL,
      @RecurrenceID                 datetime            = NULL,
      @tp_ColumnSet                 xml                 = NULL,
      @eventData                    varbinary(max)      = NULL,
      @acl                          varbinary(max)      = NULL,
      @DocClientId                  varbinary(16)       = NULL,
      @tp_AppAuthor                 int                 = NULL,
      @tp_AppEditor                 int                 = NULL,
      @RequestGuid                  uniqueidentifier    = NULL      OUTPUT
);
```

@SiteId: The **site collection identifier** for the **site collection** containing the list (1) that the list item is being added to.

@WebId: The **site identifier** for the **site** containing the list (1) that the list item is being added to.

@ListID: The **list identifier** of the list (1) 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 2.2.6.2). 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** that is unique within the list (1).

@UserId: The **user identifier** for the current user. **proc_AddListItem** 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 (1) with a **List Server Template** type value (section 2.2.1.2.12) 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 and the list item to be added does not share the security scope of the master recurrence item, the new list item MUST be given 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** is checked for the current user before adding the list item. The bit MUST be set to one for **Disk Quota** to be checked for the current user before adding the list item; otherwise, the bit MUST be set to zero.

@ItemDocType: The **Document Store Type** (section 2.2.2.4) of the list item being added to the list (1).

@SortTypeReversed: Specifies whether or not the list item sorts using the behavior of folder or file types. The bit MUST be set to one for folders to sort like files; otherwise, it MUST be set to zero. It MUST be set to zero 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, **proc_AddListItem** MUST use this value for the list item identifier to be added.

@DocIdAdded: The optional **document identifier** of the document to be inserted as a list item in the list (1). 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, a new document identifier MUST be generated for the list item. If this list item is being restored to the list (1) as part of an implementation-specific backup restore operation, as specified by the value of *@RetainObjectIdentity*,

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 (1) if this list item is being restored to the list (1) as part of a back-up restore operation.

@RetainObjectIdentity: A bit flag specifying whether this list item is being restored to the list (1) as part of an implementation-specific backup restore operation. If *@RetainObjectIdentity* is set to one, this list item is being restored to the list (1) as part of an implementation-specific backup restore operation and MUST have the *@DocIdAdded* value specified in *@RetainId*.

@Level: The **Publishing Level Type** value (section 2.2.2.6) 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 (1):

- 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 one.
- For the other calls to **proc_AddListItem** for the additional rows to be added for the list item, this value MUST be zero.

@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 (1) 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 2.2.1.2.11) of the list (1) containing the list item.

@CheckSchemaVersion: 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: Specifies the threading structure for this list item in a deprecated discussion board list (1) (a list (1) with a **List Base Type** of three) as a concatenation of UTC date/time stamp values in yyyyMMddHHmmss format. For all list items in lists (1) with other **List Base Types**, this parameter MUST be NULL.

@tp_ThreadIndex: Specifies the list item position within a threaded discussion board list (1) (a list (1) with a **List Base Type** of three) as a binary structure. For all list items in lists (1) 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 2.2.1.2.13) 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: Specifies the implementation-specific order in which the list item is displayed with other list items from the same list (1). This value can be the same as other list items in the list (1).

@tp_InstanceID: Specifies a **meeting instance** identifier if the list item is associated with the particular meeting instance of a recurring meeting. For all other list items, this parameter MUST be NULL.

@tp_GUID: A **GUID** that uniquely identifies the list item within the **AllUserData** table.

@tp_Id: The optional **List Item Identifier** (section 2.2.1.1.6) 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 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, 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, this value MUST be zero.

@tp_WorkflowVersion: If the list item is part of a **workflow**, this parameter specifies the value to set denoting the state of the list item within that workflow. If the list item is not part of a workflow, this MUST be NULL.

@tp_WorkflowInstanceID: A **workflow identifier** that specifies the currently active **workflow instance** on this list item. If this list item is not part of a workflow, this MUST be NULL.

@nvarchar1: User-defined column in the list (1) containing a value of type nvarchar. If the column contains no data, the value MUST be NULL.

@MtgEventType: If *@ServerTemplate* corresponds to a **Meeting Workspace site**, this parameter specifies the type of calendar event.

@EventUID: If *@ServerTemplate* corresponds to a Meeting Workspace site, this parameter specifies the unique identifier for the meeting instance.

@RecurrenceID: If *@ServerTemplate* corresponds to a Meeting Workspace site, this parameter specifies a particular instance of a recurring meeting.

@tp_ColumnSet: The **XML** fragment that specifies the values of user-defined columns in the list (1). The name of each element specified in this parameter MUST be one of the columns defined in the **AllUserData** table with the SPARSE keyword.

@eventData: Contains implementation-specific **event (1)** data that is significant to the front-end Web server but is 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 2.2.3.6) **ACL** for the data supplied in *@eventData*. To be stored with the data.

@DocClientId: An optional parameter that specifies the client identifier of the list item to be saved. This can be NULL.

@tp_AppAuthor: The identifier of the **app principal** who created the list item.

@tp_AppEditor: The app principal identifier of the app principal who last edited the list item.

@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.
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, this procedure also can update the document's **moderation status**, 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** 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** (section 2.2.2.6) value indicating the requested publishing level for the document. If the document is currently checked out and *@NewLevel* is set to 1 (Published), then the document's level value MUST be changed to 1 (Published). Otherwise, the document's level value MUST be changed to 255 (Checked Out). If *@NewLevel* is set to 255 (Checked Out), *@EnableMinorVersions* is 0 and *@NewLevel* is set to 1 (Published), then the document's level value MUST NOT change, and output parameters MUST not be changed. If the document's current Level is 255 (Checked Out) and the *@NewLevel* is 2 (Draft), 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 2.2.1.2.13). See the Moderation Status section for a list of all valid values.

@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 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 2.2.6.3) 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: This 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.

This procedure MUST NOT return a result set.

3.1.5.6 proc_CheckRbsInstalled

The **proc_CheckRbsInstalled** 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_CheckRbsInstalled** 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_CheckRbsInstalled** 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 for the site collection containing the document to be checked out, locked, or unlocked.

@WebId: The site identifier 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 2.2.1.1.13) 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 locking 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.

Value	Description
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. A shared short-term lock is used for read operations that do not change or update data, such as a SELECT statement.
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. An exclusive short-term lock is used for data-modification operations, such as INSERT, UPDATE, or DELETE. Such a lock ensures that multiple updates cannot be made to the same resource at the same time.

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 the 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 the Common Result Sets **Link Information Result Set** (section 2.2.4.13).

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 the Common Result Sets **Document Metadata Result Set** (section 2.2.4.8).

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 (2)** 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 2.2.1.2.5) of 3 (list item) for this document.

The **Event Receivers Result Set** is defined in the Common Result Sets **Event Receivers Result Set** (section 2.2.4.11).

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 the Common Result Sets **Site Audit Mask Result Set** (section 2.2.4.22).

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 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,
      @AppPrincipalId             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 2.2.1.1.9) for the **site collection** containing the directory to be created.

@DirWebId: The **site identifier** for the **site (2)** 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** value (section 2.2.2.6) specifying the publishing status of the directory to be created. This value MUST be one (Published) if the directory is not being created in a **list (1)** or **document library**. If this parameter is two, specifying that the **list item** is to be created as a draft, the *@UserId* value MUST be set as the draft owner for the created directory.

@AddMinorVersion: If this parameter has a value of one, **proc_CreateDir** MUST set the major version number of the created directory to zero and the minor version number to one. If this parameter has a value of zero, **proc_CreateDir** MUST set the major version number of the created directory to one and the minor version number to zero. This parameter MUST NOT be NULL.

@DocFlags: A **Doc Flags** value (section 2.2.2.3) specifying metadata about the directory to be created. If the *@DocFlags* value has the 0x00002000 bit set (specifying a custom ordering of content types), 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 zero, with none of the bits set, or bits MUST be set as follows.

Value	Description
0x00000007	These three bits contain an **Attachment Flag** value (section 2.2.1.2.1).
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** value (section 2.2.1.2.13).
0xFFFFFC40	These values are currently unused and MUST be ignored.

@UserId: The **User Identifier** (section 2.2.1.1.13) for the current user who is directly or indirectly requesting the directory creation. If this parameter is not NULL, it MUST be used by **proc_CreateDir** for permission checking and setting ownership of the created directory. If this parameter is NULL, the ownership of the created directory MUST be set from the parent.

@AppPrincipalId: The **app principal** identifier of the **app** that is requesting this operation. If the operation was not requested by an app then it MUST be set to 0.

@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 zero 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, this value MUST be unique in the back-end database server and MUST be used as the **document identifier** for the directory to be created. If this parameter is passed in with a NULL value, a new uniqueidentifier value MUST be generated for the directory. If **proc_CreateDir** creates a new directory, 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), 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 2.2.1.1.8) 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 2.2.1.1.6) 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, **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 sorts 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, it MUST be set as the **Scope Identifier** (section 2.2.1.1.8) 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 zero if the directory did not already exist and set to one 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.

Value	Description
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 (1).
1816	There is not enough database quota for the current user to complete the operation.

The **proc_CreateDir** stored procedure MUST NOT return any 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,

      @AppPrincipalId               int,
      @DeleteOp                     int,
      @Level                        tinyint,
      @RequestGuid                  uniqueidentifier   = NULL      OUTPUT
);
```

@DocSiteId: The **site collection identifier** (section 2.2.1.1.9) 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 2.2.1.1.13) of the current user requesting the deletion. This value MUST refer to an existing user identifier for the site collection specified by *@DocSiteId*.

@AppPrincipalId: The **app principal** identifier of the **app** that is requesting this operation. If the request if not made by an app then it MUST be set to 0.

@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 any result sets.

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 2.2.1.1.9) 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,
      @AppPrincipalId               int,
      @DeleteOp                     int
      @RequestGuid                  uniqueidentifier   = NULL      OUTPUT
);
```

@DocSiteId: The **site collection identifier** (section 2.2.1.1.9) 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 2.2.1.1.13) of the current user requesting the deletion. This value MUST refer to an existing user identifier for the specified site collection.

@AppPrincipalId: The **app principal** identifier of the **app** requesting the deletion. If the request is not made by an app then it MUST be set to 0.

@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.
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 any result sets.

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,
    @AppPrincipalId                 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 2.2.1.1.9) of the **site collection** containing the document.

@WebId: The **site identifier** (section 2.2.1.1.11) of the **site (2)** containing the document.

@Url: The store-relative form **URL** of the document.

@UserId: The **user identifier** (section 2.2.1.1.13) 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.

@AppPrincipalId: The **app principal** identifier of the **app** requesting this operation. If this operation was not requested by an app then it MUST be set to 0.

@LogChange: This parameter is reserved for internal use and MUST be the default value.

@ListDeletedUrls: If this bit is set to 1, then the result set information about the deleted documents is requested. See the **Deleted Documents Result Set** (section 3.1.5.13.1).

@ListDeletedAliases: If this bit is set to 1, then the result set information about deleted lists that had configured e-mail addresses is requested. See the **Deleted Aliased Lists Result Set** (section 3.1.5.13.2).

@AttachmentsFlag: An **Attachments Flag** (section 2.2.1.2.1) 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.

Value	Description
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 (2) 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 because of 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 e-mail address. Such lists provide implementation-specific e-mail 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 of the site collection containing the deleted list.

tp_WebId: The site identifier of the **site (2)** containing the list.

tp_Id: The list identifier 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**.

```
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 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 (2)** usage statistics. The *@DepDesc* parameter is the store-relative form URL of the site (2).

@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 MUST return value 0.

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.

Value	Description
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 (1)** in a **site (2)**, along with their associated metadata.

```
PROCEDURE proc_EnumLists (
      @SiteId                     uniqueidentifier,
      @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
);
```

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

@WebId: The **site identifier** for the site (2) containing the requested lists (1).

@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 2.2.1.2.4) 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 2.2.1.2.11). 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 this procedure.

@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 2.2.1.2.12) 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 2.2.1.1.8), 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 2.2.1.1.13) that specifies 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. 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 (1)** contained in the specified **site (2)**. This 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,
Followable                      bit,
tp ValidationFormula            nvarchar(1024),
tp_ValidationMessage            nvarchar(1024),
TitleResource                   nvarchar(256),
DescriptionResource             nvarchar(256),
tp_ACL                          varbinary(max);
```

tp_DocTemplateUrl: Contains the **store-relative URL** 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 2.2.1.1.5) 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 2.2.1.2.11) value from which the list is derived.

tp_FeatureId: Contains a Feature Identifier (section 2.2.1.1.4) 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 2.2.1.2.12) enumeration or can be a custom value as defined in the disk-based template of the list. A custom list template value MUST be unique and MUST be greater than 10000.

tp_Created: Contains the time in **Coordinated Universal Time (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 is left to right.
2	Text flow is 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 2.2.2.15) 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 2.2.1.1.13) 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 e-mail address of the list. This alias is used to allow files to be sent directly to the list through an implementation-specific e-mail handling feature. Can be NULL if the list is not an **e-mail enabled list**.

tp_WebFullUrl: Contains the store-relative URL of the site (2) that contains the list.

tp_WebId: Contains the Site Identifier (section 2.2.1.1.11) of the site (2) 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** 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 2.2.1.1.16) 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 per-list permissions 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.

Followable: If this list is followable, then this value MUST be 1. Otherwise, it MUST be 0.

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_ReadStream

The **proc_ReadStream** **stored procedure** is called to read a specific range of data from a document's **stream binary piece**.

```
PROCEDURE proc_ReadStream (
      @SiteId               uniqueidentifier,
      @ParentId             uniqueidentifier,
      @DocId                uniqueidentifier,
      @Level                tinyint,
      @Partition            tinyint,
      @BSN                  bigint,
      @Offset               int,
      @Length               int
);
```

@SiteId: The **site collection identifier** for a **site collection** containing the document.

@ParentId: The **document identifier** of the document's parent container.

@DocId: The document identifier of the document to read from.

@Level: The **publishing level** of the document.

@Partition: The **stream partition** which contains the stream binary piece to read.

@BSN: The **BLOB** sequence number of the stream binary piece to read.

@Offset: The offset, in bytes, into the stream binary piece to begin reading.

@Length: The number of bytes to read from the stream binary piece.

Return Values: This 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_ReadStream** stored procedure MUST return the **Document Stream Chunk Result Set** (section 3.1.5.18.1).

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 **stream binary piece**.

```
{Content}                          varbinary(max);
```

{Content}: The data read from the document's stream binary piece.

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),
      @AppPrincipalName             nvarchar(256),
      @IsHostHeaderAppPrincipalName bit,
      @CurrentVirusVendorID         int,
      @PrefetchListScope            bit,
      @ChunkSize                    int,
      @DGCacheVersion               bigint,
      @MaxCheckinLevel              tinyint,
      @HonorLevel                   bit,
      @CurrentFolderUrl             nvarchar(260),
      @ThresholdRowCount            int,
      @StreamPartition              tinyint,
      @Level                        tinyint           OUTPUT,
      @FetchStreamIfNeeded          bit = 1,
      @RequestGuid                  uniqueidentifier = NULL OUTPUT
);
```

@DocSiteId: The **site collection identifier** 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", 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. If this parameter is "0", information to satisfy an **HTTP GET** request MUST be returned, including the **stream** of the document. If this parameter is "1", information needed only to satisfy an **HTTP HEAD** request MUST be returned, which does not include the stream of the document. All nonzero values MUST also be treated as "1".

@ValidationType: This parameter MUST have a value of "0", "1", "2", or "3". If this parameter is "0", *@ClientVerson*, *@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", 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** of the document to fetch. If *@ValidationType* is "1" or "3", 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 or an implementation-specific Web page that renders an item or items in a **list (1)** 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 (1) metadata, users and permission levels metadata, Web Parts, and related list (1) metadata. In addition, a value of one indicates that the metadata is security trimmed to the user specified by *@SystemId*. The value zero 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, the **Document Build Dependency Set Result Set** (section 3.1.5.19.12) and **Document Build Dependency Metadata Result Set** (section 3.1.5.19.13) MUST be returned. Otherwise, the result sets MUST NOT be returned.

@SystemID: The **security identifier (SID)**, as specified in [MS-DTYP] section 2.4.2, of the current user, or NULL to indicate an anonymous user.

@AppPrincipalName: The **app principal** identifier or the **app web domain identifier** associated with the current user. If the current user is not associated with an app principal, this parameter MUST be NULL.

@IsHostHeaderAppPrincipalName: If this parameter is set to one, the *@AppPrincipalName* parameter MUST be the app principal identifier, else it MUST be the app web domain identifier.

@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 (1) scope that contains the document MUST be returned.

@ChunkSize: Specifies the maximum size, in bytes, of the document content to be returned in the **{Content}** column in the **Document Information (Get) Stream Result Set** (section 3.1.5.19.9). For an uncustomized document, this MUST be NULL. Otherwise, if the data in the **{Content}** column 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.

@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", 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", 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 (1) 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", this parameter MUST be ignored.

@StreamPartition: Specifies the **stream partition** of the document to fetch.

@Level: An output parameter specifying the publishing level of the document to be fetched.

@FetchStreamIfNeeded: Specifies if the stream of the document is 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 one of those 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.
1168	The site collection specified by *@DocSiteId* was not found.
1271	The site (2) is read locked.

Result Sets: 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 zero (File):

- **Site Collection Audit Mask Result Set** (section 3.1.5.19.10)
- **List Audit Mask Result Set** (section 3.1.5.19.11)
- **Document Build Dependency Set Result Set** (section 3.1.5.19.12)
- **Document Build Dependency Metadata Result Set** (section 3.1.5.19.13)
- **Domain Group Cache Versions Result Set** (section 3.1.5.19.2)
- **Site Metadata Result Set** (section 3.1.5.19.14)

- **Event Receivers Result Set** (section 3.1.5.19.15)
- **Web Event Receiver Result Set** (section 3.1.5.19.16)
- **Site Features List Result Set** (section 3.1.5.19.17)
- **WebParts Metadata, Personalized Result Set** (section 3.1.5.19.18)
- **WebParts Metadata, Nonpersonalized Result Set** (section 3.1.5.19.19)
- **ListMetadata Result Set** (section 3.1.5.19.20)
- **List Event Receivers Result Set** (section 3.1.5.19.21)
- **List Security Information Result Set** (section 3.1.5.19.22)
- **Site Collection Custom Actions Result Set** (section 3.1.5.19.23)
- **Site Custom Actions Result Set** (section 3.1.5.19.24)
- **List Custom Actions Result Set** (section 3.1.5.19.25)
- **List Web Parts Result Set** (section 3.1.5.19.26)
- **Content Type Order Result Set** (section 3.1.5.19.27)
- **Current Folder Scope Result Set** (section 3.1.5.19.28)
- **Navigation Context Security Information Result Set** (section 3.1.5.19.29)
- **NULL Navigation Context Security Information Result Set** (section 3.1.5.19.30)
- **Empty Navigation Context Security Information Result Set** (section 3.1.5.19.31)

Other **proc_FetchDocForHttpGet** result sets return conditionally, as described in each result set section, in the following sections.

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),
        {VirusInfoEx},                  varbinary(max),
        {ContentModifiedSince}          bit,
        {ProgId}                        nvarchar(255),
        {DoclibRowId}                   int,
        {Language}                      int,
        {DirName}                       nvarchar(256),
        {UIVersion}                     int,
        {ContentVersion}                int,
        {RbsCollectionId}               int,
        {NextBSN}                       bigint,
        {StreamSchema}                  byte,
        {InternalVersion}               int,
        {WebFlags}                      int,
        {AppWebDomainId}                varchar(8),
        {SiteAppHostHeader}             nvarchar(55),
        {DocScopeId}                    uniqueidentifier,
        {DenyPermMask}                  tPermMask
;
```

{Size}: The size, in bytes, of the **document stream**.

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

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

{WebId}: The **site identifier** of the **site (2)** containing the document.

{FirstUniqueWebId}: The site identifier of the site (2) whose security permissions are the effective security permission for the site (2) containing the specified document.

{SecurityProvider}: COM CLSID of the **security provider** for this site (2).

{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 zero. 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** 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 to read the document, this value MUST be "1".

{SiteFlags}: A **Site Collection Flags** value (section 2.2.2.9) describing the configuration of the **site collection** containing the document.

Acl: The **WSS ACL Format** (section 2.2.3.6) permissions for the document.

AnonymousPermMask: A **WSS Rights Mask** (section 2.2.2.15) that indicates the rights granted to an anonymous user on this document.

{ListIdForPermissionCheck}: The **GUID** for the **list (1)** containing the document. This value MUST be NULL if this document is not in a list (1).

{PermCheckedAgainstUniqueList}: MUST be zero.

DraftOwnerId: The identifier for the user who published the document as a draft version. If the document is not a draft version, this value MUST be NULL.

ListFlags: A **List Flags** value (section 2.2.2.5) describing the list (1) that contains the document. If the document is not in a list (1), the value MUST be zero.

Level: The **publishing level** value of this document.

{IsCurrentVersion}: If the document being returned is the **current version**, as defined by the implementation, 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 2.2.1.2.17) 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}: An anti-virus scanner specific message returned by the anti-virus scanner when it last processed the document.

{VirusInfoEx}: A binary value that specifies extensible virus information for this document, which is opaque to the back-end server.

{ContentModifiedSince}: A bit indicating whether the document has been modified, depending on the validation type in use. This MUST be set to one 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 zero.

{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 (1). If the requested document is not contained in a list (1), this value MUST be NULL.

{Language}: The **LCID** of the **locale** of the site (2).

{DirName}: The **directory name** of the requested document.

{UIVersion}: The **displayed 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.

{NextBSN}: The current BLOB sequence number of the document being returned.

{StreamSchema}: The current **stream schema** of the document being returned.

{InternalVersion}: The current **internal version number** of the document being returned.

{WebFlags}: A **Site Property Flags** value (section 2.2.2.11) describing the configuration of the site (2) containing the document.

{AppWebDomainId}: The **app web domain identifier** of the site (2) containing the document being returned.

{SiteAppHostHeader}: The **app host header name** of the site collection containing the document being returned.

{DocScopeId}: The GUID identifying the **security scope** of the document. This value MUST NOT be NULL.

{DenyPermMask}: A **WSS Rights Mask** (section 2.2.2.15) specifying the rights to deny to the current user.

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 the Common Result Sets **Domain Group Cache Versions Result Set** (section 2.2.4.4).

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 2.2.4.4).

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 the Common Result Sets **Domain Group Cache BEDS Update Result Set** (section 2.2.4.3).

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 2.2.4.3).

The **Domain Group Cache WFE Update Result Set** is defined in the Common Result Sets **Domain Group Cache WFE Update Result Set** (section 2.2.4.5).

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.

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 e-mail 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** for the specified user was last updated.

tp_Token: A **WSS User Token** value specifying the permission level 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 (2)** 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 (2)** 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** 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 (2)** 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 2.2.1.2.15) of the URL. This parameter MUST NOT be 0 .For all other valid values, see the **Redirect Type** section.

{RedirectUrl}: The full redirect URL.

{WelcomePageParameters}: MUST be NULL.

{ContentTypeId}: The document's content type identifier.

3.1.5.19.8 Document Information (Get) Metadata Result Set

This result set contains the metadata for the document's binary **stream**. This row set MUST be returned only if *@FetchType* is not set to 1, otherwise, the row set MUST not be returned.

This 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.

The value returned in the **{HistVersion}** column MUST be 0.

This result set is defined as **Document Content Metadata Result Set** (section 2.2.4.6) in the Common Result Sets (section 2.2.4).

3.1.5.19.9 Document Information (Get) Stream Result Set

This result set contains a row for each of the document's **stream binary pieces**. This row set MUST be returned only if *@FetchType* is not set to 1.

This result set MUST return zero or more rows. If *@FetchStreamIfNeeded* is not set to 1 or the document requires dependency update processing or the document is an uncustomized document, this row set MUST contain 0 rows.

This result set is defined as **Document Content Stream Result Set** (section 2.2.4.7) in the Common Result Sets (section 2.2.4).

3.1.5.19.10 Site Collection Audit Mask Result Set

The **Site Collection Audit Mask Result Set** contains the information about the **Audit Flags** associated with the site collection containing the specified document.

The **Site Collection Audit Mask Result Set** is defined in the Common Result Sets **Site Audit Mask Result Set** (section 2.2.4.22).

3.1.5.19.11 List Audit Mask Result Set

The **List Audit Mask Result Set** contains information about the **Audit Flags** 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 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.

3.1.5.19.12 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.

3.1.5.19.13 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. This result set MUST be returned only if the *@FetchBuildDependencySet* is set to one and the binary array returned in the **Document Build Dependency Set Result Set** (section 3.1.5.19.12) 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** of the closest **site (2)** 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 (2). This MUST be NULL for sites (2) using the native security implementation.

Acl: The binary serialization of the **WSS ACL Format** (section 2.2.3.6) **ACL** for this site (2).

AnonymousPermMask: A **WSS Rights Mask** (section 2.2.2.15) 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 (2) containing the document.

ListId: The **list identifier** for the **list (1)** containing the document.

UniqueList: This value MUST be zero.

InDocLibrary: The value MUST be set to one if the document is contained within a **document library**. Otherwise, MUST be set to zero.

DraftOwnerId: The **user identifier** for the owner of the last checked-in draft version of the document.

ListFlags: A **List Flags** value (section 2.2.2.5) describing the list (1) that contains this document. This value MUST be NULL if the document is not stored in a list (1).

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 an **uncustomized** 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 uncustomized. 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	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).
15	The **SetupPath** is relative to the install location of Microsoft SharePoint Foundation 2013 on the front-end Web server (for example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\15).

SetupPath: For a document that is now or once was uncustomized, 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 uncustomized.

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 (2) for use in pages of the site (2) when rendered on the front-end Web server.

CustomMasterUrl: The URL for an alternate master page registered on the site (2) for use in pages of the site (2) rendered on the front-end Web server.

Version: A counter incremented any time a change is made to this document.

Id: The **document identifier**.

BuildDependencySet: A **varbinary(max)** value specifying further document dependencies.

3.1.5.19.14 Site Metadata Result Set

The **Site Metadata Result Set** contains metadata for the **site (2)** 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 the Common Result Sets **Site Metadata Result Set** (section 2.2.4.24).

3.1.5.19.15 Event Receivers Result Set

The **Event Receivers Result Set** contains information about the event receivers defined for the **site (2)** containing the specified document.

The Event Receivers Result Set is part of a series of result sets describing view Web page-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 (2). The **Event Receivers Result Set** is defined in the Common Result Sets **Event Receivers Result Set** (section 2.2.4.11).

3.1.5.19.16 Web Event Receiver Result Set

The **Web Event Receivers Result Set** contains information about the event receivers defined for the **site (2)** containing the specified document.

The **Web Event Receivers Result Set** is part of a series of result sets describing view Web page-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 (2). The Event Receiver Result Set is defined in the Common Result Sets **Event Receivers Result Set** (section 2.2.4.11).

3.1.5.19.17 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 (2)** 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 Features List Result Set** is defined in the Common Result Sets **Site Feature List Result Set** (section 2.2.4.23).

3.1.5.19.18 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 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** 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 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** of this Web Part. This value can be NULL.

tp_Source: Properties of the Web Part, as specified by a compatible HTML editor. 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.

3.1.5.19.19 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 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 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 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 of this Web Part. This value can be NULL.

tp_Source: Properties of the Web Part as specified by a compatible HTML editor. 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.

3.1.5.19.20 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 3.1.5.19.18), or **Web Parts Metadata, Nonpersonalized** (section 3.1.5.19.19)).

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 the Common Result Sets **List Metadata Result Set** (section 2.2.4.14).

3.1.5.19.21 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 2.2.4.14) 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 the Common Result Sets **Event Receivers Result Set** (section 2.2.4.11).

3.1.5.19.22 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 2.2.4.14) 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 of the list.

ScopeId: The Scope Identifier 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** on the list in effect for anonymous users. This MUST be 0 if **ScopeId** is zero.

3.1.5.19.23 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 Web page-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 the Common Result Sets **Custom Actions From Scope Result Set** (section 2.2.4.2).

3.1.5.19.24 Site Custom Actions Result Set

The **Site Custom Actions Result Set** MUST return 1 row for each custom action defined for the site collection containing the specified document.

The **Site Custom Actions Result Set** is part of a series of result sets describing view Web page-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 the Common Result Sets **Custom Actions From Scope Result Set** (section 2.2.4.2).

3.1.5.19.25 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 Web page-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 2.2.4.14) 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 the Common Result Sets **Custom Actions From Scope Result Set** (section 2.2.4.2).

3.1.5.19.26 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 2.2.4.14) 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 the Common Result Sets **List Web Parts Result Set** (section 2.2.4.15).

3.1.5.19.27 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'**.

3.1.5.19.28 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 Web page 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 of the security scope effective for the specified folder.

{FolderId}: The integer identifier of the folder document within the containing list.

3.1.5.19.29 Navigation Context Security Information Result Set

The **Navigation Context Security Information Result Set** contains Security information about the **site (2)** 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 the Common Result Sets **Unique Permissions Result Set** (section 2.2.4.26).

3.1.5.19.30 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 (2)** 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 the Common Result Sets **NULL Unique Permissions Result Set** (section 2.2.4.17).

3.1.5.19.31 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 3.1.5.19.29) nor the **NULL Navigation Context Security Information** (section 3.1.5.19.30) 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 the Common Result Sets **Empty Result Set** (section 2.2.4.10).

3.1.5.19.32 App Principal Information Result Set

If the *@AppPrincipalName* parameter is not NULL, the App Principal Info Result Set MUST be returned as defined in section 2.2.4.29.

3.1.5.19.33 App Principal Permissions Result Set

The **App Principal Permissions Result Set** is returned to indicate what permissions the specified **app principal** has for the document as specified in section 2.2.4.30. If the specified app principal has permissions for the specified document, the **App Principal Permissions Result Set** MUST be returned.

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,
      @StreamPartition              tinyint,
      @Level                        tinyint OUTPUT,
      @UIVersion                    int OUTPUT,
      @RequestGuid                  uniqueidentifier = NULL OUTPUT
);
```

@DocSiteId: The **Site Collection Identifier** (section 2.2.1.1.9) of the **site collection** containing the requested document.

@DocWebId: The **Site Identifier** (section 2.2.1.1.11) of the **site (2)** 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 3.1.5.20.14) will indicate if the ManageLists bit of the **WSS Rights Mask** (section 2.2.2.15) is set.

@GetContent: A bit flag specifying whether to return the document stream of the document or not (0).

@GetWebListForNormalization: A bit flag specifying whether to return the subsite of the site (2) specified by *@DocWebId*. If this flag is set to 1, then this 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 (1) containing the requested document. If this flag is set to 1, then this 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** (section 3.1.5.20.14).

@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 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.

@GetLinkInfo: Specifies whether to return the link information about the requested document. If this flag is set to 1, then this procedure MUST return the **Link Info Single Doc Result Set** (section 3.1.5.20.2).

@UserId: The **User Identifier** (section 2.2.1.1.13) 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 **{Content}** column in the **Document Information and Content (Read) Stream Result Set** (section 3.1.5.20.11) and the **Document Version Information and Content Stream Result Set** (section 3.1.5.20.13). For an **uncustomized** document, this MUST be NULL. Otherwise, if the data in the **{Content}** column 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.

@StreamPartition: Specifies the **stream partition** of the document to fetch.

@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 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 URLs** for all subsites within the **site collection** whose parent **site (2)** 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 the Common Result Sets **URL Result Set** (section 2.2.4.27).

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 the Common Result Sets **Link Information Result Set** (section 2.2.4.13).

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 the Common Result Sets **Document Metadata Result Set** (section 2.2.4.8).

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 section **Document Version Metadata Result Set** (section 2.2.4.9), located in the Common Result Sets (section 2.2.4).

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

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 2.2.1.2.5) of 3 (list item) for this document.

The **Event Receivers Result Set** is defined in the Common Result Sets **Event Receivers Result Set** (section 2.2.4.11).

3.1.5.20.7 List Metadata Result Set

The **List Metadata Result Set** contains the metadata associated with the **list (1)** 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 the Common Result Sets **List Metadata Result Set** (section 2.2.4.14).

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 the Common Result Sets **Empty Result Set** (section 2.2.4.10).

3.1.5.20.9 Event Receivers Result Set (1)

The **Event Receivers Result Set** contains the **event receivers** associated with the **list (1)** 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 2.2.1.2.5) of 2 (List) for the List.

The **Event Receivers Result Set** is defined in the Common Result Sets **Event Receivers Result Set** (section 2.2.4.11).

3.1.5.20.10 Document Information and Content (Read) Metadata Result Set

This result set contains metadata about the **document stream** for the requested **document**.

This result set MUST only be returned if *@GetContent* is set to 1 and *@Version* is negative. If the document exists, it MUST contain one row; otherwise, it MUST contain zero rows.

The value returned for the **{HistVersion}** column MUST be 0.

This result set is defined as **Document Content Metadata Result Set** (section 2.2.4.6) in the list of Common Result Sets (section 2.2.4).

3.1.5.20.11 Document Information and Content (Read) Stream Result Set

This result set contains a row for each of the document's **stream binary pieces**. This result set MUST only be returned if *@GetContent* is set to "1" and *@Version* is negative.

This result set MUST return zero or more rows. If the requested document is an uncustomized document, this result set MUST contain 0 rows.

This result set is defined as **Document Content Stream Result Set** (section 2.2.4.7) in the Common Result Sets (section 2.2.4).

3.1.5.20.12 Document Version Information and Content Result Set

This result set contains metadata about the document stream for the requested document.

This result set MUST only be returned if *@GetContent* is set to 1 and *@Version* is not negative. If the document exists, it MUST contain one row; otherwise, it MUST contain zero rows.

The value returned for the **{HistVersion}** column MUST be the **UI version** of the requested document version.

This result set is defined as **Document Content Metadata Result Set** (section 2.2.4.6) in the list of Common Result Sets (section 2.2.4).

3.1.5.20.13 Document Version Information and Content Stream Result Set

This result set contains a row for each of the document's stream binary pieces. This result set MUST only be returned if *@GetContent* is set to 1 and *@Version* is not negative.

This result set MUST return zero or more rows.

This result set is defined as **Document Content Stream Result Set** (section 2.2.4.7) in the Common Result Sets (section 2.2.4).

3.1.5.20.14 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 2.2.1.2.1) for more information.

{NeedManageListRight}: This specifies whether operations on this document require the user to have the **ManageLists** bit of the **WSS Rights Mask** set (section 2.2.2.15). 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.

3.1.5.20.15 Audit Mask Result Set

The **Audit Mask Result Set** contains the information about the **Audit Flags** 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 the Common Result Sets Site **Audit Mask Result Set** (section 2.2.4.22).

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,
      @StreamPartition              tinyint,
      @Level                        tinyint          OUTPUT,
      @RequestGuid                  uniqueidentifier = NULL OUTPUT
);
```

@DocSiteId: The Site Collection Identifier (section 2.2.1.1.9) of the **site collection** containing the requested document.

@DocWebId: The Site Identifier (section 2.2.1.1.11) of the **site (2)** 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 2.2.1.1.13) 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 for to return either the **Document Information and Content Result Set** (section 3.1.5.21.14), the **Document Version 1 Information and Content Result Set** (section 3.1.5.21.16), or the **Document Version 2 Information and Content Result Set** (section 2).

@GetWebListForNormalization: A bit flag specifying whether to return the subsite of the site (2) 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 2.2.6.3) 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 this procedure MUST return the **Event Receivers Result Set** (section 3.1.5.21.6) 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 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 3.1.5.21.18) will indicate if the **ManageLists** bit of the **WSS Rights Mask** (section 2.2.2.15) is set.

@ChunkSize: Specifies the maximum size, in bytes, of the document content to be returned in the **{Content}** column in the **Document Information and Content (Update) Stream Result Set** and the **Document Version Information and Content Stream Result Set**. For an **uncustomized** document, this MUST be NULL. Otherwise, if the data in the **{Content}** column 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.

@StreamPartition: Specifies the **stream partition** of the document to fetch.

@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: This procedure MUST return an integer return code of 0.

This procedure returns a **Publishing Level Type** value in the parameters as specified earlier. This procedure MUST return multiple result sets. Some of the result sets MUST NOT be returned, depending upon input parameters or calculations performed in this procedure, 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 URLs** for all subsites within the **site collection** whose parent **site (2)** 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 zero rows if there are no such subsites.

The **Subsite List Result Set** is defined in the Common Result Sets **URL Result Set** (section 2.2.4.27).

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 the Common Result Sets **ACL and Permission Result Set** (section 2.2.4.1).

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 zero rows.

The **Document Metadata Result Set** is defined in the in the Common Result Sets **Document Metadata Result Set** (section 2.2.4.8).

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 zero rows.

The **Document Version Metadata Result Set** is in section **Document Version Metadata Result Set** (section 2.2.4.9), located in the Common Result Sets (section 2.2.4).

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

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 (2)** specified by *@DocWebId*. The **Event Receivers Result Set** MUST contain one row for each event receiver registered with an **Event Host Type** (section 2.2.1.2.5) of 3 (list item) for this document.

The result set is defined in the Common Result Sets **Event Receivers Result Set** (section 2.2.4.11).

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 a **server-relative URL**. Otherwise, MUST be 0 to indicate that the link URL is not a server-relative URL.

LinkStatus: A **Document Store Type** 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 (such as 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 of the Web Part.

tp_Level: **Publishing Level Type** 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 (1) or not (0).

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 the Common Result Sets **List Metadata Result Set** (section 2.2.4.14).

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 the Common Result Sets **Empty Result Set** (section 2.2.4.10).

3.1.5.21.13 Event Receivers Result Set (1)

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 2.2.1.2.5) of 2 (List) for the List.

The **Event Receivers Result Set** is defined in the Common Result Sets **Event Receivers Result Set** (section 2.2.4.11).

3.1.5.21.14 Document Information and Content (Update) Metadata Result Set

This result set contains metadata about the document stream for the current version of the requested document.

This 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, this result set MUST contain one row; otherwise, this result set MUST contain no rows.

The value returned for the **{HistVersion}** column MUST be 0.

This result set is defined as **Document Content Metadata Result Set** (section 2.2.4.6) in the Common Result Sets (section 2.2.4).

3.1.5.21.15 Document Information and Content (Update) Stream Result Set

This result set contains a row for each of the document's **stream binary pieces**. This result set MUST only be returned if *@GetContent* is set to 1 and *@Version* is negative.

This result set MUST return zero or more rows. If the requested document is an uncustomized document, this result set MUST contain zero rows.

This result set is defined by the **Document Content Stream Result Set** (section 2.2.4.7) in the Common Result Sets (section 2.2.4).

3.1.5.21.16 Document Version Information and Content (Update) Metadata Result Set

This result set contains information about the metadata for the **document stream** for the requested version of the **document**.

This 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*.

This result set MUST contain one row if it is returned. The value returned for the **{HistVersion}** column MUST be the **user interface (UI) version** of the requested version of the document.

This result set is defined by the **Document Content Metadata Result Set** (section 2.2.4.6) in the Common Result Sets (section 2.2.4).

3.1.5.21.17 Document Version Information and Content (Update) Stream Result Set

This result set contains a row for each of the document's **stream binary pieces**.

This 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*.

This result set MUST contain zero or more rows.

This result set is defined by the **Document Content Stream Result Set** (section 2.2.4.7) in the Common Result Sets (section 2.2.4).

3.1.5.21.18 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 2.2.1.2.1) for more information.

{NeedManageListRight}: This specifies whether operations on this document require the user to have the **ManageLists** bit of the **WSS Rights Mask** set (section 2.2.2.15). 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 this procedure for the same List.

```
PROCEDURE proc_GenerateNextId (
      @SiteId                       uniqueidentifier,
      @ListId                       uniqueidentifier,
      @NumIds                       int = 1,
      @RequestGuid                  uniqueidentifier = NULL     OUTPUT
);
```

@SiteId: The Site Collection Identifier (section 2.2.1.1.9) of the **site collection** containing the List.

@ListId: The List Identifier (section 2.2.1.1.5) of the List.

@NumIds: **proc_GenerateNextId** MUST increment the value of the list item identifier 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: This procedure returns a positive integer return code, which is the available list item Identifier for the specified List. This 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** of the site collection containing the List

@ScopeId: The **scope identifier** of the security scope for which roles are to be returned.

@PrincipalIdsCsv: The **principal (1)** identifiers for which roles are to be returned. This parameter MUST be an **nvarchar(max)** which conforms to the following **ABNF**:

```
PrincipalIdsCsv = #PrincipalId
PrinicipalId = 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**.

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 (1)** 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** of the object.

@SiteId: The Site Collection Identifier 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** set (section 2.2.2.1) 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 the Common Result Sets Site **Audit Mask Result Set** (section 2.2.4.22).

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 2.2.1.2.2) of the object.

@SiteId: A **Site Collection Identifier** (section 2.2.1.1.9) 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** 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 (1)**, **site (2)**, 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_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 2.2.1.1.9) of the site collection containing the list.

@WebId: The Site Identifier (section 2.2.1.1.11) of the **site (2)** 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 (2) 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 (2) 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 in the following sections.

3.1.5.27.1 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 section **List Metadata Result Set** (section 2.2.4.14), which can be found in the Common Result Sets (section 2.2.4).

3.1.5.27.2 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 section **Empty Result Set** (section 2.2.4.10), which can be found in the Common Result Sets (section 2.2.4).

3.1.5.27.3 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 2.2.1.2.5) of 2 for the List.

This result set is defined in **Event Receivers Result Set** (section 2.2.4.11), which can be found in the Common Result Sets (section 2.2.4).

3.1.5.28 proc_GetContainingListCore

The **proc_GetContainingListCore** stored procedure is invoked to get metadata and event receiver information about the list containing a specified URL.

```
PROCEDURE proc_GetContainingListCore (
      @SiteId                       uniqueidentifier,
```

```
        @WebId                          uniqueidentifier,
        @Url                            nvarchar(260),
        @bReturnListMetadata            bit,
        @ListId                         uniqueidentifier OUTPUT,
        @BaseType                       int = NULL OUTPUT,
        @ExcludedType                   int = NULL OUTPUT,
        @ListFlags                      bigint = NULL OUTPUT,
        @bNeedNoThrottle                bit = NULL OUTPUT,
        @ItemCount                      int = NULL OUTPUT
);
```

@SiteId: The **Site Collection Identifier** (section 2.2.1.1.9) of the site collection containing the list.

@WebId: The **Site Identifier** (section 2.2.1.1.11) of the **site (2)** containing the list.

@Url: The store-relative form URL 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 *@Url* is ignored, since it could point to a nonexistent document or list item.

@bReturnListMetadata: Specifies whether the **List Metadata Result Set** (section 3.1.5.28.1) returned by this stored procedure contains any rows. When this parameter is set to 0, the result set MUST be empty.

@ListId: An output parameter that specifies the **List Identifier** (section 2.2.1.1.5) of the list that contains the specified URL. If the containing list was not found, this parameter MUST be NULL.

@BaseType: An output parameter that specifies the **List Base Type** (section 2.2.1.2.11) of the list that contains the specified URL. If the containing list was not found, this parameter MUST be NULL.

@ExcludedType: An output parameter that specifies whether the specified URL is within a special folder type in the containing list. The parameter MUST bet set to one the following values.

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: An output parameter that specifies the **List Flags** (section 2.2.2.5) of the list that contains the specified URL. If the containing list was not found, this parameter MUST be NULL.

@bNeedNoThrottle: An output parameter that specifies whether the list is exempt from resource throttling operations. If the containing list was not found, this parameter MUST be NULL.

@ItemCount: An output parameter that specifies the number of list items that are stored within the list that contains the specified URL. If the containing list was not found, this parameter MUST be NULL.

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 (2) 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 (2) that was specified by the *@WebId* parameter.

This stored procedure MUST return one result set, as defined following.

3.1.5.28.1 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 be returned if a containing list is found for the value for the *@Url* parameter and the *@bReturnListMetadata* parameter is set to 1. The **List Metadata Result Set** MUST contain one row.

This result set is defined in section **List Metadata Result Set** (section 2.2.4.14), which can be found in the Common Result Sets (section 2.2.4).

3.1.5.28.2 Empty Result Set

The **Empty Result Set** MUST be returned if a containing list is not found for the value in the *@Url* parameter, or if the *@bReturnListMetadata* parameter is set to 0.

This result set is defined in section **Empty Result Set** (section 2.2.4.10), which can be found in the Common Result Sets (section 2.2.4).

3.1.5.29 proc_GetDocsMetaInfo

The **proc_GetDocsMetaInfo** stored procedure is invoked to request **document** metadata information for up to ten documents within a specified **site (2)**.

```
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** for the **site collection** containing the specified documents.

@WebFullUrl: The **store-relative URL** of the site (2) 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 3.1.5.29.5).

@UserId: The user identifier 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 thru 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 2.2.1.2.1) 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 which MUST be 0.

The **proc_GetDocsMetaInfo** stored procedure MUST return multiple result sets:

3.1.5.29.1 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 2.2.4.16) MUST be returned instead. If returned, the **Individual URL Security Result Set** MUST contain a single row.

The result set is defined in the Common Result Sets **Individual URL Security Result Set** (section 2.2.4.12).

3.1.5.29.2 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 2.2.4.12) 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 the Common Result Sets **NULL Individual URL Security Result Set** (section 2.2.4.16).

3.1.5.29.3 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 the Common Result Sets **Server Time Result Set** (section 2.2.4.20).

3.1.5.29.4 Subsite List Result Set

The **Subsite List Result Set** contains an unordered list of store-relative form URLs for all subsites whose parent **site (2)** 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 the Common Result Sets **URL Result Set** (section 2.2.4.27).

3.1.5.29.5 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 the Common Result Sets **Single Doc Link Information Result Set** (section 2.2.4.21).

3.1.5.29.6 Multiple Document Metadata Result Set

The **Multiple Document Metadata Result Set** contains the metadata for the specified documents. This 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(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,
{bContentVersionIsDirty}    bit,
{NextBSN}                   bigint,
{StreamSchema}              byte,
{InternalVersion}           int
```

DocId: If the specified document exists, this field MUST be the **document identifier**; 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, it SHOULD be returned as **nvarchar(128)**. If it has an empty leaf name, 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 used for internal conflict detection that is incremented any time a change is made to this document. This value MUST be NULL if the document does not exist.

DocFlags: A **Doc Flags** value (section 2.2.2.3) describing the document. This value MUST be NULL if the document does not exist.

{ListType}: A combination of the **List Base Type** (section 2.2.1.2.11) and **List Server Template** (section 2.2.1.2.12) values of the associated **list (1)** 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 is not in a list (1).

tp_Name: The identifier of the list (1) that contains this document. This value MUST be NULL if the document does not exist or is not in a list (1).

{ListTitle}: If this document is the root folder of a list (1), this contains the display name of the list (1). This value MUST be NULL if the document does not exist or is not in a list (1).

{CacheParseId}: This value MUST be NULL.

GhostDirName: The directory name as passed to **proc_GetDocsMetaInfo** (section 3.1.5.29) 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, **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. This value MUST be NULL if the document does not exist or is currently checked in.

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, 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** value (section 2.2.1.2.17) 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 has not been processed by a virus scanner.

SetupPathVersion: For an **uncustomized** 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).
15	This is relative to the install location of Microsoft SharePoint Foundation 2013 on the front-end Web server (for example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\15).

SetupPath: For a document that is or has been uncustomized, this contains the setup path 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 if the document does not exist or was never uncustomized.

SetupPathUser: If this document is now or once was uncustomized, this contains the login name of the user that created the uncustomized document. This value MUST be NULL if the document does not exist or was never uncustomized.

NextToLastTimeModified: The value of **TimeLastModified** from the previous time when the document was last saved. This value MUST be NULL for documents that do not exist.

UIVersion: The **UI version** number of the document. This value MUST be NULL for documents that do not exist.

CheckinComment: An optional user-supplied description provided when a document is checked in or published. This value MUST be NULL for documents that do not exist or are 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. This value MUST be NULL if the document does not exist or a welcome page is not specified.

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 '#'. This value MUST be NULL if the document that do not exist or parameters are not specified.

tp_Flags: The **List Flags** value (section 2.2.2.5) for the list (1) that contains this document. This value MUST be NULL for documents that do not exist or are not in a list (1).

Acl: The binary serialization of the **WSS ACL Format** (section 2.2.3.6) **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 2.2.2.15) 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 2.2.1.1.13) of the user that published this document as a draft. This value MUST be NULL if the document does not exist or is not a draft version.

Level: A **Publishing Level Type** value (section 2.2.2.6) 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, 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 **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, 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 between the original and transformed document is broken. This value MUST be NULL if the document does not exist or is not the product of a document transformation.

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 editors 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 identifier for a row in a **document library** for this item. This value MUST be NULL if the document does not exist or is not contained in a list (1).

tp_DefaultWorkflowId: The **workflow identifier** 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. This value MUST be NULL if the document does not exist or is not contained in a list (1) with a configured approval workflow.

ListId: The **list identifier** of the list (1) that contains the document. This value MUST be NULL if the document does not exist or is not contained in a list (1).

ItemChildCount: The number of non-folder children of the document.

FolderChildCount: The number of folder children of the document.

MetaInfoVersion: A counter used for internal conflict detection that is incremented any time a change is made to the **MetaInfo** for the document.

{CurVerMetaInfo}: This value MUST be NULL.

ContentVersion: A counter used for internal conflict detection that is incremented any time a change is made to the binary contents of this document. 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 zero; otherwise, the value MUST be one.

{NextBSN}: The current **BLOB** sequence number of the document being returned.

{StreamSchema}: The current **stream schema** of the document being returned.

{InternalVersion}: The **internal version number** of the document being returned.

3.1.5.30 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 2.2.1.1.9) 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.

3.1.5.30.1 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 section **Link Information Result Set** (section 2.2.4.13), located in the Common Result Sets (section 2.2.4).

3.1.5.31 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 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.

3.1.5.31.1 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 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.

3.1.5.32 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(
      @SiteId                   uniqueidentifier,
      @WebId                    uniqueidentifier,
      @ListId                   uniqueidentifier,
      @RequestGuid              uniqueidentifier = NULL     OUTPUT
);
```

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

@WebId: The **site identifier** of the **site (2)** containing the list.

@ListId: The **list identifier** of the list.

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

Return Values: This procedure returns an integer return code which MUST be 0. This procedure MUST return a single result set as follows.

3.1.5.32.1 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 **Compressed** structure. 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 2.2.7.3.5.

3.1.5.33 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 2.2.1.1.9) for a site collection that contains the **site (2)** specified by the *@WebId* parameter.

@WebId: The **Site Identifier** (section 2.2.1.1.11) for the site (2) containing the list specified by the *@ListId* parameter.

@ListId: The **List Identifier** (section 2.2.1.1.5) for the list.

@PrefetchListScope: Specifies whether to include the **Unique Permissions Result Set** (section 3.1.5.33.2) or **NULL Unique Permissions Result Set** (section 3.1.5.33.3) 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 2.2.4.28) (located in the Common Result Sets (section 2.2.4)) MUST be returned.

@PrefetchWebParts: Specifies whether to include the **List Web Parts Result Set** (section 3.1.5.33.5) 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 2.2.1.1.13) 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 order defined in the following sections.

3.1.5.33.1 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 2.2.1.1.5) 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 section **List Metadata Result Set** (section 2.2.4.14), located in the Common Result Sets (section 2.2.4).

3.1.5.33.2 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** 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 section **Unique Permissions Result Set** (section 2.2.4.26), located in the Common Result Sets (section 2.2.4).

3.1.5.33.3 NULL Unique Permissions Result Set

The **NULL Unique Permissions Result Se**t 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 section **NULL Unique Permissions Result Set** (section 2.2.4.17), located in the Common Result Sets (section 2.2.4).

3.1.5.33.4 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 section **Event Receivers Result Set** (section 2.2.4.11), located in the Common Result Sets (section 2.2.4).

3.1.5.33.5 List Web Parts Result Set

The **List Web Parts Result Set** contains information about the list Web Parts defined for this **site (2)**. 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 section **List Web Parts Result Set** (section 2.2.4.15), located in the Common Result Sets (section 2.2.4).

3.1.5.33.6 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 section **List Related Fields Result Set** (section 2.2.4.28), located in the Common Result Sets (section 2.2.4).

3.1.5.34 proc_getObject

The **proc_getObject** stored procedure is invoked to retrieve a single Configuration Object (section 2.2.5.1) from the Configuration Database (section 3.1.1). The **proc_getObject** 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.

3.1.5.34.1 Object Result Set

The **Object Result Set** returns the Configuration Object (section 2.2.5.1) 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.

ParentId: Contains the GUID for the **Parent** (section 2.2.5.1.4) of the Configuration Object.

ClassId: Contains the GUID for the **Class** (section 2.2.5.1.1) of the Configuration Object.

Name: Contains the **Name** (section 2.2.5.1.3) of the requested Configuration Object.

Status: Contains the **Status** (section 2.2.5.1.5) of the Configuration Object.

Version: Contains the **Version** (section 2.2.5.1.6) of the Configuration Object.

Properties: Contains the properties definition of the Configuration Object, which is opaque to the back-end database server.

3.1.5.35 proc_getObjectsByBaseClass

The **proc_getObjectsByBaseClass** stored procedure is invoked to return a list of **GUIDs** for child Configuration Objects (section 2.2.5.1) of the specified parent Configuration Object that inherit from the specified base **Class** (section 2.2.5.1.1). 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 2.2.5.1.4) 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.

3.1.5.35.1 Object ID Result Set

The **Object ID Result Set** contains the **GUIDs** of the matching Configuration Objects (section 2.2.5.1), 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 the Common Result Sets **Object ID Result Set** (section 2.2.4.18).

3.1.5.36 proc_getObjectsByClass

The **proc_getObjectsByClass** stored procedure is invoked to retrieve the **GUIDs** of Configuration Objects (section 2.2.5.1) based upon the **Class** (section 2.2.5.1.1) type, the parent Configuration Object, and the **Name** (section 2.2.5.1.3) 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** (section 2.2.5.1.4) 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.

3.1.5.36.1 Object ID Result Set

The **Object ID Result Set** contains the **GUIDs** of the matching Configuration Object (section 2.2.5.1), 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 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 the Common Result Sets **Object ID Result Set** (section 2.2.4.18).

3.1.5.37 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 2.2.1.1.9) 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.

3.1.5.37.1 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**.

3.1.5.38 proc_getSiteMap

The **proc_getSiteMap** **stored procedure** is invoked to determine the **Site Map** (section 3.1.5.38.1) 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**.

3.1.5.38.1 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,
      SubscriptionName              nvarchar(48),
      AppSiteDomainId               varchar(6);
```

Id: Contains the **Site Collection Identifier** (section 2.2.1.1.9) 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 an 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.

SubscriptionName: The name corresponding to the **site subscription identifier** of the implementation-specific subscription for the requested **site collection**.

AppSiteDomainId: The **app site domain identifier** of the site collection.

3.1.5.39 proc_getSiteMapById

The **proc_getSiteMapById** **stored procedure** is invoked to determine complete Site Map (section 3.1.5.38.1) 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 2.2.1.1.9) of the site collection to get Site Map (section 3.1.5.38.1) 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**.

3.1.5.39.1 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 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,
      SubscriptionName            nvarchar(48),
      AppSiteDomainId             varchar(6);
```

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 2.2.1.1.9) 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 (2)** 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 an 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.

SubscriptionName: The name corresponding to the **site subscription identifier** of the implementation-specific subscription for the requested **site collection**.

AppSiteDomainId: The **app site domain identifier** of site collection.

3.1.5.40 proc_GetTpWebMetaDataAndListMetaData

The **proc_GetTpWebMetaDataAndListMetaData** **stored procedure** retrieves metadata for a particular **site (2)** or **list (1)**.

```
PROCEDURE proc_GetTpWebMetaDataAndListMetaData(
      @WebSiteId                    uniqueidentifier,
      @WebId                        uniqueidentifier,
      @Url                          nvarchar(260),
      @ListId                       uniqueidentifier,
      @ViewId                       uniqueidentifier,
      @RunUrlToWebUrl               bit,
      @DGCacheVersion               bigint,
      @SystemId                     varbinary(512)     = NULL,
      @AppPrincipalName             nvarchar(256),
      @IsHostHeaderAppPrincipalName bit,
      @MetadataFlags                int                = 0,
      @ThresholdScopeCount          int                = 0,
      @CurrentFolderUrl             nvarchar(260)      = NULL,
      @RequestGuid                  uniqueidentifier   = NULL OUTPUT
);
```

@WebSiteId: The **site collection identifier** for a **site collection** containing the site (2).

@WebId: The **site identifier** of the site (2) 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 (2) or list to retrieve metadata for.

@ListId: The **list identifier** 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 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 2.2.4.15) (found in the Common Result Sets (section 2.2.4)) will be set to NULL.

@RunUrlToWebUrl: A bit flag specifying whether the *@URL* parameter is for a site (2) 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 (2).

@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 2.2.4.4) 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 2.2.4.3) and the **Domain Group Cache WFE Update Result Set** (section 2.2.4.5) MUST NOT be returned.

@SystemId: The **SystemId** of the current user requesting the metadata.

@AppPrincipalName: The **app principal** identifier or the **app web domain identifier** associated with the current user. If the current user is not associated with an app principal, this parameter MUST be NULL.

@IsHostHeaderAppPrincipalName: If this parameter is set to 1, the *@AppPrincipalName* parameter MUST be the app principal identifier, else it MUST be the app web domain identifier.

@MetadataFlags: A bit field used to determine the types of metadata returned by this procedure. The only valid values of the *@MetadataFlags* 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** (section 3.1.5.40.21).
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 3.1.5.40.18) 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 2.2.6.1).
METADATA_WEB,	Retrieve information pertaining to the provided site (2)

Symbolic name, Value	Description
0x00000010	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 (2) identified by *@WebSiteId*, the **Document Metadata Result Set** (section 2.2.4.8) 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 2.2.4.14), **List Unique Permissions Result Set** (section 3.1.5.40.18), **Event Receivers Result Set** (section 2.2.4.11), and the **List Web Parts Result Set** (section 3.1.5.40.21).
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 does not match a list identifier contained within the site (2) specified by *@WebSiteId*, returns the **Event Receivers Result Set** (section 3.1.5.40.20) and the **List Web Parts Result Set** (section 3.1.5.40.21).
METADATA_WEB_PROP, 0x00000100	Return site data. Setting this flag in combination with **METADATA_WEB** will retrieve the **Site Metainfo Result Set** (section 2.2.4.25).
METADATA_WEB_FEATURES, 0x00000200	Return site features. Setting this flag in combination with **METADATA_WEB** will retrieve the **Site Feature List Result Set** (section 2.2.4.23).
METADATA_WEB_NAVSTRUCT, 0x00000400	Return navigation structure. Setting this flag in combination with **METADATA_WEB** will retrieve the **Site Unique Permissions Result Set** (section 3.1.5.40.10).
METADATA_LISTMETADATA_NOFETCH, 0x00000800	Do not return list metadata. Setting this flag will suppress the retrieval of the **List Metadata Result Set** (section 3.1.5.40.16), **Unique Permissions Result Set** (section 3.1.5.40.23), and the **Event Receivers Result Set** (section 3.1.5.40.20) 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** (section 3.1.5.40.16), **Unique Permissions Result Set** (section 2.2.4.26), **Event Receivers Result Set** (section 3.1.5.40.20), and the **List Web Parts Result Set** (section 3.1.5.40.21) using the User identification parameters.
METADATA_URL_REALLY, 0x00002000	Return single document metadata. Setting this flag will return the **Document Metadata Result Set** (section 3.1.5.40.27).
METADATA_WEB_WELCOMEPAGE, 0x00004000	Redirect welcome page. Setting this flag in combination with **METADATA_WEB** will retrieve the **Redirect URL Result Set** (section 3.1.5.40.13).
METADATA_PREFETCH_RELATEDFIELDS, 0x00008000	Return the **List Related Fields Result Set** (section 2.2.4.28) 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** (section 3.1.5.40.18).

@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 (2) is blocked because the site collection is locked.

Result Sets: The **proc_GetTpWebMetaDataAndListMetaData** stored procedure returns up to 30 **result sets**, depending on the state of the input parameters.

3.1.5.40.1 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**, 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.

3.1.5.40.2 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 (2)** 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 (2), or if *@RunUrlToWebUrl* is set to 1, a valid location within a site (2), 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 the Common Result Sets **Domain Group Cache Versions Result Set** (section 2.2.4.4).

3.1.5.40.3 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**.

The **Domain Group Cache BEDS Update Result Set** schema is defined in the Common Result Sets **Domain Group Cache BEDS Update Result Set** (section 2.2.4.3).

3.1.5.40.4 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**. 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 the Common Result Sets **Domain Group Cache WFE Update Result Set** (section 2.2.4.5).

3.1.5.40.5 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** is returned, and it MUST contain one row.

The **Site Metadata Result Set** is defined in the Common Result Sets **Site Metadata Result Set** (section 2.2.4.24).

3.1.5.40.6 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 2.2.1.2.5) of 0 registered for the site collection specified by the *@WebSiteId* parameter.

The **Site Collection Event Receivers Result Set** is defined in **Event Receivers Result Set** (section 2.2.4.11), located in the Common Result Sets (section 2.2.4).

3.1.5.40.7 Site Event Receivers Result Set

The **Event Receivers Result Set** contains information about the event receivers defined for this **site (2)**. 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 (2). 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 the Common Result Sets **Event Receivers Result Set** (section 2.2.4.11).

3.1.5.40.8 Site MetaInfo Result Set

The **Site MetaInfo Result Set** contains the metadict of the **site (2)**. 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_W**EB flag set and the **METADATA_WEB_PROP** flag set, and the site (2) 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 the Common Result Sets **Site MetaInfo Result Set** (section 2.2.4.25).

3.1.5.40.9 Site Feature List Result Set

The **Site Feature List Result Set** contains the list of **Feature Identifiers** (section 2.2.1.1.4) of the **site (2)**. 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 (2).

The **Site Feature List Result Set** is defined in the Common Result Sets **Site Feature List Result Set** (section 2.2.4.23).

3.1.5.40.10 Site Unique Permissions Result Set

The **Unique Permissions Result Set** contains security scope and **WSS ACL** (section 2.2.3.6) information of the **site (2)** 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 (2) contains cached navigation scope information.

The **Unique Permissions Result Set** is defined in the Common Result Sets **Unique Permissions Result Set** (section 2.2.4.26).

3.1.5.40.11 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 (2)** 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 the Common Result Sets **NULL Unique Permissions Result Set** (section 2.2.4.17).

3.1.5.40.12 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 (2)** specified by *@WebId* does not contain cached navigation scope information, or the site is marked as **dirty**.

The **Empty Result Set** is defined in the Common Result Sets **Empty Result Set** (section 2.2.4.10).

3.1.5.40.13 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 2.2.1.2.15) 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** 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).

3.1.5.40.14 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 2.2.1.2.15) 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.

3.1.5.40.15 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 (2) 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 (2)** and contains the file, list or list item specified by *@Url*

```
{ListIdSelected}              uniqueidentifier,
{TypeSelected}                int,
{ItemIdSelected}              int;
```

{ListIdSelected}: The **List Identifier** 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 2.2.1.2.15).

{ItemIdSelected}: **Document Library Row Identifier** of the list item taken from the **Docs View** (section 2.2.6.3) for the list. This value MUST be NULL if *@Url* does not specify a list item.

3.1.5.40.16 List Metadata Result Set

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 (2)** containing the specified list.

The **List Metadata Result Set** is defined in the Common Result Sets **List Metadata Result Set** (section 2.2.4.14).

3.1.5.40.17 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 2.2.4.28), located in the Common Result Sets (section 2.2.4).

3.1.5.40.18 List Unique Permissions Result Set

The **Unique Permissions Result Set** contains all security scope and **WSS ACL** (section 2.2.3.6) information from the specified list. The **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 and
- *@MetadataFlags* has the **METADATA_LISTMETADATA_NOFETCH** flag not set

The **Unique Permissions Result Set** is defined in **Unique Permissions Result Set** (section 2.2.4.26).

3.1.5.40.19 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 **Unique Permissions Result Set** (section 2.2.4.26).

3.1.5.40.20 Event Receivers Result Set

The **Site Event Receivers Result Set** contains information about the event receivers defined for the **site (2)** 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 2.2.1.2.5) of 2 registered for the site (2) specified by *@WebId*.

The **Site Event Receivers Result Set** is defined in **aa** (section 2.2.4.11), located in the Common Result Sets (section 2.2.4).

3.1.5.40.21 List Web Parts Result Set

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 the Common Result Sets **List Web Parts Result Set** (section 2.2.4.15).

3.1.5.40.22 List Metadata Result Set (1)

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** and the site collection has a user information list
- The **List Identifier** 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 the Common Result Sets List **Metadata Result Set** (section 2.2.4.14).

3.1.5.40.23 Unique Permissions Result Set

The **Unique Permissions Result Set** contains all security scope and **WSS ACL** (section 2.2.3.6) 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** and the site collection has a user information list
- The **List Identifier** 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 2.2.4.26).

3.1.5.40.24 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** and the site collection has a user information list
- The **List Identifier** 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 (2)** of the site collection.

The **NULL Unique Permissions Result Set** is defined in the in the Common Result Sets **NULL Unique Permissions Result Set** (section 2.2.4.17).

3.1.5.40.25 Event Receivers Result Set (1)

The **Event Receivers Result Set** contains information about the event receivers defined for the **site (2)** 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** and the site collection has a user information list
- The **List Identifier** 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 2.2.1.2.5) of 2 (List) that is registered for the root site (2) of the site collection specified by *@WebSiteId*.

The **Event Receivers Result Set** is defined in the Common Result Sets **Event Receivers Result Set** (section 2.2.4.11).

3.1.5.40.26 List Web Parts Result Set (1)

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** and the site collection has a user information list
- The **List Identifier** 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 the Common Result Sets **List Web Parts Result Set** (section 2.2.4.15).

3.1.5.40.27 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 the Common Result Sets **Document Metadata Result Set** (section 2.2.4.8).

3.1.5.40.28 NULL Result Set

The **NULL Result Set** returns no data. The **NULL Result Set** MUST be returned if the **Document Metadata Result Set** 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.

3.1.5.40.29 App Principal Information Result Set

If the *@AppPrincipalName* parameter is not NULL, the **App Principal Info Result Set** MUST be returned as defined in section 2.2.4.29.

3.1.5.40.30 App Principal Permissions Result Set

The **App Principal Permissions Result Set** is returned to indicate what permissions the specified **app principal** has for the **site (2)** or **list (1)** as specified in section 2.2.4.30. If the specified app principal has permissions for the specified site (2) or list (1), the **App Principal Permissions Result Set** MUST be returned.

3.1.5.41 proc_GetUniqueScopesInList

The **proc_GetUniqueScopesInList** stored procedure is invoked to get the **WSS ACL Format** (section 2.2.3.6) 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** of the site collection containing the list.

@WebId: The **Site Identifier** of the **site (2)** that contains the list.

@ListId: The **List Identifier** 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 2.2.4.26), 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.

3.1.5.41.1 Unique Permissions Result Set

The **Unique Permissions Result Set** returns the **WSS ACL Format** (section 2.2.3.6) 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 the Common Result Sets **Unique Permissions Result Set** (section 2.2.4.26).

3.1.5.41.2 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 the Common Result Sets **NULL Unique Permissions Result Set** (section 2.2.4.17).

3.1.5.42 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 that matches 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 MUST return value 0.

The **front-end Web server** can define one or more version numbers. Each of the version numbers is identified by a version identifier. 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 2.2.6.12). When the front-end Web server connects to a back-end database server, the front-end Web server SHOULD retrieve the version identifiers using stored procedure **proc_GetVersion**<6> and make sure the versions are within an acceptable range defined by the front-end Web server. Otherwise, the front-end Web server MUST NOT connect to the back-end database server.

The version identifiers and the acceptable version numbers for this protocol that are 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 identifiers 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. 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 that 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**.

3.1.5.43 proc_GetWebMetaInfo

The **proc_GetWebMetaInfo** **stored procedure** is invoked to request metadata information for a **site (2)**.

```
PROCEDURE proc_GetWebMetaInfo(
      @WebSiteId                    uniqueidentifier,
      @WebDirName                   nvarchar(256),
      @WebLeafName                  nvarchar(128),
      @DGCacheVersion               bigint,
      @SystemId                     varbinary(512) = NULL,
      @AppPrincipalName             nvarchar(256),
      @IsHostHeaderAppPrincipalName bit,
      @RequestGuid                  uniqueidentifier = NULL OUTPUT
);
```

@WebSiteId: The **site collection identifier** for the **site collection** containing the site (2) 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 3.1.5.43.2) 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 3.1.5.43.3) and the **Domain Group Cache WFE Update Result Set** (section 3.1.5.43.4) MUST NOT be returned.

@SystemId: The **SystemID** of the current user requesting the site metadata information.

@AppPrincipalName: The **app principal** identifier or the **application web domain Id** associated with the current user. If the current user is not associated with an application principal, this parameter MUST be NULL.

@IsHostHeaderAppPrincipalName: If this parameter is set to "1", the *@AppPrincipalName* parameter MUST be the app principal identifier; otherwise, it MUST be the application web domain Id.

@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 (2) was not found at the specified location.
1271	The site collection is locked for read and write operations.

Result Sets: This stored procedure MUST return **result sets** as described in the following sections: **Site MetaInfo Result Set** (section 3.1.5.43.1), **Domain Group Cache Versions Result Set** (section 3.1.5.43.2), **Domain Group Cache BEDS Update Result Set** (section 3.1.5.43.3), **Domain Group Cache WFE Update Result Set** (section 3.1.5.43.4), **Site Metadata Result Set** (section 3.1.5.43.5), and **Event Receivers Result Set** (section 3.1.5.43.6).

3.1.5.43.1 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 (2) 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 section **Site MetaInfo Result Set** (section 2.2.4.25), located in the Common Result Sets (section 2.2.4).

3.1.5.43.2 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 (2)** 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 2.2.4.4), located in the Common Result Sets (section 2.2.4).

3.1.5.43.3 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 3.1.5.43.2) 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 2.2.1.1.13) of the domain groups.

This result set is defined in section **Domain Group Cache BEDS Update Result Set** (section 2.2.4.3), located in the Common Result Sets (section 2.2.4).

3.1.5.43.4 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** 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 section **Domain Group Cache WFE Update Result Set** (section 2.2.4.5), located in the Common Result Sets (section 2.2.4).

3.1.5.43.5 Site Metadata Result Set

The **Site Metadata Result Set** contains metadata for the specified **site (2)**.

The **Site Metadata Result Set** MUST be returned if the specified site (2) is found, and MUST contain a single row corresponding to the specified site. If the site (2) 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 section **Site Metadata Result Set** (section 2.2.4.24), located in the Common Result Sets (section 2.2.4).

3.1.5.43.6 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 (2). The **Event Receivers Result Set** MUST be returned on successful execution.

The **Event Receivers Result Set** is defined in section **Event Receivers Result Set** (section 2.2.4.11), located in the Common Result Sets (section 2.2.4).

3.1.5.44 proc_GetWebMetainfoByUrl

The **proc_GetWebMetainfoByUrl** **stored procedure** is invoked to request site metadata information for the **site (2)** containing a specified **Uniform Resource Locator (URL)**.

```
PROCEDURE proc GetWebMetainfoByUrl(
      @WebSiteId                    uniqueidentifier,
      @Url                          nvarchar(260),
      @DGCacheVersion               bigint,
      @SystemId                     varbinary(512)    = NULL,
      @AppPrincipalName             nvarchar(256),
      @IsHostHeaderAppPrincipalName bit,
      @RequestGuid                  uniqueidentifier = NULL OUTPUT
);
```

@WebSiteId: The **Site Collection Identifier** (section 2.2.1.1.9) for the **site collection** containing the site (2) or **document** whose metadata is requested.

@Url: A URL in store-relative form for the site (2) or for a document within the site (2).

@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 3.1.5.44.3) 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 3.1.5.44.4) and the **Domain Group Cache WFE Update Result Set** (section 3.1.5.44.5) MUST NOT be returned.

@SystemId: The **SystemID** of the current user requesting the site metadata information.

@AppPrincipalName: The **app principal** identifier or the **application web domain Id** associated with the current user. If the current user is not associated with an application principal, this parameter MUST be NULL.

@IsHostHeaderAppPrincipalName: If this parameter is set to "1", the *@AppPrincipalName* parameter MUST be the app principal identifier; otherwise, it MUST be the application web domain Id.

@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 (2) 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 sections.

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

```
{WebUrl}                    nvarchar(260),
{AppWebDomainId}            varchar(8);
```

{WebUrl}: The **URL** in store-relative form of the **site (2)** that contains the URL specified in the *@Url* parameter.

{AppWebDomainId}: The **application web domain Id** associated with the site (2) that contains the URL specified in the *@Url* parameter. This MAY be NULL.

3.1.5.44.2 Site Metainfo Result Set

The **Site MetaInfo Result Set** MUST contain a single row containing the metainfo for the **site (2)**.

For details about the **Site MetaInfo Result Set**, see section 2.2.4.25.

3.1.5.44.3 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 2.2.4.4.

3.1.5.44.4 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 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**. 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 that is a member of a site permission level in the **site collection**, ordered by the **User Identifier** (section 2.2.1.1.13) of the domain groups.

This result set is defined in section 2.2.4.3.

3.1.5.44.5 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 (2)** 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**. 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 section 2.2.4.5.

3.1.5.44.6 Site Metadata Result Set

The **Site Metadata Result Set** contains specialized site metadata. If the **site (2)** 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 section 2.2.4.24.

3.1.5.44.7 Event Receivers Result Set

The **Event Receivers Result Set** contains information about the **event receivers** defined for this **site (2)**. There MUST be one row in the **Event Receivers Result Set** for each event receiver registered for this site (2). The **Event Receivers Result Set** MUST be returned on successful completion.

The **Event Receivers Result Set** is defined in section 2.2.4.11.

3.1.5.45 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 2.2.1.1.9) of the site collection containing the document.

@DocWebId: The **Site Identifier** (section 2.2.1.1.11) of the **site (2)** 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 3.1.5.45.3) 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 2.2.1.1.13) 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.

3.1.5.45.1 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 3.1.5.45.2) 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 section **Individual URL Security Result Set** (section 2.2.4.12), located in the Common Result Sets (section 2.2.4).

3.1.5.45.2 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 section **NULL Individual URL Security Result Set** (section 2.2.4.16), located in the Common Result Sets (section 2.2.4).

3.1.5.45.3 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,
VersionNumber                     int,
Size                        int,
CheckinComment              nvarchar(1023),
MetaInfo                    varbinary(max),
{IsTip}                     bit,
Level                       tinyint,
DraftOwnerId                int,
StreamSchema                tinyint,
{NextBSN}                   bigint;
```

TimeCreated: A timestamp in **UTC** format specifying when this document was created.

VersionNumber: 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** for more information regarding the different publishing levels.

StreamSchema: The current stream schema of the document being returned.

{NextBSN}: The current **BLOB** sequence number of the document being returned.

3.1.5.46 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**.

3.1.5.46.1 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.

3.1.5.47 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 2.2.1.1.9) of the **site collection** containing the document specified by a URL.

@DirWebId: The **Site Identifier** (section 2.2.1.1.11) of the **site (2)** containing the document.

@DirFullUrl: The URL in **store-relative form** specifying the document.

@AttachmentsFlag: An **Attachments Flag** (section 2.2.1.2.1) 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 in the following sections for the specific effects of this parameter value.

@FetchLinkInfo: A bit flag specifying whether to return the **Link Info Result Set** (section 3.1.5.47.6). 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 3.1.5.47.7). 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 2.2.1.1.13) 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 (2).
36	The stored procedure failed to complete successfully. This value is returned if the *@ItemLimit* parameter is greater than 0, this **list (1)** is not exempt from resource throttling operations, and the number of list items combined with the number of folders in the list (1) exceeds the *@ItemLimit* parameter.

This stored procedure MUST return two to six result sets, as described in the following sections, in the following order.

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 (1)** or **document library**, security information is returned from the document's parent object, such as the list (1) 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 (1) or document library. Otherwise, the **NULL Individual URL Security Result Set** MUST be returned instead. If returned, the **Individual URL Security Result Set** MUST contain a single row.

The result set is defined in **Individual URL Security Result Set** (section 2.2.4.12), located in the Common Result Sets (section 2.2.4).

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 (1)** or **document library**. The **NULL Individual URL Security Result Set** MUST only be returned if the document URL is not contained within a list (1) or document library.

The result set MUST return a single row and is defined in **NULL Individual URL Security Result Set** (section 2.2.4.16), located in the Common Result Sets (section 2.2.4).

3.1.5.47.3 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 result set is defined in **Server Time Result Set** (section 2.2.4.20), located in the Common Result Sets (section 2.2.4).

3.1.5.47.4 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 (2)** 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 (2), and it MUST contain no rows if there are no such subsites.

The result set is defined in **URL Result Set** (section 2.2.4.27), located in the Common Result Sets (section 2.2.4).

3.1.5.47.5 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,
{NextBSN}                   bigint,
{StreamSchema}              tinyint,
{InternalVersion}           int
```

Id: The **Document Identifier** (section 2.2.1.1.2) 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 used for internal conflict detection that is incremented any time a change is made to the document.

DocFlags: A **Doc Flags** value (section 2.2.2.3) 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 2.2.1.2.11) and **List Server Template** (section 2.2.1.2.12) 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 2.2.1.1.5) of the **list (1)** containing this document.

{ListTitle}: If the document URL is the **root folder** of a list (1), this contains the **display name** of the list (1). 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 2.2.1.2.17) 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 an **uncustomized** 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	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).
15	The **SetupPath** is relative to the install location of Microsoft SharePoint Foundation 2013 on the front-end Web server (for example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\15).

SetupPath: If the document is now or once was uncustomized, this contains the setup path fragment relative to the base setup path described above by the **SetupPathVersion** value where the document stream of the document can be found. This value can be NULL.

SetupPathUser: If the document is now or once was uncustomized, this contains the login name of the user who created the uncustomized 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, 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 **UI 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** value (section 2.2.2.5) for the list (1) that contains the document.

Acl: The binary serialization of the **WSS ACL Format** (section 2.2.3.6) 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 2.2.2.15) 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 2.2.1.1.13) 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** value (section 2.2.2.6) specifying the publishing status of the document.

ParentVersion: If the document is a transformed version of another document, this is the 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. 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 editors for .**HTML** or .**XML** files). This value MUST be NULL if a **ProgId** was not specified when the document was saved.

DoclibRowId: The identifier for a row in a **document library** for the document. If the document is not contained in a list (1), this value MUST be NULL.

tp_DefaultWorkflowId: The **Workflow Identifier** (section 2.2.1.1.16) corresponding to the **workflow** to be invoked if the document is in a moderated list (1) and the document is submitted for approval as part of a check in.

ListId: The **list identifier** of the list (1) that contains the document. If the document is not contained in a list (1), this value MUST be NULL.

ItemChildCount: The value MUST be the number of **list items** for the list (1) that contains the document.

FolderChildCount: The value MUST be the number of folders that exist in the list (1) that contains the document.

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

{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 zero.

{NextBSN}: The current **BLOB** sequence number of the document being returned.

{StreamSchema}: The current **stream schema** of the document being returned.

{InternalVersion}: The **internal version number** of the document being returned.

3.1.5.47.6 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 section 2.2.4.21, located in the Common Result Sets (section 2.2.4).

3.1.5.47.7 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 that matches the following conditions:

- It belongs to **site collection** whose directory name is equal to the value of the *@DirFullUrl* parameter.
- Either:
  - It is a published or draft version which is not checked out to the specified user.
  - Or, it is a checked out version which is checked out to the user.

- Either:
  - The contained document has a **Document Store Type** (section 2.2.2.4) of zero and the *@IncludeListItems* parameter is set to one, or 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 one.
- Either:
  - The **app web domain identifier** of the **site (2)** containing the document is equal to the app web domain identifier of the site (2) whose identifier is equal to *@DirWebId* parameter.
  - Or, both site (2) containing the document and the site (2) whose identifier is equal to *@DirWebId* parameter, don't have app web domain identifier.

```
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,
{NextBSN}                   bigint,
{StreamSchema}              tinyint,
```

```
{InternalVersion}              int
```

Id: The **Document Identifier** (section 2.2.1.1.2) of this contained document.

{FullUrl}: The **URL** in **store-relative form** for this document.

Type: The **Document Store Type** 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 used for internal conflict detection that isincremented any time a change is made to the document.

DocFlags: A **Doc Flags** value (section 2.2.2.3) 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 2.2.1.2.11) and **List Server Template** (section 2.2.1.2.12) values of the **list (1)** 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 2.2.1.1.5) of the list (1) containing this document.

{ListTitle}: If the containing document is the root folder of the list (1) which contains this document, this contains the display name of the list (1). 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, 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, 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** value (section 2.2.1.2.17) 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 an **uncustomized** 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	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).
15	The **SetupPath** is relative to the install location of Microsoft SharePoint Foundation 2013 on the front-end Web server (for example, Program Files\Common Files\Microsoft Shared\Web Server Extensions\15).

SetupPath: If the document is now or once was uncustomized, this contains the setup path fragment relative to the base setup path described above by the **SetupPathVersion** value, where the document stream of the document can be found. This value can be NULL.

SetupPathUser: If the document is now or once was uncustomized, this contains the login name of the user who created the uncustomized 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, 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 **UI 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** value (section 2.2.2.5) for the list (1) that contains the document.

Acl: The binary serialization of the **WSS ACL Format** (section 2.2.3.6) 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 2.2.2.15) 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 2.2.1.1.13) 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** value (section 2.2.2.6) specifying the publishing status of the document.

ParentVersion: If the document is a transformed version of another document, this is the 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. 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 editors for .**HTML** or .**XML** files). This value MUST be NULL if a **ProgId** was not specified when the document was saved.

DoclibRowId: The identifier for a row in a document library for the document. If the document is not contained in a list (1), this value MUST be NULL.

tp_DefaultWorkflowId: The **Workflow Identifier** (section 2.2.1.1.16) corresponding to the **workflow** to be invoked if the document is in a moderated list (1) and the document is submitted for approval as part of a check in.

ListId: The **List Identifier** (section 2.2.1.1.5) of the list (1) that contains the document. If the document is not contained in a list (1), this value MUST be NULL.

ItemChildCount: The number of **list items** for the list (1) that contains the document if the document is contained in a list (1).

FolderChildCount: The number of folders that exist in the list (1) that contains the document if the document is contained in a list (1).

MetaInfoVersion: A counter used for internal conflict detection that isincremented any time a change is made to the **MetaInfo** for this document.

{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 zero.

{NextBSN}: The current **BLOB** sequence number of the document being returned.

{StreamSchema}: The current **stream schema** of the document being returned.

{InternalVersion}: The **internal version number** of the document being returned.

3.1.5.48 proc_RenameUrl

The **proc_RenameUrl** stored procedure renames a document or folder within a specified **site (2)**.

```
PROCEDURE proc_RenameUrl(
      @SiteId                     uniqueidentifier,
      @SubWebId                   uniqueidentifier,
      @OldUrl                     nvarchar(260),
      @NewUrl                     nvarchar(260),
      @UserId                     int,
      @AppPrincipalId             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 2.2.1.1.9) of the **site collection** containing the document or folder.

@SubWebId: The **site identifier** of the site (2) 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 of the user who is requesting the move.

@AppPrincipalId: The **app principal** identifier of the **app** that is requesting the operation. If the operation was not requested by an app then it MUST be 0.

@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 2.2.1.2.1) 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 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 (2).
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 (2), 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 (2), 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.

Value	Description
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.

3.1.5.48.1 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.

3.1.5.48.2 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.

3.1.5.49 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 of the site collection containing the role and security principal (2).

@WebId: The site identifier of the **site (2)** containing the role and security principal.

@ScopeId: The scope identifier 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 for the security principal (2) 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 (2). 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 (2) 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.

3.1.5.49.1 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 (2)**. 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 the Common Result Sets **Site Audit Mask Result Set** (section 2.2.4.22).

3.1.5.50 proc_SecAddRoleDef

The **proc_SecAddRoleDef** stored procedure creates a new role definition for a specified **site (2)** 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 of the site collection to add the role definition to.

@WebId: The site identifier of the site (2) to add the role definition to. The site (2) 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 (2), 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 value for the role definition to be added.

@PermMask: A **WSS Rights Mask** (section 2.2.2.15) 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 (2) has been reached.

Upon successful execution, the **proc_SecAddRoleDef** stored procedure MUST return a single result set.

3.1.5.50.1 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 (2)**. 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 the Common Result Sets **Site Audit Mask Result Set** (section 2.2.4.22).

3.1.5.51 proc_SecAddUser

The **proc_SecAddUser** stored procedure is invoked to add a new entry for a **security principal** to the UserInfo Table (section 2.2.6.10) that contains descriptive properties and security information about security principals (2), 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,
      @AccountType                  int,
      @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 of the site collection to associate with the **principal (1)**.

@SystemId: The **SystemID** 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 2.2.3.2) 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.

@AccountType: The account type of user. If *@SystemId* isn't null or NULLSID (value is S-1-0-0, which represents a **group (2)** with no members), and *@AccountType* is 2, then the added user's User Identifier represents an app account identifier (1073741822). All other values will be ignored.

@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 2.2.2.12) specifying the principal's options.

@MembershipWebId: The site identifier of the **site (2)** 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 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, and does not need to be added 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.
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 (2) has been exceeded and no additional users can be added to the site.

The **proc_SecAddUser** stored procedure MUST NOT return any result sets.

3.1.5.52 proc_SecAddUserToSiteGroup

The **proc_SecAddUserToSiteGroup** stored procedure is invoked to add a user to a **permission level** in the **site collection**. The user is added to the permission level and the root **site (2)** of the site collection, the user's **User Token** 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,
      @AppPrincipalId               int,
      @SiteAdmin                    bit,
      @BelongsToGroup               bit,
      @GroupOwnerId                 int,
      @CurrentUserIsOwner           bit,
      @RequestGuid                  uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The **site collection identifier** (section 2.2.1.1.9) for the site collection that contains the permission level specified by *@GroupId*.

@GroupId: A permission level identifier for the permission level that the user specified by *@UserIDToBeAdded* refers to. This value MUST refer to an existing permission level within the site collection specified by *@SiteId*.

@UserIDToBeAdded: A user identifier for the user being added to the permission level 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 permission level specified by *@GroupId*. This value MUST refer to an existing user Identifier in the site collection specified by *@SiteId*.

@AppPrincipalId: The **app principal** identifier of the **app** that is requesting the operation. If the operation was not requested by an app then it MUST be set to 0.

@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 permission level. If this parameter is set to 0, then the user specified by *@UserID* is not a member of the permission level.

@GroupOwnerId: The user identifier for the owner of the permission level specified by *@GroupId*. *@GroupOwnerId* MUST refer to an existing owner, which can be either a user or another permission level.

@CurrentUserIsOwner: If this parameter is set to 1, then the user specified by *@GroupOwnerId* is the owner of the permission level 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 permission level.
5	The user specified by *@UserID* does not have rights to modify the membership of the permission level specified by *@GroupID*.

The **proc_SecAddUserToSiteGroup** stored procedure MUST NOT return any result sets.

3.1.5.53 proc_SecAddWebMembership

The **proc_SecAddWebMembership** stored procedure is invoked to associate an existing user within a site collection to a **site (2)** within the same site collection.

```
PROCEDURE proc_SecAddWebMembership(
      @SiteId                       uniqueidentifier,
      @WebId                        uniqueidentifier,
      @UserId                       int,
      @RequestGuid                  uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The site collection identifier of the site collection containing the requested site (2) specified by *@WebId* and requested user identifier specified by *@UserId*.

@WebId: The site identifier of the site (2) 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 (2). 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 (2) 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.

3.1.5.54 proc_SecChangeToInheritedList

The **proc_SecChangeToInheritedList** stored procedure is invoked to change the scope of the specified list to the specified **site (2)** and remove any **role assignment**s 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(
      @SiteId                       uniqueidentifier,
      @WebId                        uniqueidentifier,
      @ListId                       uniqueidentifier,
      @RequestGuid                  uniqueidentifier = NULL OUTPUT
);
```

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

@WebId: The **site identifier** of the site (2) containing the list specified by *@ListId*.

@ListId: The **list identifier** of the list.

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

Return Values: This procedure returns an integer return code which MUST be 0, and MUST return one result set on successful completion. If this procedure encounters a failure, then no result set is returned.

3.1.5.54.1 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 (2)**. 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 the Common Result Sets **Site Audit Mask Result Set** (section 2.2.4.22).

3.1.5.55 proc_SecChangeToInheritedWeb

The **proc_SecChangeToInheritedWeb** stored procedure changes a **site (2)** from having its own unique permissions to instead use the permissions inherited from its nearest ancestor with unique permissions. When a site (2) 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 for the site collection containing the site (2) to be changed specified by *@WebId*.

@WebId: The site identifier for the site (2) 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 (2) 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.

3.1.5.55.1 Inherited Site Result Set

The **Inherited Site Result Set** returns the scope identifier and site identifier that the **site (2)** 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 (2) specified by *@WebId* now inherits from.

{ParentWebPermAncestor}: The site identifier for an ancestor site that the changed site (2) specified by *@WebId* now derives permissions from.

3.1.5.55.2 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 (2)** 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 the Common Result Sets **Site Audit Mask Result Set** (section 2.2.4.22).

3.1.5.56 proc_SecChangeToUniqueScope

The **proc_SecChangeToUniqueScope** **stored procedure** sets a **securable object** such as a **site (2)**, **list (1)**, 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** 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** of the site (2) that is or contains the securable object.

@OldScopeId: The **scope identifier** 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** 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 (2). If this parameter is set to one, the securable object is a site (2), and all **subsites** which inherit their security scope from the site (2) specified by *@SiteId* MUST have their inheritances changed to the new security scope. This parameter MUST be set to zero if *@Url* does not point to a site (2), and this parameter MUST be set to one if *@Url* points to a site (2).

@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 one, 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 one, the role assignments MUST be copied from the *@CopyFromScopeId* parameter into the new security scope. If

both *@bBreakBySiteOwner* and *@CopyRoleAssignments* are set to one, 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 (2), list (1) 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 one, every site (2), list (1) 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 (2) owner in the administrator role instead of the principal specified by *@UserId*. If this parameter is set to one, the site (2) owner MUST be added to the administrator role in the new security scope definition. If both *@bBreakBySiteOwner* and *@CopyRoleAssignments* are set, 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** (section 3.1.5.56.1). If this parameter is set to one, 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 (1). If a **list item's** security scope is changed and the list (1) including this list item already has *@MaxScopeInList* unique security scopes, the stored procedure MUST return an error using the integer return code 1340.

@NewScopeId: The **Scope Identifier** (section 2.2.1.1.8) 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 match the scope identifier of the securable object, or the securable object has unique permissions.
1340	The security scope of a list item is changed and the list (1) including this list item already has *@MaxScopeInList* unique security scopes.

If *@ReturnAuditMask* has a value of one, the **proc_SecChangeToUniqueScope** stored procedure MUST return a single result set on successful completion; otherwise, zero result sets MUST be 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 **site (2)**. 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 the Common Result Sets **Site Audit Mask Result Set** (section 2.2.4.22).

3.1.5.57 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 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.

3.1.5.57.1 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 2.2.6.10); otherwise zero rows are returned.

```
      tp_Login                      nvarchar(255);
```

tp_Login: The login name for the user.

3.1.5.58 proc_SecCloneRoleDefinitions

The **proc_SecCloneRoleDefinitions** stored procedure creates a copy of the current role definition for a **site (2)**. After successful execution, the site (2) will have its own copy of the role definition and unique role assignments, and the site (2) 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 of the site collection containing the site (2).

@WebId: The site identifier of the site (2) 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 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 for the security scope of the site (2). If the site (2) 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 (2) 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 (2) was not found. Either the site (2) specified by *@SiteId*/*@WebId* does not exist, or the site already has its own role definition.
80	The site (2) 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.

3.1.5.58.1 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 (2)**. 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 in Common Result Sets **Site Audit Mask Result Set** (section 2.2.4.22).

3.1.5.59 proc_SecCreateSiteGroup

The **proc_SecCreateSiteGroup** stored procedure is invoked to add a new **permission level** to a **site collection**.

```
PROCEDURE proc_SecCreateSiteGroup(
      @SiteId                       uniqueidentifier,
      @Title                        nvarchar(255),
      @Description                  nvarchar(512),
      @OwnerID                      int,
      @OwnerIsUser                  bit,
      @AppPrincipalId               int,
      @FirstMemberId                int,
      @UseExisting                  bit,
      @SelfOwner                    bit,
      @Flags                        int,
      @GroupID                      int                 OUTPUT,
      @RequestGuid                  uniqueidentifier = NULL     OUTPUT
);
```

@SiteId: The **site collection identifier** (section 2.2.1.1.9) of the site collection containing the new permission level to be created.

@Title: The title of the new permission level. This parameter MUST NOT be NULL.

@Description: A description of the new permission level.

@OwnerID: A user identifier or site group identifier 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.

@AppPrincipalId: The **app principal** identifier of the **app** that is requesting the operation. If the operation was not requested by an app then it MUST be set to 0.

@FirstMemberId: A user identifier for the first member of the new site group. This value MUST correspond to an existing user identifier 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.

@Flags: Contains a set of flags specifying properties about 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 e-mail 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.
0x00000010	The site group is hidden in the user interface.

@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 NOT return any result sets.

3.1.5.60 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 account 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 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.

3.1.5.61 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 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.

3.1.5.61.1 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** of the matching User.

3.1.5.62 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 for the site collection containing the scope.

@ScopeId: The Scope Identifier 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.

3.1.5.62.1 ACL and Permission Result Set

The **ACL and Permission Result Set** (section 2.2.4.1) 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 Common Result Sets **ACL and Permission Result Set** section.

3.1.5.63 proc_SecGetAllAclsForSite

The **proc_SecGetAllAclsForSite** stored procedure is invoked to list all Scope Identifiers 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 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.

3.1.5.63.1 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 of the scope.

Acl: The Access Control List (ACL) associated with the scope.

AnonymousPermMask: The **WSS Rights Mask** for anonymous users for this scope.

3.1.5.64 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 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.

3.1.5.64.1 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, 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** 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.

3.1.5.64.2 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**, 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 of the member who belongs to the site group specified by the **GroupId** in the first column.

tp_SystemID: The SystemID 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 2.2.2.12.

3.1.5.65 proc_SecGetApplicationPrincipalAndUserToken

The **proc_SecGetApplicationPrincipalAndUserToken** stored procedure is invoked to return information about a **principal (1)** based on the principal's login name and user identifier.

```
PROCEDURE proc SecGetApplicationPrincipalAndUserToken (
      @SiteId                       uniqueidentifier,
      @AppLogin                     nvarchar(255),
      @UserId                       int,
      @RequestGuid                  uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The Site Collection Identifier (section 2.2.1.1.9) 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.

3.1.5.65.1 Application Principal Result Set

The **Application Principal Result Set** returns information on a **principal (1)** 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 e-mail address.

tp_Token: A **WSS User Token** (section 2.2.3.10) value specifying the site group membership of the principal.

tp_Flags: Contains the user information flags value associated with this principal.

3.1.5.65.2 External Token Result Set

The **External Token Result Set** returns information on a **principal (1)** 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 2.2.3.2) value encoding information on external group membership derived from an external authentication role provider.

3.1.5.66 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: Site Collection Identifier of the site collection containing the Site whose permission information is requested.

@WebId: Site Identifier 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.

3.1.5.66.1 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 or user identifier of the **principal (1)**.

tp_SystemID: The **SystemId** of the principal

tp_DomainGroup: Set to 1 if the principal is a domain group; otherwise, 0.

3.1.5.67 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 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.

3.1.5.67.1 User Count Result Set

The **User Count Result Set** returns the number of Users registered with this site collection when configured in **Active Directory account creation mode**. When not in Active Directory account 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 account 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.

3.1.5.68 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.

@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.

3.1.5.68.1 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 the Common Result Sets **Domain Group Cache Versions Result Set** (section 2.2.4.4).

3.1.5.68.2 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 2.2.4.4). 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 the Common Result Sets **Domain Group Cache BEDS Update Result Set** (section 2.2.4.3).

3.1.5.68.3 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 2.2.4.4). 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 the Common Result Sets **Domain Group Cache WFE Update Result Set** (section 2.2.4.5).

3.1.5.69 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** of the site collection to search for the Group.

@GroupId: The identifier (a **Site Group Identifier** for site groups, or a **User Identifier** 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.

3.1.5.70 proc_SecGetGroupOwner

The **proc_SecGetGroupOwner** stored procedure is invoked to retrieve the User Identifier or Site Group Identifier 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: Site Collection Identifier for the site collection containing the site group.

@GroupId: The site group's 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.
1319	The site group does not exist.

The **proc_SecGetGroupOwner** stored procedure MUST NOT return a result set.

3.1.5.71 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 of the site collection containing the site group.

@PrincipalId: The site group identifier.

@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.

3.1.5.71.1 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}**.

3.1.5.72 proc_SecGetIndividualUrlSecurityCheckEventReceivers

The **proc_SecGetIndividualUrlSecurityCheckEventReceivers** stored procedure is invoked to request security information and event receivers information about a document at a specified location. 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,
      @Level                        tinyint                = NULL,
      @HasEventReceiver             bit                    OUTPUT,
      @MinLevel                     tinyint                OUTPUT,
      @RequestGuid                  uniqueidentifier       = NULL OUTPUT
);
```

@SiteId: The site collection identifier of the site collection containing the document location.

@WebId: The site identifier of the **site (2)** 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 of the current user, used to check for access privileges.

@AttachmentsFlag: An **Attachments Flag** (section 2.2.1.2.1) 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 3.1.5.72.1). 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.

@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.

3.1.5.72.1 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 3.1.5.72.2) 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 the Common **Result Sets Individual URL Security Result Set** (section 2.2.4.12).

3.1.5.72.2 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 the Common Result Sets **NULL Individual URL Security Result Set** (section 2.2.4.16).

3.1.5.72.3 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 the Common Result Sets **List Metadata Result Set** (section 2.2.4.14).

3.1.5.72.4 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 the Common Result Sets **Event Receivers Result Set** (section 2.2.4.11).

3.1.5.72.5 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 for the scope associated with the document location.

Acl: The ACL in **WSS ACL Format** (section 2.2.3.6) of the scope associated with the document location.

AnonymousPermMask: The **WSS Rights Mask** (section 2.2.2.15) that applies to an anonymous user, or a user with no assigned rights, in the scope associated with the document location.

3.1.5.73 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 of the site collection containing the list items.

@WebId: The Site Identifier of the **site (2)** containing the list items.

@ListId: The List Identifier 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 2.2.1.2.11) of issues list, or a **List Server Template** (section 2.2.1.2.12) 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.

3.1.5.73.1 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. 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.

3.1.5.74 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 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.

3.1.5.74.1 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 the Common Result Sets **Principal User Information Result Set** (section 2.2.4.19).

3.1.5.75 proc_SecGetPrincipalById

The **proc_SecGetPrincipalById** stored procedure is invoked to return information about a **principal (1)** or collection of principals based on a specified site group identifier or user identifier.

```
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 of the site collection containing the principal.

@PrincipalId: The site group identifier of the site group or user identifier 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.

3.1.5.75.1 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 of the principal.

tp_SystemID: The principal's **SystemId** (section 2.2.1.1.12).

tp_Title: The principal's display name.

tp_Login: The principal's login name.

tp_Email: The principal's e-mail 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 user information flags value describing this principal.

{tp_ExternalTokenLastUpdated}: A **timestamp** in UTC format specifying the time when the **External Group Token** (section 2.2.3.2) 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 2.2.3.10) value specifying the site group membership of the principal. This parameter MUST be NULL if *@GetSTSToken* is 0.

3.1.5.76 proc_SecGetPrincipalByIdEx

The **proc_SecGetPrincipalByIdEx** stored procedure is invoked to return information about a principal based on a specified user identifier or a collection of principals based on a specified site group identifier. 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 of the site collection containing the principal(s).

@PrincipalId: The site group identifier of the site group or user identifier 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.

3.1.5.76.1 Extended Principal User Information Result Set

The **Extended Principal User Information Result Set** returns information on a specified **principal (1)** 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 of the principal.

tp_SystemID: The principal's **SystemId** (section 2.2.1.1.12).

tp_Title: The principal's display name.

tp_Login: The principal's login name.

tp_Email: The principal's e-mail 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 **User Information Flags** value describing this principal.

{tp_ExternalTokenLastUpdated}: A **timestamp** in UTC format specifying the time when the **External Group Token** (section 2.2.3.2) 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 2.2.3.10) 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.

3.1.5.77 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 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 2.2.3.10) 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 2.2.3.2) 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.

3.1.5.77.1 User Information Result Set

The **User Information Result Set** returns security and attribute information about a specified **principal (1)**. 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 2.2.6.10); 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 for the specified principal.

tp_SystemID: The **SystemId** (section 2.2.1.1.12) 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 **User Information Flags** value describing this principal.

{tp_ExternalTokenLastUpdated}: A **datetime** in UTC format specifying the time when the external group token was last updated. This value MUST be NULL if the *@GetExternalToken* parameter is not set to 1.

{tp_ExternalToken}: An **External Group Token** (section 2.2.3.2) 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** value specifying the site group membership of the principal. This value MUST be NULL if the *@GetSTSToken* parameter is not set to 1.

3.1.5.78 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 for the site collection that contains the security principals (2).

@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##*:

3.1.5.78.1 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 (2) ID, otherwise it MUST contain no rows. The **User Information Result Set** is defined in the **proc_SecGetPrincipalByLogin User Information Result Set** (section 3.1.5.19.5).

3.1.5.79 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 for the site collection containing the security principals (2) and the Security Groups to be listed.

@WebId: A site identifier for a **site (2)**. 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_SecGetPrincipalDisplayInformation**20 stored procedure returns an integer return code which MUST be 0.

The **proc_SecGetPrincipalDisplayInformation**20 stored procedure MUST return one or two result sets for each non-NULL *@PrincipalId##*:

3.1.5.79.1 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 (1)**. 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. 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.

3.1.5.79.2 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 of the principal. MUST match *@PrincipalId##*.

UserSID: The **SystemID** 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 user information flags 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.

3.1.5.80 proc_SecGetRoleAssignments

The **proc_SecGetRoleAssignments** stored procedure is invoked to request the **WSS ACE** (section 2.2.3.5) 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 of the site collection containing the requested security scope.

@ScopeID: The scope identifier of the security scope containing the requested WSS ACEs.

@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.

3.1.5.80.1 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 2.2.3.5).

```
PrincipalId                    int,
PermMask                       bigint;
```

PrincipalId: A 4 byte signed integer specifying the user identifier of the principal for this **WSS ACE**.

PermMask: A **WSS Rights Mask** (section 2.2.2.15) that contains the list of rights that is granted to the principal.

3.1.5.81 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 of the site collection containing the security scope.

@ScopeId: The scope identifier 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.

3.1.5.81.1 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 of the principal.

3.1.5.82 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 for the desired site collection. This parameter MUST correspond to a valid site collection.

@ScopeId: The scope identifier 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.

3.1.5.82.1 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 2.2.1.2.16) of the role.

WebId: The Site Identifier for the **site (2)** 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 2.2.2.15) 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.

3.1.5.83 proc_SecGetSecurityInfo

The **proc_SecGetSecurityInfo** **stored procedure** retrieves security permissions information about a document. The document can be specified using its **document identifier** 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 2.2.1.1.9) of the site collection containing the document.

@WebId: The **Site Identifier** (section 2.2.1.1.11) of the **site (2)** containing the document. This parameter MUST be ignored.

@Url: The **store-relative form** URL for the document. The *@Url* parameter MUST be NULL to specify the document with the *@DocID* parameter.

@ThresholdCount: If *@Url* is a folder and the **list (1)** does not have more **list items** than *@ThresholdCount*, the **BigListFolder** field in the **Security Information Result Set** (section 3.1.5.83.1) MUST be zero.

@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 3.1.5.83.1) upon successful execution, or no result sets if the document is not found.

3.1.5.83.1 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;
        {ListId}                      uniqueidentifier
```

{ScopeId}: Contains the **scope identifier** 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 (ACL)** of the scope that contains the document.

AnonymousPermMask: The **WSS Rights Mask** (section 2.2.2.15) 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.

{ListId}: The **list identifier** of the list containing the document.

3.1.5.84 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 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

3.1.5.84.1 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 of the security principal (2) who is a site collection administrator.

tp_SystemID: The **SystemID** of the security principal (2).

tp_Title: The name of the security principal (2), used for display on the front-end Web server.

tp_Login: The login name of the security principal (2), for example, EXAMPLE\username.

tp_Email: The email address of the security principal (2), 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 (2) is a site collection administrator. This value is set to 0 if the security principal (2) is not a site collection administrator.

tp_DomainGroup: This value is set to 1 if the security principal (2) is a domain group. This value is set to 0 if the security principal (2) is not a domain group.

tp_Flags: A 4-byte integer bit mask determining the security principal's (2) options as specified in **UserInfo Flags** (section 2.2.2.12).

3.1.5.85 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 = NULL OUTPUT
);
```

@SiteId: The Site Collection Identifier 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.

3.1.5.85.1 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 the **Sec_SiteGroupsView** (section 2.2.6.5).

3.1.5.86 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 = NULL OUTPUT
);
```

@SiteId: The Site Collection Identifier 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.

3.1.5.86.1 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 the **Sec_SiteGroupsView** (section 2.2.6.5).

3.1.5.87 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 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##*.

3.1.5.87.1 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 by the **Sec_SiteGroupsView** (section 2.2.6.5).

3.1.5.88 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 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.

3.1.5.88.1 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.

3.1.5.89 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 of the site collection containing the site group specified by *@GroupId*.

@GroupId: The site group identifier of the site group whose permissions are to be returned.

@UserId: The user identifier 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.

3.1.5.90 proc_SecGetWebsAndListWithUniquePermissions

The **proc_SecGetWebsAndListsWithUniquePermissions** **stored procedure** 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** of the **site collection**.

@WebId: The **site identifier** of the site (2) to query.

@WebUrl: The **URL** 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.

Result Sets: **proc_SecGetWebsAndListsWithUniquePermissions** MUST return four **result sets**:

- **Site Information Result Set** (section 3.1.5.90.1)
- **Sub Site Information Result Set** (section 3.1.5.90.2)
- **Site List Information Result Set** (section 3.1.5.90.3)

- **Sub Site List Information Result Set** (section 3.1.5.90.4)

3.1.5.90.1 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 (2) has no unique permissions.

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

FullUrl: The **URL** of the site (2) specified by *@WebId*.

Id: The **site identifier** of the site (2) specified by *@WebId*.

Title: The title associated with the site (2) specified by *@WebId*.

3.1.5.90.2 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** of the subsite.

Title: The title associated with the subsite.

3.1.5.90.3 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 (2) specified by *@WebId*.

Id: The **site identifier** of the site (2) specified by *@WebId*.

Title: The title associated with the site (2) specified by *@WebId*.

{ListRootFolderUrl}: Root folder of the list which has unique permissions in the site (2) specified by *@WebId*.

tp_ID: The list identifier 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 (2). This parameter is set to 1 if the list permissions is not as the same as the site (2).

3.1.5.90.4 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 (2)** specified by the *@WebId*.

There MUST be a row for each list in the subsite of the site (2) 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 (2) specified by the *@WebId*.

Id: The **site identifier** of the site (2) specified by the *@WebId*.

Title: The title associated with the site (2).

{ListRootFolderUrl}: Root folder of the list which has unique permissions in the site (2).

tp_ID: The list identifier 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 (2). This parameter is set to 1 if the list permissions is not as the same as the site (2).

3.1.5.91 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 of the site collection containing the requested security principal (2) 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.

3.1.5.91.1 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 that uniquely identifies the **security principal** with the site collection.

tp_SystemID: The **SystemID** of the security principal.

tp_Title: The display name of the security principal (2).

tp_Login: The login name of the security principal (2).

tp_Email: The email address of the security principal (2), 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 (2) is a site collection administrator. This value is set to 0 if the security principal (2) is not a site collection administrator.

tp_DomainGroup: This value is set to 1 if the security principal (2) is a domain group. This value is set to 0 if the security principal (2) is not a domain group.

tp_Flags: A 4-byte integer bit mask specifying the security principal's (2) options as specified in **UserInfo Flags** (section 2.2.2.12).

3.1.5.92 proc_SecListAllWebMembers

The **proc_SecListAllWebMembers** stored procedure is invoked to list all non-deleted user and domain group accounts registered with a specified **site (2)**.

```
PROCEDURE proc_SecListAllWebMembers(
      @SiteId                   uniqueidentifier,
      @WebId                    uniqueidentifier,
      @RequestGuid              uniqueidentifier          OUTPUT
);
```

@SiteId: The site collection identifier of the site collection containing the site (2) specified by *@WebId*.

@WebId: The site identifier of the site (2) 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.

3.1.5.92.1 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 (2)**. 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 (2) 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 of the **security principal**.

tp_SystemID: The **SystemID** of the security principal.

tp_Title: The display name of the security principal (2).

tp_Login: The login name of the security principal (2).

tp_Email: The email address of the security principal.

tp_Notes: A descriptive text associated with the security principal (2).

tp_SiteAdmin: Contains a bit flag set to 1 if the security principal (2) 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 (2) is a domain group. This value is set to 0 if the security principal (2) is not for a domain group.

tp_Flags: A 4-byte integer bit mask determining the security principal's (2) options as specified in **UserInfo Flags** (section 2.2.2.12).

3.1.5.93 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 of the site collection that contains the site groups that have the role.

@ScopeId: The Scope Identifier 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.

3.1.5.93.1 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 the **Sec_SiteGroupsView** (section 2.2.6.5).

3.1.5.94 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 of the site collection which contains the scope whose site groups with role assignments are being listed.

@ScopeId: The Scope Identifier 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.

3.1.5.94.1 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 the **Sec_SiteGroupsView** (section 2.2.6.5).

3.1.5.95 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 of the site collection containing the scope.

@ScopeId: The scope identifier 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.

3.1.5.95.1 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 *@Site*Id or *@ScopeId* do not match existing site collection or scope identifiers, respectively.

The **User Information Result Set** is defined in the Common Result Sets **Principal User Information Result Set** (section 2.2.4.19).

3.1.5.96 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 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.

3.1.5.96.1 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 the Common Result Sets **Principal User Information Result Set** (section 2.2.4.19).

3.1.5.97 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 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.

3.1.5.97.1 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 the **Sec_SiteGroupsView** (section 2.2.6.5).

3.1.5.98 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 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 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.

3.1.5.98.1 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 the **Sec_SiteGroupsView** (section 2.2.6.5).

3.1.5.99 proc_SecListSiteGroupsWhichUserOwns

The **proc_SecListSiteGroupsWhichUserOwns** stored procedure returns site group information for the site groups owned by the specified **principal (1)**.

```
PROCEDURE proc_SecListSiteGroupsWhichUserOwns(
      @SiteId                       uniqueidentifier,
      @UserId                       int,
      @RequestGuid                  uniqueidentifier
);
```

@SiteId: The site collection identifier of the site collection containing the principal and the site groups to be listed.

@UserId: The user identifier 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.

3.1.5.99.1 Site Group Information Result Set

The **Site Group Information Result Set** returns site group information for all site groups which 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 the **Sec_SiteGroupsView** (section 2.2.6.5).

3.1.5.100 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 of the site collection containing the scope, role, and principals to be listed.

@ScopeId: The scope identifier 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.

3.1.5.100.1 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 (1)**.

```
        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 of the principal.

tp_SystemID: The **SystemID** 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 user information flags value of the principal.

3.1.5.101 proc_SecMigrateUser

The **proc_SecMigrateUser** stored procedure updates the SystemID and login name for a **principal (1)** in the **UserInfo** (section 2.2.6.10) and **AllUserData** (section 2.2.6.2) 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 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 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 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.

3.1.5.102 proc_SecReCalculateWebFGP

The **proc_SecReCalculateWebFGP** stored procedure is invoked to update the bit at 0x00000400 of the **Site Property Flags** (section 2.2.2.11) to indicate whether the **site (2)** has at least one uniquely secured object within it. This bit can be used by a search tool to determine whether it avoids indexing pages in this site (2) so as not to reveal secure information.

```
PROCEDURE proc_SecReCalculateWebFGP(
      @SiteId                       uniqueidentifier,
      @WebId                        uniqueidentifier,
      @RequestGuid                  uniqueidentifier
);
```

@SiteId: The site collection identifier of the site collection containing the site (2) to be updated. This parameter MUST correspond to a valid site collection, and MUST NOT be NULL.

@WebId: The site identifier of the site (2) 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.

3.1.5.103 proc_SecRefreshToken

The **proc_SecRefreshToken** **stored procedure** is invoked to update the **UserInfo Table** (section 2.2.6.10) 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** of the **site collection** containing the user to update in the **UserInfo Table** (section 2.2.6.10).

@UserId: The user identifier of the user.

@ExternalToken: An **External Group Token** (section 2.2.3.2) 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 if the table row value in **tp_ExternalTokenLastUpdated** is not greater than the value of *@ExternalTokenTime*.

@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 if the table row value in **tp_ExternalTokenLastUpdated** is not greater than the value of *@ExternalTokenTime*.

@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**.

3.1.5.104 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,
      @AppPrincipalId               int,
      @SiteAdmin                    bit,
      @GroupOwnerId                 int,
      @CurrentUserIsOwner           bit,
      @ReturnDLAlias                bit,
      @CheckIfInUse                 bit,
      @RequestGuid                  uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The site collection identifier of the site collection containing the site group to be removed.

@GroupId: The site group identifier for the site group to be removed.

@UserID: The user identifier 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.

@AppPrincipalId: The **app principal** identifier of the **app** that is requesting this operation. If the operation was not requested by an app then it MUST be set to 0.

@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.

Value	Description
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.

3.1.5.104.1 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 e-mail distribution address for the site group.

3.1.5.105 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,
      @AppPrincipalId               int,
      @SiteAdmin                    bit,
      @GroupOwnerId                 int,
      @CurrentUserIsOwner           bit,
      @DeletionWebId                uniqueidentifier,
      @RequestGuid                  uniqueidentifier
);
```

@SiteId: The **site collection identifier** (section 2.2.1.1.9) of the **site collection** containing the site group to be marked for removal.

@GroupId: The site group identifier for the site group to be marked for removal.

@UserID: The user identifier for the current user.

@AppPrincipalId: The **app principal** identifier of the **app** requesting this operation. If the request is not made by an app then it MUST be set to 0.

@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 of the **site (2)** 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.

3.1.5.105.1 DLAlias Result Set

If the site group existed and was marked for removal successfully, the **DLAlias Result Set** MUST return the e-mail distribution address of the site group in a single row. **The DLAlias Result Set** MUST be empty if the e-mail distribution address was blank.

```
DLAlias                 varchar(128);
```

DLAlias: The e-mail distribution address for the site group.

3.1.5.106 proc_SecRemovePrincipalFromScope

The **proc_SecRemovePrincipalFromScope** stored procedure removes a principal from a **security role** associated with a **site (2)** 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 of the site collection containing the principal and scope.

@WebId: The Site Identifier of the site (2) containing the principal and scope.

@WebScopeId: The Scope Identifier 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 = *@ScopeId*	Remove the principal from the specified role or roles within the Site and within all 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 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.

3.1.5.106.1 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 the Common Result Sets **Site Audit Mask Result Set** (section 2.2.4.22).

3.1.5.107 proc_SecRemoveRoleDef

The **proc_SecRemoveRoleDef** stored procedure removes a role definition, and any role assignments which use that role definition, from a specified **site (2)**.

```
PROCEDURE proc SecRemoveRoleDef(
      @SiteId                       uniqueidentifier,
      @WebId                        uniqueidentifier,
```

```
      @RoleId                       int,
      @RequestGuid                  uniqueidentifier = NULL OUTPUT
);
```

@SiteId: The site collection identifier of the site collection containing the role definition to be removed.

@WebId: The site identifier 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.

3.1.5.107.1 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 the Common Result Sets **Site Audit Mask Result Set** (section 2.2.4.22).

3.1.5.108 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 (2)**.

```
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 of the site collection containing the user and scope.

@WebId: The site identifier of the site (2) using the role.

@WebScopeId: The Scope Identifier of the scope which includes the site (2) 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 (2) specified by *@WebId*.
- If *@WebScopeId* is not equal to *@ScopeId*, remove the specified user from all scopes associated with the site (2) 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 is 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.

3.1.5.108.1 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 the Common Result Sets **Site Audit Mask Result Set** (section 2.2.4.22).

3.1.5.109 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 (2)** owner.

```
PROCEDURE proc_SecRemoveUserFromSite(
      @SiteId                       uniqueidentifier,
      @NewSiteGroupOwnerId          int,
      @UserId                       int,
      @RequestGuid                  uniqueidentifier   = NULL OUTPUT
);
```

@SiteId: The Site Collection Identifier 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 in the specified site collection, and *@NewSiteGroupOwnerId* MUST NOT have the same value as *@UserId*. If the value is -1, or system account (1073741823) or app account (1073741822) 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.
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.

3.1.5.110 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,
      @AppPrincipalId               int,
      @SiteAdmin                    bit,
      @BelongsToGroup               bit,
      @GroupOwnerId                 int,
      @CurrentUserIsOwner           bit,
      @RequestGuid                  uniqueidentifier   = NULL OUTPUT
);
```

@SiteId: The **site collection identifier** (section 2.2.1.1.9) of the site collection that will be queried for the requested site group.

@GroupId: The site group identifier for the site group that the user specified by *@UserIDToBeDeleted* is to be removed from.

@UserIDToBeDeleted: The user identifier 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 (2)**.

@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.

@AppPrincipalId: The **app principal** identifier of the **app** that is requesting this remove operation. If the request is not made by an app then it MUST be set to 0.

@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 specified user from the site group.

The **proc_SecRemoveUserFromSiteGroup** stored procedure MUST NOT return any result sets.

3.1.5.111 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 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 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 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.

3.1.5.112 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 2.2.1.1.9) of the **site collection** containing the list item.

@WebId: The **Site Identifier** (section 2.2.1.1.11) of the **site (2)** containing the item.

@OldScopeId: The **Scope Identifier** (section 2.2.1.1.8) of the **security scope** of the item.

@Url: The **store-relative form URL** of the item.

@DocId: The **document identifier** 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 the **Site Audit Mask Result Set** (section 3.1.5.112.1).

3.1.5.112.1 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 the Common Result Sets **Site Audit Mask Result Set** (section 2.2.4.22).

3.1.5.113 proc_SecResetWebToDefaultRoleDefinition

The **proc_SecResetWebToDefaultRoleDefinition** stored procedure is invoked to replace existing roles, role assignments, and Permissions for a specified **site (2)** and its **subsites** with a default set of role definitions, as specified by input parameters. If the specified site (2) 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 (2) does not exist or if the specified site (2) has a unique role definition.

```
PROCEDURE proc_SecResetWebToDefaultRoleDefinition(
      @SiteId                       uniqueidentifier,
      @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
);
```

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

@WebId: The **Site Identifier** (section 2.2.1.1.11) of the site (2).

@AuthorID: The user identifier of the **principal (1)** 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 2.2.2.15) 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** which indicates the rights granted to a User that is anonymous, or has no specific rights.

@ScopeId: The **scope identifier** 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** (section 3.1.5.113.1), repeated five times.

3.1.5.113.1 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 the Common Result Sets **Site Audit Mask Result Set** (section 2.2.4.22).

3.1.5.114 proc_SecResolvePrincipal

The **proc_SecResolvePrincipal** stored procedure retrieves information about a **principal (1)** or site group in the specified site collection, given the principal's login name, e-mail 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 of the site collection.

@Input: The principal's display name, e-mail 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 e-mail address. If the principal is a site group whose e-mail address is available, then this is used to construct the full e-mail address in the **{Email}** column in the **Principal Information Result Set** (section 3.1.5.114.1). 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

Value	Description
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.

3.1.5.114.1 Principal Information Result Set

The **Principal Information Result Set** MUST return a single row containing information about the **principal (1)** 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 e-mail 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 e-mail address, for example, someone@example.com, of the principal. This parameter is NULL if no match was found, if no e-mail 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.

3.1.5.115 proc_SecSetSiteGroupProperties

The **proc_SecSetSiteGroupProperties** **stored procedure** is invoked to update properties of a **site (2)** 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 2.2.1.1.9) of the **site collection** containing the site (2) group to update.

@GroupId: The identifier for the site (2) group to update.

@Title: The new user-friendly display name for the site (2) group. 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 five.

@Description: The new description for the site (2) group. This parameter can be NULL.

@OwnerID: The identifier of a user, domain group, or site (2) group to be set as the owner of the site (2) group. This parameter MUST contain the identifier of an existing user, domain group, or site (2) group.

@OwnerIsUser: A bit which specifies whether the value in *@OwnerId* is a user or domain group rather than a site (2) group. A value of one 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 (2) group identifier.

@UserID: The **User Identifier** (section 2.2.1.1.13) 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 one specifies that the current user has site collection administrator permission level. This parameter is used to check user's permission to change the site (2) group permissions and it can be NULL.

@GroupOwnerId: The **User Identifier** or **Site Group Identifier** (section 2.2.1.1.10) to compare with the existing owner of the site (2) group. Site (2) group management permissions MUST be granted if the value in *@GroupOwnerId* matches the identifier of the existing owner of the site (2) group and the value of *@CurrentUserIsOwner* is "1". This parameter is used to check user's permission to change the site (2) 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 (2) group. Site (2) group management permissions MUST be granted if the value in *@GroupOwnerId* matches the identifier of the existing owner of the site (2) 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 five.

@DLAlias: The new e-mail address for the **distribution list** associated with this site (2) 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 the **Sec_SiteGroupsView** (section 2.2.6.5). This parameter can be NULL.

@DLJobId: Contains the job identifier of a new currently pending asynchronous distribution **list (1)** operation. A value of zero or NULL specifies there is no pending operation. This parameter can be NULL.

@DLArchives: An array of bytes containing a new list (1) of pairs of **site identifiers** and **list identifiers** defining additional lists (1), which are the recipients of e-mail sent to the distribution list via the e-mail address in *@DLAlias*. Each pair MUST be stored as a string, with commas separating the list's (1) parent site (2) **document identifier** and the list identifier, and with semicolons following each pair. This parameter can be NULL (this will overwrite the current value).

@RequestEmail: The new e-mail address used to send a request to join or depart this site (2) group. This parameter can be NULL (this will overwrite the current value).

@Flags: Contains the new membership permissions bit flags for the site (2) group. The bit flags are defined in the description of the **Flags** column of the **Sec_SiteGroupsView** table.

@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 (2) 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.

3.1.5.116 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 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** 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.

3.1.5.117 proc_SecSetWebRequestAccess

The **proc_SecSetWebRequestAccess** stored procedure is invoked to set the request access email address for a specified **site (2)**. Site access requests to the particular site (2) are routed to the specified request access email address.

```
PROCEDURE proc_SecSetWebRequestAccess(
      @SiteId                       uniqueidentifier,
      @WebId                        uniqueidentifier,
      @RequestAccessEmail           nvarchar(255),
      @RequestGuid                  uniqueidentifier = NULL OUTPUT
);
```

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

@WebId: The **site identifier** of the site (2) 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: This procedure MUST return an integer return code of 0, and MUST NOT return a result set.

3.1.5.118 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 of the site collection containing the scope.

@WebId: The Site Identifier of the **site (2)** associated with the scope.

@ScopeId: The Scope Identifier of the scope containing the permission set.

@AnonymousPermMask: A WSS Rights Mask (section 2.2.2.15). 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.

3.1.5.119 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 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 the **WSS ACL** (section 2.2.3.6). This parameter MUST NOT be NULL.

@DomainGroupMapCache: The domain group map value is defined in the **WSS External Group Map Cache Format** (section 2.2.3.7).

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.

3.1.5.120 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 2.2.1.2.16), and **WSS Rights Mask** (section 2.2.2.15).

```
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 of the site collection containing the role definition to be updated.

@WebId: The site identifier of the **site (2)** 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 (2), 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** 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 (2) 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.

3.1.5.120.1 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 (2)**. 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 the Common Result Sets **Site Audit Mask Result Set** (section 2.2.4.22).

3.1.5.121 proc_SecUpdateUser

The **proc_SecUpdateUser** **stored procedure** is invoked to update the **display name**, **e-mail address**, notes, or administrative privileges listed in the user information associated with a **principal (1)**.

```
PROCEDURE proc SecUpdateUser(
      @SiteId                       uniqueidentifier,
      @CurrentUserId                int,
      @AppPrincipalId               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** of the **site collection** containing both the current user and the principal to be updated.

@CurrentUserId: The user identifier of the current user making the update request.

@AppPrincipalId: The **app principal** identifier of the **app** making the update request. If the request is not made by an app then this MUST be 0.

@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 e-mail 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 this parameter is set to "0", **proc_SecUpdateUser** MUST clear the site collection administrator flag in the user information for the principal.
- If this parameter is NULL, then **proc_SecUpdateUser** MUST make no changes to the site collection administrator flag.

@Flags: A flag specifying the user information flags (section 2.2.2.12) 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.
4335	Cannot successfully remove the site collection administrator privilege from the principal, as no other site collection administrator was found.

Result Sets: The **proc_SecUpdateUser** stored procedure MUST NOT return a **result set**.

3.1.5.122 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 of the site collection containing the user.

@UserId: The user identifier 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.

3.1.5.123 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 of the site collection containing the document.

@ListId: The List Identifier of the list containing the document.

@DirName: The directory name of the document.

@LeafName: The leaf name of the document.

@UserId: The User Identifier 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.
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.

3.1.5.124 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 the **proc_UncheckoutDocument** 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 of the site collection containing the document to have its short-term lock released or its check out reverted.

@WebId: The Site Identifier of the **site (2)** containing the document.

@DirName: The Document's directory name.

@LeafName: The Document's leaf name.

@UserId: The user identifier 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 checkout 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.

proc_UncheckoutDocument MUST return 2 or 3 result sets in the order specified in the following sections.

3.1.5.124.1 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 the Common Result Sets **Link Information Result Set** (section 2.2.4.13).

3.1.5.124.2 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 the Common Result Sets **Document Metadata Result Set** (section 2.2.4.8).

3.1.5.124.3 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.

3.1.5.124.4 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 2.2.1.2.5) 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 the Common Result Sets **Event Receivers Result Set** (section 2.2.4.11).

3.1.5.125 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 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** 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.

3.1.5.126 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),
        @VirusInfoEx                   varbinary(max),
        @LockTimeout                   int,
        @RefreshLock                   bit,
        @SharedLock                    bit,
        @FileFragmentPartitionToCheck  tinyint,
        @FileFragmentIdToCheck         bigint,
        @FileFragmentPartitionsToDelete nvarchar(max),
        @PartitionsToDelete            nvarchar(max),
        @DocContentVerBump             int,
        @NextBSNCheck                  bigint,
        @BSNBump                       bigint,
        @StreamSchema                  tinyint,
        @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** for the **site collection** containing the document to be updated.

@DocWebId: The **site identifier** for the **site (2)** 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**. This attribute MUST be one if the document stream of the document is intended to be stored in the back-end database server; otherwise, it MUST be zero.

@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 zero if *@SendingContent* is zero.

@DocMetaInfoSize: Size in bytes of the document's metadata. This MUST be zero 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, the entire document MUST NOT be updated. If the current internal version number value is NULL or *@PutFlags* contains 0x00800000 (from **Put Flags Type** (section 2.2.2.7)), 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, the entire document MUST NOT be updated. If the document was updated, 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, 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", the document is a thicket main file. If this parameter is set to "0", the document is not a thicket main file or a thicket supporting file. If this parameter is NULL, this document is a thicket supporting file.

@GetWebListForNormalization: If this parameter is set to "1", upon successful completion the procedure MUST return **list (1)** of sites (2) in a **Site List For Normalization Result Set** (section 3.1.5.126.1) whose immediate parent site is the site (2) containing the document to be updated.

@VersioningEnabled: A bit flag specifying whether historical versioning is enabled for the list (1) that contains this document. If this parameter is set to "0", a new historical version MUST NOT be created as part of this update. This parameter MUST be ignored for **uncustomized** 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", minor versions MUST be enabled; otherwise, minor versions MUST NOT be enabled. This parameter MUST be ignored for uncustomized items and documents that are not list items or in a document library.

@UserId: The **user identifier** of the current user that is making the request to the **front-end Web server**.

@AttachmentOp: An integer containing **Attachments Flags** (section 2.2.1.2.1) describing the document.

@PutFlags: A **Put Flags Type** value that specifies document update options.

@UpdateFlags: A bit field that tracks additional document change options. If this parameter is "1", 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, 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 editors for **HTML** or **XML** files). If this value IS NULL, 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", the **Site Audit Mask Result Set** (section 3.1.5.126.2) 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", 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, 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, 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, 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** value (section 2.2.1.2.17) 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.

@VirusInfoEx: A binary value that specifies extensible virus information for this document, which is opaque to the back-end server.

@LockTimeout: An integer value specifying the number of minutes a short-term lock is to be maintained for the document. A value of zero 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", 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", 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, this parameter MUST be NULL.

@SharedLock: Specifies whether the short-term lock to obtain, refresh, or release is of type shared or exclusive. If this parameter is "1", the lock type MUST be shared. Otherwise, the lock type MUST be exclusive. If *@LockTimeout* is NULL, this parameter MUST be ignored.

@FileFragmentPartitionToCheck: Specifies the implementation-specific file fragment partition identifier used with *@FileFragmentIdToCheck*.

@FileFragmentIdToCheck: Specifies the implementation-specific file fragment identifier to use for checking the implementation-specific file fragments, along with *@FileFragmentPartitionToCheck*. If *@FileFragmentPartitionToCheck* or *@FileFragmentIdToCheck* is NULL or the largest implementation-specific file fragment identifier in the partition specified by *@FileFragmentPartitionToCheck* is equal to @FileFragmentIdToCheck, the document MUST be updated and the current internal version number MUST be incremented. Otherwise it MUST NOT be incremented.

@FileFragmentPartitionsToDelete: A string of comma-separated integers that specifies which implementation-specific file fragment partition identifier to delete. If *@SystemUpdate* is one or *@SendingContent* is zero or *@VirusStatus* is not NULL and not three, this parameter MUST be ignored. Otherwise, the implementation-specific file fragment in those partitions MUST be deleted. 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
```

@PartitionsToDelete: A string of comma-separated integers that specifies which **stream partition** has all of its **stream binary pieces** deleted. If *@SystemUpdate* is one or *@SendingContent* is zero or *@VirusStatus* is not NULL and not three, this parameter MUST be ignored. Otherwise, the stream binary piece specified by the stream partitions MUST be deleted. The format MUST be the same as file-fragment-partition-list.

@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 one. If *@SendingContent* is set to zero, this parameter MUST be ignored.

@NextBSNCheck: Specifies the current **BLOB** sequence number to check for the document. If this parameter is NULL or the current BLOB sequence number of the document is equal to *@NextBSNCheck*, the document MUST be updated and the current **internal version number** MUST be incremented. Otherwise, the current internal version number MUST NOT be incremented.

@BSNBump: Specifies the amount to increase the current BLOB sequence number of the document. If *@SendingContent* is one, the current BLOB sequence number MUST be set to the current BLOB sequence number + @BSNBump.

@StreamSchema: Specifies the new **stream schema** of the document. If this parameter is not NULL, the current stream schema of the document MUST be set to *@StreamSchema*.

@DocId: Output parameter. The **document identifier** of the updated document.

@Level: Output parameter. A **Publishing Level Type** value (section 2.2.2.6) 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 (1).

@DocDTM: Output parameter. The datetime in UTC of the last modification time stamp of the updated document. If *@DocIncomingDTM* is NULL, this value MUST be set to the current UTC datetime according to the back-end database server; otherwise, the value MUST be set to *@DocIncomingDTM*.

@DocUIVersion: Output parameter. The **UI 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 **Doc Flags** value (section 2.2.2.3), set to the document flag 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 doesn't 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.
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.
6002	A short-term lock could not be obtained or modified because a shared short-term lock already exists. A shared short-term lock is used for read operations that do not change or update data, such as a SELECT statement.
6009	A short-term lock could not be obtained or modified because an exclusive short-term lock already exists. An exclusive short-term lock is used for data-modification operations, such as INSERT, UPDATE, or DELETE. Such a lock ensures that multiple updates cannot be made to the same resource at the same time.

The **proc_UpdateDocument** stored procedure can return zero to three result sets in the order listed in the following sections.

3.1.5.126.1 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 the Common Result Sets **URL Result Set** (section 2.2.4.27).

3.1.5.126.2 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 the Common Result Sets **Site Audit Mask Result Set** (section 2.2.4.22).

3.1.5.126.3 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.

3.1.5.127 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                   tinyint,
      @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,
        @tp_ColumnSet                   xml                 = NULL,
        @nvarchar1                      nvarchar(255)       = NULL,
        @MtgEventType                   int                 = NULL,
        @RecurrenceID                   datetime            = NULL,
        @IsDetached                     int                 = NULL,
        @eventData                      varbinary(max)      = NULL,
        @acl                            varbinary(max)      = NULL,
        @IsFirstRow                     bit                 = 1,
        @tp_AppAuthor                   int                 = NULL,
        @tp_AppEditor                   int                 = NULL,
        @RequestGuid                    uniqueidentifier    = NULL OUTPUT
);
```

@SiteId: The site collection identifier for the site collection containing the list item to be updated.

@WebId: The site identifier for the **site (2)** containing the list item.

@ListId: The list identifier 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 2.2.6.2). 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 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** 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** 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 (1)** 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 (1) because of 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 2.2.1.2.11) 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 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 is 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. For details on the **tContentTypeId** custom type, see section 2.2.1.1.1.

@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 2.2.4.14) 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 3.1.5.127.1).

@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 (2), then this specifies the value to set denoting the state of this list item within that workflow (2). If this list item is not part of a workflow (2), then this value MUST be NULL.

@tp_ColumnSet: The XML fragment that specifies the values of user-defined columns in the list. The name of each element specified in this parameter MUST be one of the columns defined in the **AllUserData table** (see section 2.2.6.2) with the SPARSE keyword.

@nvarchar1: User-defined column in the list containing a value of type **nvarchar**. If the column contains no data, then the value MUST be NULL.

@MtgEventType: If *@IsMeetingsList* is "1", this parameter specifies the type of calendar event.

@RecurrenceID: If *@IsMeetingsList* is "1", this parameter specifies a particular instance of a recurring meeting.

@IsDetached: If *@IsMeetingsList* is "1" and this parameter is "1", it specifies that this list item is a detached item from a meeting series.

@eventData: Contains implementation-specific event (1) 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 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.

@tp_AppAuthor: The **app principal** identifier of the app principal who created the list item.

@tp_AppEditor: The app principal identifier of the app principal who last edited the list item.

@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.

Value	Description
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.

3.1.5.127.1 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 2.2.6.2) 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 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 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.

3.1.5.128 proc_UpdateListSettings

The **proc_UpdateListSettings** **stored procedure** is invoked to update **list (1)** 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,
      @Followable                     bit,
      @ValidationFormula              nvarchar(1024),
      @ValidationMessage              nvarchar(1024),
      @RequestGuid                    uniqueidentifier   = NULL OUTPUT
);
```

@SiteId: The **site collection identifier** of the **site collection** containing the list (1) whose metadata is being updated.

@WebId: The **site identifier** of the **site (2)** containing the list (1).

@ListId: The **list identifier** of the list (1) to be updated.

@BaseType: A **List Base Type** value (section 2.2.1.2.11). If *@ConvertToGlobalMtgDataList* is equal to "1", 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 2.2.1.2.12) defining the base structure of this list (1). If *@ConvertToGlobalMtgDataList* is equal to "1", 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 (1). If this parameter is NULL, 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 (1). Valid read security values are listed in the following table.

Value	Description
1	Users with read permissions on this list (1) can read all list items.
2	Users with read permissions on this list (1) can only read their own list items.
NULL	There MUST be no changes to read security for the list (1).

@WriteSecurity: Specifies the security policy that MUST be set for write access on list items in the list (1). Valid write security values are listed in the following table.

Value	Description
1	Users with write permissions on this list (1) can create list items and have write access to all list items.
2	Users with write permissions on this list (1) 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 (1).

@NewDefaultView: A **View Identifier** (section 2.2.1.1.14) of a defined **list view** to be displayed by default when navigating to the list (1). If this parameter is NULL, the current value MUST NOT be changed.

@Description: A user-provided readable text description for the list (1). If this parameter is NULL, the current value MUST NOT be changed.

@TemplateDirName: Directory name for the **document template** associated with the list (1). 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 (1). If *@TemplateLeafName* is NULL, there MUST be no change to the current document template. If *@TemplateLeafName* contains an empty string, the current document template MUST be set to NULL. If *@TemplateLeafName* is not NULL and is not an empty string, the current document template MUST be set to the **Document Identifier** (section 2.2.1.1.2) of the document template specified by *@TemplateDirName* and *@TemplateLeafName*.

@Fields: Specifies the field definitions for the list (1), 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, 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 (1). If this parameter is NULL, 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 is left to right.
2	Text flow is right to left.

@EventSinkAssembly: A .NET assembly name for an **event sink** handler for the list (1). If this parameter is NULL, the current value MUST not be changed. If this parameter is an empty string, the current value MUST be set to NULL.

@EventSinkClass: A .NET assembly class identifier for an event sink handler for the list (1). If this parameter is NULL, 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 (1). If this parameter is NULL, the current value MUST not be changed. If this parameter is an empty string, the current value MUST be set to NULL.

@Flags: A **List Flags** value (section 2.2.2.5) for the list (1). If *@Fields* is not NULL, this value MUST be a bitwise OR with 0x800000 (LP_FIELDSCHEMAMODIFIED). If this parameter is NULL, the current value MUST NOT be changed.

@ImageUrl: A **site-relative URL** holding an image associated with the list (1). If this parameter is NULL, the current value MUST NOT be changed.

@ThumbnailSize: A value specifying the width in pixels used by lists (1) that have a **List Server Template** value (section 2.2.1.2.12) of 109 (Image Library Template) to determine the rendering size of an image thumbnail. If this parameter is NULL, the current value MUST NOT be changed.

@WebImageWidth: A value specifying the width in pixels used by lists (1) 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, the current value MUST NOT be changed.

@WebImageHeight: A value specifying the height in pixels used by lists (1) 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, the current value MUST NOT be changed.

@Version: An integer used to match on the internal version counter of the list (1) 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 (1) will be incremented by one. If this parameter does not match the current version, the list (1) MUST not be updated.

@ConvertToGlobalMtgDataList: Used to convert the data of the list (1) to use global meeting data. If this parameter is set to one, the 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
202	Meeting User
212	Homepage

If set to any other value or NULL, the meeting data for this list (1) MUST NOT be changed.

@EmailAlias: The email alias of the list (1). This alias is used to allow documents and list items to be sent directly to this list (1) through an implementation-specific email-handling feature. If this parameter is NULL, the current value MUST NOT be changed.

@SendToLocation: An implementation-specific Send-To string holding a **URL** used for the list (1). If this parameter is NULL, 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 (1). If *@AddOrRemoveFromNavBar* is NULL or zero, this parameter MUST be ignored. Otherwise, it MUST contain a valid navigation node element identifier.

@AddOrRemoveFromNavBar: Specifies whether to add or remove this list (1) from the site's top link bar. If this parameter is set to "0" and the list (1) was updated, the list (1) MUST be removed from the site's top link bar. If this parameter is NULL, there MUST be no changes to the site's top link bar. If this parameter is set to "1" and the list (1) was updated and the list (1) is not in the site's top link bar, the list (1) 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 root folder of the list (1) is used for the list (1) entry in the site's top link bar, instead of the default view page. If this parameter is set to "1" and the list (1) was updated, the root folder of the list (1) MUST be set as the top link bar target. If this parameter is set to "0" and the list (1) was updated, the default view page for the list (1) MUST be set as the top link bar target. If the *@AddOrRemoveFromNavBar* parameter is set to NULL, 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 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 (1).

@MajorMinorVersionCount: Specifies the maximum number of **minor versions** of list items to be retained by the list (1).

@WorkFlowId: Specifies the **Workflow Identifier** (section 2.2.1.1.16) of the default **workflow** of the list (1).

@RemoveSystemAlerts: A bit flag specifying whether to remove system **alerts** for this list (1) from the email subscriptions for the system administration user. If this parameter is set to "1", system alerts on the list (1) will be removed.

@Followable: If this list (1) is followable, this value MUST be one. Otherwise, it MUST be zero.

@ValidationFormula: The implementation-specific validation formula to be used for items in this list (1). 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 (1) validation formula fails. If this parameter is NULL, the current value MUST NOT be changed.

@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.

Value	Description
3	The site (2) does not exist or list (1) does not exist.
13	*@ConvertToGlobalMtgDataList* is set to one and *@ServerTemplate* is 200, 202, or 212 so list (1) cannot use global meeting data.
15	The document template URL (combination of *@TemplateDirName* and *@TemplateLeafName*) is not valid.
52	Another list (1) with *@Title* already exists.
1150	*@Version* did not match the current version of the list (1).
5069	*@ReadSecurity* is 0x00000002 (LSReadOnlyMyItems) and *@WriteSecurity* is not NULL and the list (1) contains a **field** with an implementation-specific list (1) validation constraint.

The **proc_updateListSettings** stored procedure MUST NOT return any result sets.

3.1.5.129 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,
      @MUILanguages                 varchar(64)        = NULL,
      @ContentLanguages             varchar(64)        = NULL,
      @RequestGuid                  uniqueidentifier   = NULL OUTPUT
);
```

@SiteId: The **Site Collection Identifier** (section 2.2.1.1.9) of the **site collection** containing the user whose information is to be updated.

@UserId: The User Identifier (section 2.2.1.1.13) 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 2.2.1.2.3) 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 for the time zone to be used when displaying time values for this User.

@Collation: The **Collation Order** (section 2.2.1.2.4) 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 2.2.3.1) which specifies the calendar display options setting for this User.

@WorkDays: A set of **Workdays Flags** (section 2.2.2.14) 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.

@MUILanguages: A string that contains the distinct culture(s) for the user's preferred display language(s), separated by semicolon. This string can be NULL.

@ContentLanguages: A string that contains the distinct culture(s) for the user's preferred content language(s), separated by semicolon. This string can be NULL.

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

Return Values: This procedure returns an integer return code which MUST be 0.

This procedure MUST return no result sets.

3.1.5.130 proc_UrlToWebUrl

The **proc_UrlToWebUrl** stored procedure returns the store-relative form URL of the **site (2)** 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 2.2.1.1.9) 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 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.

3.1.5.130.1 Web URL Result Set

The **Web URL Result Set** returns the store-relative form URL of the **site (2)** 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 (2) 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*.

3.1.5.131 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** of the **site collection** containing the document.

@DocId: The **document identifier** of the document.

@Offset: This parameter MUST be zero, if no document content corresponds to *@DocId*; otherwise, it MUST be the length, in bytes, of the document content which corresponded to *@DocId* before the stored procedure was invoked.

@00 -- @FF: 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, all subsequent parameters MUST be NULL.

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

Return Values: The **proc_WriteChunkToAllDocStreams** stored procedure returns 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.

3.1.5.132 proc_WriteStreamToRBS

The **proc_WriteStreamToRBS** **stored procedure** establishes the remote **BLOB** storage identifier representing a new **stream binary piece** for a document. For additional information regarding remote BLOB storage, see [MS-WSSO] section 2.1.2.3.8.

```
PROCEDURE dbo.proc_WriteStreamToRBS (
    @SiteId       uniqueidentifier,
    @DocId        uniqueidentifier,
    @Partition    tinyint,
    @BSN          bigint,
    @Size         int,
    @Type         tinyint,
    @Expiration   datetime,
    @RbsId        varbinary(64)
)
```

@SiteId: The **site collection identifier** of the **site collection** containing the document.

@DocId: The **document identifier** of the document.

@Partition: The **stream partition** that the new stream binary piece belongs to.

@BSN: The BLOB sequence number of the new stream binary piece.

@Size: The size, in bytes, of the new stream binary piece.

@Type: The **stream** type of the new stream binary piece.

@Expiration: The timestamp in **UTC** format of the stream binary piece. If the stream binary piece does not have an expiration, this parameter MUST be NULL.

@RbsId: The remote BLOB storage identifier for the new stream binary piece. For additional information regarding remote BLOB storage, see [MS-WSSO] section 2.1.2.3.8.

Return Values: The **proc_WriteStreamToRBS** stored procedure returns an integer return code which MUST be listed in the following table.

Value	Description
0	Successful execution.
29	Unexpected error.
80	A stream binary piece with the same BLOB sequence number or **stream identifier** already exists.

The **proc_WriteStreamToRBS** stored procedure MUST NOT return a result set.

3.1.5.133 proc_WriteStreams

The **proc_WriteStreams** **stored procedure** creates a new **stream binary piece** for a document.

```
PROCEDURE dbo.proc_WriteStreams (
    @SiteId       uniqueidentifier,
    @DocId        uniqueidentifier,
    @Partition    tinyint,
    @StreamData   tvpStreamData READONLY,
    @BSNData      tvpBSNMetadata2 READONLY,
    @AddStream    bit,
    @OverwriteStream bit
```

```
)
```

@SiteId: The **site collection identifier** of the **site collection** containing the document.

@DocId: The **document identifier** of the document.

@Partition: The **stream partition** that the new stream binary piece belongs to.

@StreamData: Contains subsets of data for the stream binary piece, corresponding to the Data, Offset, and Length columns.

@BSNData: Contains the metadata for the subsets of data for the stream binary piece, corresponding to *@StreamData*. The subset data in a position of *@StreamData* MUST correspond to the **BLOB** sequence number and **StreamId** specified in the same position in *@BSNData*.

@AddStream: If this parameter is one and a stream binary piece already exists with the same BLOB sequence number or **stream identifier** specified by *@BSNData***,** BLOB sequence number and **StreamId**, this stored procedure MUST return 80.

@OverwriteStream: If this parameter is one, *@AddStream* MUST be ignored.

Return Values: The **proc_WriteStreams** stored procedure returns an integer return code which MUST be listed in the following table.

Value	Description
0	Successful execution.
2	The document associated with the stream binary pieces to add does not exist.
29	Unexpected error.
80	A stream binary piece with the BLOB sequence number or stream identifier specified from *@BSNData* already exists.

The **proc_WriteStreams** stored procedure MUST NOT return a result set.

3.1.5.134 proc_WriteStreamToSQL

The **proc_WriteStreamToSQL** **stored procedure** creates a new **stream binary piece** for a document.

```
PROCEDURE dbo.proc_WriteStreamToSQL (
    @SiteId       uniqueidentifier,
    @DocId        uniqueidentifier,
    @Partition    tinyint,
    @BSN          bigint,
    @Size         int,
    @Type         tinyint,
    @Expiration   datetime,
    @Data0 varbinary(max), @Length0 bigint, @Offset0 bigint,
    @Data1 varbinary(max), @Length1 bigint, @Offset1 bigint,
    @Data2 varbinary(max), @Length2 bigint, @Offset2 bigint,
    @Data3 varbinary(max), @Length3 bigint, @Offset3 bigint,
    @Data4 varbinary(max), @Length4 bigint, @Offset4 bigint,
    @Data5 varbinary(max), @Length5 bigint, @Offset5 bigint,
    @Data6 varbinary(max), @Length6 bigint, @Offset6 bigint,
    @Data7 varbinary(max), @Length7 bigint, @Offset7 bigint,
    @Data8 varbinary(max), @Length8 bigint, @Offset8 bigint,
    @Data9 varbinary(max), @Length9 bigint, @Offset9 bigint,
    @Data10 varbinary(max), @Length10 bigint, @Offset10 bigint,
```

```
    @Data11 varbinary(max), @Length11 bigint, @Offset11 bigint,
    @Data12 varbinary(max), @Length12 bigint, @Offset12 bigint,
    @Data13 varbinary(max), @Length13 bigint, @Offset13 bigint,
    @Data14 varbinary(max), @Length14 bigint, @Offset14 bigint,
    @Data15 varbinary(max), @Length15 bigint, @Offset15 bigint,
    @Data16 varbinary(max), @Length16 bigint, @Offset16 bigint,
    @Data17 varbinary(max), @Length17 bigint, @Offset17 bigint,
    @Data18 varbinary(max), @Length18 bigint, @Offset18 bigint,
    @Data19 varbinary(max), @Length19 bigint, @Offset19 bigint,
    @Data20 varbinary(max), @Length20 bigint, @Offset20 bigint,
    @Data21 varbinary(max), @Length21 bigint, @Offset21 bigint,
    @Data22 varbinary(max), @Length22 bigint, @Offset22 bigint,
    @Data23 varbinary(max), @Length23 bigint, @Offset23 bigint,
    @Data24 varbinary(max), @Length24 bigint, @Offset24 bigint,
    @Data25 varbinary(max), @Length25 bigint, @Offset25 bigint,
    @Data26 varbinary(max), @Length26 bigint, @Offset26 bigint,
    @Data27 varbinary(max), @Length27 bigint, @Offset27 bigint,
    @Data28 varbinary(max), @Length28 bigint, @Offset28 bigint,
    @Data29 varbinary(max), @Length29 bigint, @Offset29 bigint,
    @Data30 varbinary(max), @Length30 bigint, @Offset30 bigint,
    @Data31 varbinary(max), @Length31 bigint, @Offset31 bigint,
    @Data32 varbinary(max), @Length32 bigint, @Offset32 bigint,
    @Data33 varbinary(max), @Length33 bigint, @Offset33 bigint,
    @Data34 varbinary(max), @Length34 bigint, @Offset34 bigint,
    @Data35 varbinary(max), @Length35 bigint, @Offset35 bigint,
    @Data36 varbinary(max), @Length36 bigint, @Offset36 bigint,
    @Data37 varbinary(max), @Length37 bigint, @Offset37 bigint,
    @Data38 varbinary(max), @Length38 bigint, @Offset38 bigint,
    @Data39 varbinary(max), @Length39 bigint, @Offset39 bigint,
    @Data40 varbinary(max), @Length40 bigint, @Offset40 bigint,
    @Data41 varbinary(max), @Length41 bigint, @Offset41 bigint,
    @Data42 varbinary(max), @Length42 bigint, @Offset42 bigint,
    @Data43 varbinary(max), @Length43 bigint, @Offset43 bigint,
    @Data44 varbinary(max), @Length44 bigint, @Offset44 bigint,
    @Data45 varbinary(max), @Length45 bigint, @Offset45 bigint,
    @Data46 varbinary(max), @Length46 bigint, @Offset46 bigint,
    @Data47 varbinary(max), @Length47 bigint, @Offset47 bigint,
    @Data48 varbinary(max), @Length48 bigint, @Offset48 bigint,
    @Data49 varbinary(max), @Length49 bigint, @Offset49 bigint,
    @Data50 varbinary(max), @Length50 bigint, @Offset50 bigint,
    @Data51 varbinary(max), @Length51 bigint, @Offset51 bigint,
    @Data52 varbinary(max), @Length52 bigint, @Offset52 bigint,
    @Data53 varbinary(max), @Length53 bigint, @Offset53 bigint,
    @Data54 varbinary(max), @Length54 bigint, @Offset54 bigint,
    @Data55 varbinary(max), @Length55 bigint, @Offset55 bigint,
    @Data56 varbinary(max), @Length56 bigint, @Offset56 bigint,
    @Data57 varbinary(max), @Length57 bigint, @Offset57 bigint,
    @Data58 varbinary(max), @Length58 bigint, @Offset58 bigint,
    @Data59 varbinary(max), @Length59 bigint, @Offset59 bigint,
    @Data60 varbinary(max), @Length60 bigint, @Offset60 bigint,
    @Data61 varbinary(max), @Length61 bigint, @Offset61 bigint,
    @Data62 varbinary(max), @Length62 bigint, @Offset62 bigint,
    @Data63 varbinary(max), @Length63 bigint, @Offset63 bigint
)
```

@SiteId: The **site collection identifier** of the **site collection** containing the document.

@DocId: The **document identifier** of the document.

@Partition: The **stream partition** that the new stream binary piece belongs to.

@BSN: The **BLOB** sequence number of the new stream binary piece.

@Size: The size, in bytes, of the new stream binary piece.

@Type: The **stream** type of the new stream binary piece.

@Expiration: The timestamp in **UTC** format of the stream binary piece. If the stream binary piece does not have an expiration, this parameter MUST be NULL.

@Data[0-63]: A piece of the binary data for the stream binary piece. See definition for details of format.

@Length[0-63]: This value MUST be the length of the corresponding *@Data* parameter, in bytes.

@Offset[0-63]: The offset in the stream binary piece where the corresponding *@Data* belongs. This value MUST be greater than or equal to zero, and this value plus the corresponding *@Length* parameter MUST be less than or equal to the total size of the new stream binary piece.

Return Values: The **proc_WriteStreamToSQL** stored procedure returns an integer return code which MUST be listed in the following table.

Value	Description
0	Successful execution.
29	Unexpected error.
80	A stream binary piece with the same BLOB sequence number or **stream identifier** already exists.

The **proc_WriteStreamToSQL** stored procedure MUST NOT return a result set.

3.1.5.135 proc_SetStreamsToDoc

The **proc_SetStreamsToDoc** **stored procedure** associates **stream binary pieces** with a document.

```
PROCEDURE dbo.proc_SetStreamsToDoc (
    @SiteId uniqueidentifier,
    @DocId uniqueidentifier,
    @HistVersion int,
    @Level tinyint,
    @Partition tinyint,
    @StreamsToAdd tvpBSNMetadata READONLY,
    @StreamsToDel tvpBSNMetadata READONLY,
    @DeleteAllStreams bit,
    @QuotaChange bigint = NULL OUTPUT
)
```

@SiteId: The **site collection identifier** of the **site collection** containing the document.

@DocId: The **document identifier** of the document.

@HistVersion: If the document is a historical version, this parameter MUST be the **UI version** number. Otherwise, this parameter MUST be zero.

@Level: The publishing level of the document.

@Partition: The **stream partition** to which the stream binary pieces belong.

@StreamsToAdd: Specifies the **BLOB** sequence number and **stream identifier** for the stream binary pieces to associate with the document.

@StreamsToDel: Specifies the BLOB sequence number for the stream binary pieces to unassociate with the document.

@DeleteAllStreams: If this parameter is one, all existing stream binary pieces MUST remove their association with the document.

@QuotaChange: An optional output parameter that specifies the quota change (negative value or zero), in bytes, of all the stream binary pieces that were unassociated from the document based on the *@StreamsToDel* and *@DeleteAllStreams* parameters.

Return Values: The **proc_SetStreamsToDoc** stored procedure returns an integer return code which MUST be listed in the following table.

Value	Description
0	Successful execution.
29	Unexpected error.
80	A stream binary piece with the same BLOB sequence number or stream identifier is already associated with the document.

The **proc_SetStreamsToDoc** stored procedure MUST NOT return a result set.

3.1.5.136 proc_SetStreamsToDocNoTVP

The **proc_SetStreamsToDocNoTVP** **stored procedure** associates **stream binary pieces** with a document.

```
PROCEDURE dbo.proc_SetStreamsToDocNoTVP (
    @SiteId uniqueidentifier,
    @DocId uniqueidentifier,
    @HistVersion int,
    @Level tinyint,
    @Partition tinyint,
    @BSNToAdd0 bigint, @StreamIdToAdd0 bigint,
    @BSNToAdd1 bigint, @StreamIdToAdd1 bigint,
    @BSNToAdd2 bigint, @StreamIdToAdd2 bigint,
    @BSNToAdd3 bigint, @StreamIdToAdd3 bigint,
    @BSNToAdd4 bigint, @StreamIdToAdd4 bigint,
    @BSNToAdd5 bigint, @StreamIdToAdd5 bigint,
    @BSNToAdd6 bigint, @StreamIdToAdd6 bigint,
    @BSNToAdd7 bigint, @StreamIdToAdd7 bigint,
    @BSNToAdd8 bigint, @StreamIdToAdd8 bigint,
    @BSNToAdd9 bigint, @StreamIdToAdd9 bigint,
    @BSNToAdd10 bigint, @StreamIdToAdd10 bigint,
    @BSNToAdd11 bigint, @StreamIdToAdd11 bigint,
    @BSNToAdd12 bigint, @StreamIdToAdd12 bigint,
    @BSNToAdd13 bigint, @StreamIdToAdd13 bigint,
    @BSNToAdd14 bigint, @StreamIdToAdd14 bigint,
    @BSNToAdd15 bigint, @StreamIdToAdd15 bigint,
    @BSNToAdd16 bigint, @StreamIdToAdd16 bigint,
    @BSNToAdd17 bigint, @StreamIdToAdd17 bigint,
    @BSNToAdd18 bigint, @StreamIdToAdd18 bigint,
    @BSNToAdd19 bigint, @StreamIdToAdd19 bigint,
    @BSNToAdd20 bigint, @StreamIdToAdd20 bigint,
    @BSNToAdd21 bigint, @StreamIdToAdd21 bigint,
    @BSNToAdd22 bigint, @StreamIdToAdd22 bigint,
    @BSNToAdd23 bigint, @StreamIdToAdd23 bigint,
    @BSNToAdd24 bigint, @StreamIdToAdd24 bigint,
    @BSNToAdd25 bigint, @StreamIdToAdd25 bigint,
    @BSNToAdd26 bigint, @StreamIdToAdd26 bigint,
    @BSNToAdd27 bigint, @StreamIdToAdd27 bigint,
    @BSNToAdd28 bigint, @StreamIdToAdd28 bigint,
    @BSNToAdd29 bigint, @StreamIdToAdd29 bigint,
    @BSNToAdd30 bigint, @StreamIdToAdd30 bigint,
    @BSNToAdd31 bigint, @StreamIdToAdd31 bigint,
```

```
    @BSNToDelete0 bigint,
    @BSNToDelete1 bigint,
    @BSNToDelete2 bigint,
    @BSNToDelete3 bigint,
    @BSNToDelete4 bigint,
    @BSNToDelete5 bigint,
    @BSNToDelete6 bigint,
    @BSNToDelete7 bigint,
    @BSNToDelete8 bigint,
    @BSNToDelete9 bigint,
    @BSNToDelete10 bigint,
    @BSNToDelete11 bigint,
    @BSNToDelete12 bigint,
    @BSNToDelete13 bigint,
    @BSNToDelete14 bigint,
    @BSNToDelete15 bigint,
    @BSNToDelete16 bigint,
    @BSNToDelete17 bigint,
    @BSNToDelete18 bigint,
    @BSNToDelete19 bigint,
    @BSNToDelete20 bigint,
    @BSNToDelete21 bigint,
    @BSNToDelete22 bigint,
    @BSNToDelete23 bigint,
    @BSNToDelete24 bigint,
    @BSNToDelete25 bigint,
    @BSNToDelete26 bigint,
    @BSNToDelete27 bigint,
    @BSNToDelete28 bigint,
    @BSNToDelete29 bigint,
    @BSNToDelete30 bigint,
    @BSNToDelete31 bigint,
    @DeleteAllStreams bit,
    @QuotaChange bigint = NULL OUTPUT
)
```

@SiteId: The **site collection identifier** of the **site collection** containing the document.

@DocId: The **document identifier** of the document.

@HistVersion: If the document is a historical version, this parameter MUST be the **UI version** number. Otherwise, this parameter MUST be zero.

@Level: The publishing level of the document.

@Partition: The **stream partition** to which the stream binary pieces belong.

@BSNToAdd[0-31]: The **BLOB** sequence number of the stream binary pieces to associate with the document.

@StreamIdToAdd[0-31]: The **stream identifiers** of the stream binary pieces to associate with the document. The position of the stream identifier in *@StreamIdToAdd* MUST match the position of the BLOB sequence number in *@BSNToAdd*.

@BSNToDelete[0-31]: The BLOB sequence number of the stream binary pieces to unassociate with the document.

@DeleteAllStreams: If this parameter is one, all existing stream binary pieces MUST remove their association with the document.

@QuotaChange: An optional output parameter that specifies the sum, in bytes, of all the stream binary pieces that were unassociated from the document based on the *@BSNToDelete[0-31]* and *@DeleteAllStreams* parameters.

Return Values: The **proc_SetStreamsToDocNoTVP** stored procedure returns an integer return code which MUST be listed in the following table.

Value	Description
0	Successful execution.
29	Unexpected error.
80	A stream binary piece with the same BLOB sequence number or stream identifier is already associated with the document.

The **proc_SetStreamsToDocNoTVP** stored procedure MUST NOT return a result set.

3.1.5.137 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 2.2.6.3).

```
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 DeleteTransactionId = 0x AND DirName = @DirName AND LeafName =
@LeafName AND Level = @Level.)
```

3.1.5.138 TVF_UserData_ListItemLevelRow

The **TVF_UserData_ListItemLevelRow** table value function is invoked to return a subset of rows from the **UserData View** (section 2.2.6.8).

```
FUNCTION [dbo].[TVF_UserData_ListItemLevelRow]
            (
                @tp_SiteId uniqueidentifier,
                @tp_ListId uniqueidentifier,
                @tp_Id int,
                @tp_Level tinyint,
                @tp_RowOrdinal tinyint
            )
```

@tp_SiteId: All the rows returned from the **UserData View** (section 2.2.6.8) MUST have **tp_SiteId** = *@tp_SiteId*.

@tp_ListId: All the rows returned from the **UserData View** (section 2.2.6.8) MUST have **tp_ListId** = *@tp_ListId*.

@tp_Id: All the rows return from the **UserData View** (section 2.2.6.8) MUST have **tp_Id** = *@tp_Id*.

@tp_Level: All the rows return from the **UserData View** (section 2.2.6.8) MUST have **tp_Level** = *@tp_Level*.

@tp_RowOrdinal: All the rows returned from the **UserData View** (section 2.2.6.8) MUST have **tp_RowOrdinal** = *@tp_RowOrdinal*.

Return Values: **TVF_UserData_ListItemLevelRow** (section 3.1.5.137) returns all rows from the **UserData View** (section 2.2.6.8) that are specified as follows.

```
(tp_SiteId = @tp_SiteId AND tp_ListId = @tp_ListId AND tp_DeleteTransactionId = 0x AND
tp_IsCurrentVersion = CONVERT(bit, 1) AND tp_Id = @tp_Id AND tp_CaculatedVersion = 0 AND
tp_Level = @tp_Level AND tp_RowOrdinal = @tp_RowOrdinal)
```

3.1.5.139 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 2.2.6.8).

```
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_DeleteTransactionId = 0x AND tp_IsCurrentVersion =
CONVERT(bit, 1) AND tp_ParentId = @tp_ParentId AND tp_DocId = @tp_DocId AND
tp_CalculatedVersion = 0 AND tp_Level = @tp_Level AND tp_RowOrdinal = @tp_RowOrdinal)
```

3.1.5.140 proc_HasCurrentPublishVersion

The **proc_HasCurrentPublishVersion** stored procedure is invoked to test whether the specified list item has a current published version.

```
PROCEDURE proc HasCurrentPublishVersion (
    @SiteId         uniqueidentifier,
    @ListId         uniqueidentifier,
    @ItemId         int,
    @RequestGuid    uniqueidentifier = null OUTPUT
);
```

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

@ListId: The **list identifier** of the list that contains the specified list item.

@ItemId: The **list item identifier** 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.

This stored procedure MUST NOT return a result set.

3.1.5.141 proc_GetSiteDenyPermMask

The **proc_GetSiteDenyPermMask** stored procedure is invoked to return a **WSS Rights Mask** (section 2.2.2.15) specifying the rights to deny to the current user in the specified site collection.

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

@SiteId: The Site Collection Identifier (section 2.2.1.1.9) of the site collection for which to retrieve the deny rights mask.

@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.

3.1.5.141.1 Deny Permissions Mask Result Set

The **Deny Permissions Mask Result Set** returns a **WSS Rights Mask** (section 2.2.2.15) specifying the rights to deny to the current user in the specified site collection.

```
      {DenyPermMask}                tPermMask;
```

{DenyPermMask}: A **WSS Rights Mask** (section 2.2.2.15) specifying the rights to deny to the current user.

3.1.5.142 proc_GetNewListItemId

The **proc_GetNewListItemId** stored procedure is called to get the next available row identifier of a **list (1)** and the number of rows that will be copied or moved when copying or moving a URL to that list (1). The T-SQL syntax for the stored procedure is as follows:

```
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 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 (1) where the URL specified by *@FullUrl* is being copied or moved to. The return value MUST be obtained by calling **proc_GenerateNextId** when the URL specified by *@FullUrl* is a file or folder; otherwise is undefined.

@MaxNewRows: This is an output parameter whose value indicates the total number of items as specified by *@FullUrl* that will be copied or moved. 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 is undefined.

Return values: An integer that MUST be zero.

3.1.5.143 proc_UpdateDiskUsed

The **proc_UpdateDiskUsed** stored procedure is called to update the disk-used size, in Bytes, of a site collection.

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

```
PROCEDURE proc_UpdateDiskUsed (
    @SiteId                     uniqueidentifier,
    @bUpdateTimeStampForce      bit = 0
);
```

@SiteId: The site collection identifier of the **site collection** whose disk-used size is to update.

@bUpdateTimeStampForce: A bit flag specifying whether to update the time stamp of **LastContentChange** of the site collection. If the value is 1, **LastContentChange** of the site collection MUST be updated with the current time. If the value is 0, **LastContentChange** MUST be unmodified.

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

Result Sets: MUST NOT return any result sets.

3.1.6 Timer Events

If the execution timeout **event (2)** 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 and the User Identifier 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 (2)** 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 **front-end Web server** opens an existing file stored as a document in a document library on the BEDS with a SharePoint Object Model call.

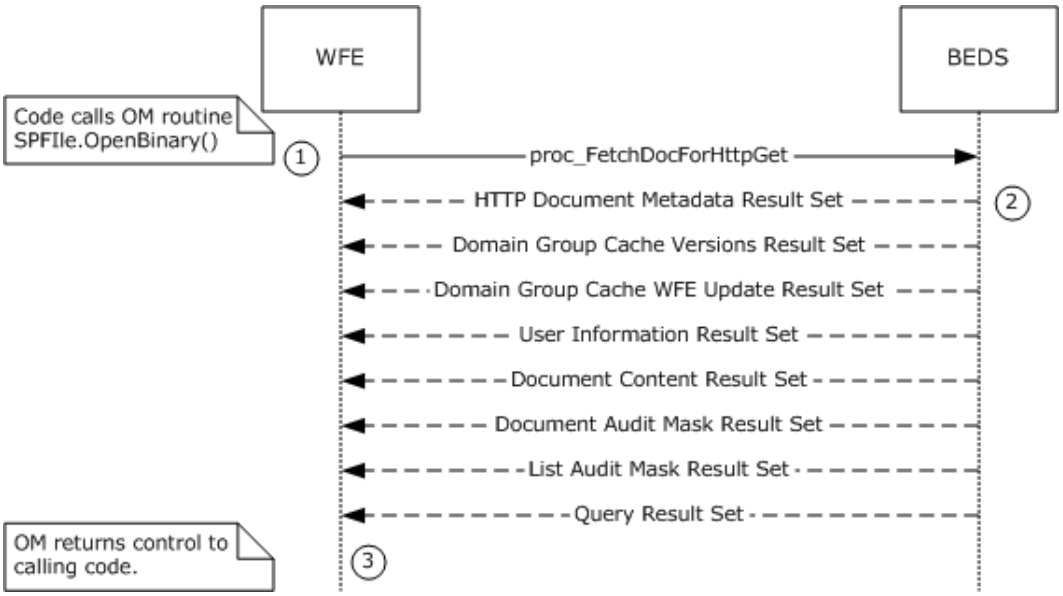


Figure 2: File: Open File OM

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 2.2.4.4). 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 2.2.4.5). 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 3.1.5.19.9). 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 2.2.4.22). Under our assumptions, auditing is not enabled on the site collection, so the **SiteGlobalAuditMask** column is NULL.
- **List Audit Mask Result Set** (section 3.1.5.19.11), 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 (2)**.
- 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 which 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.

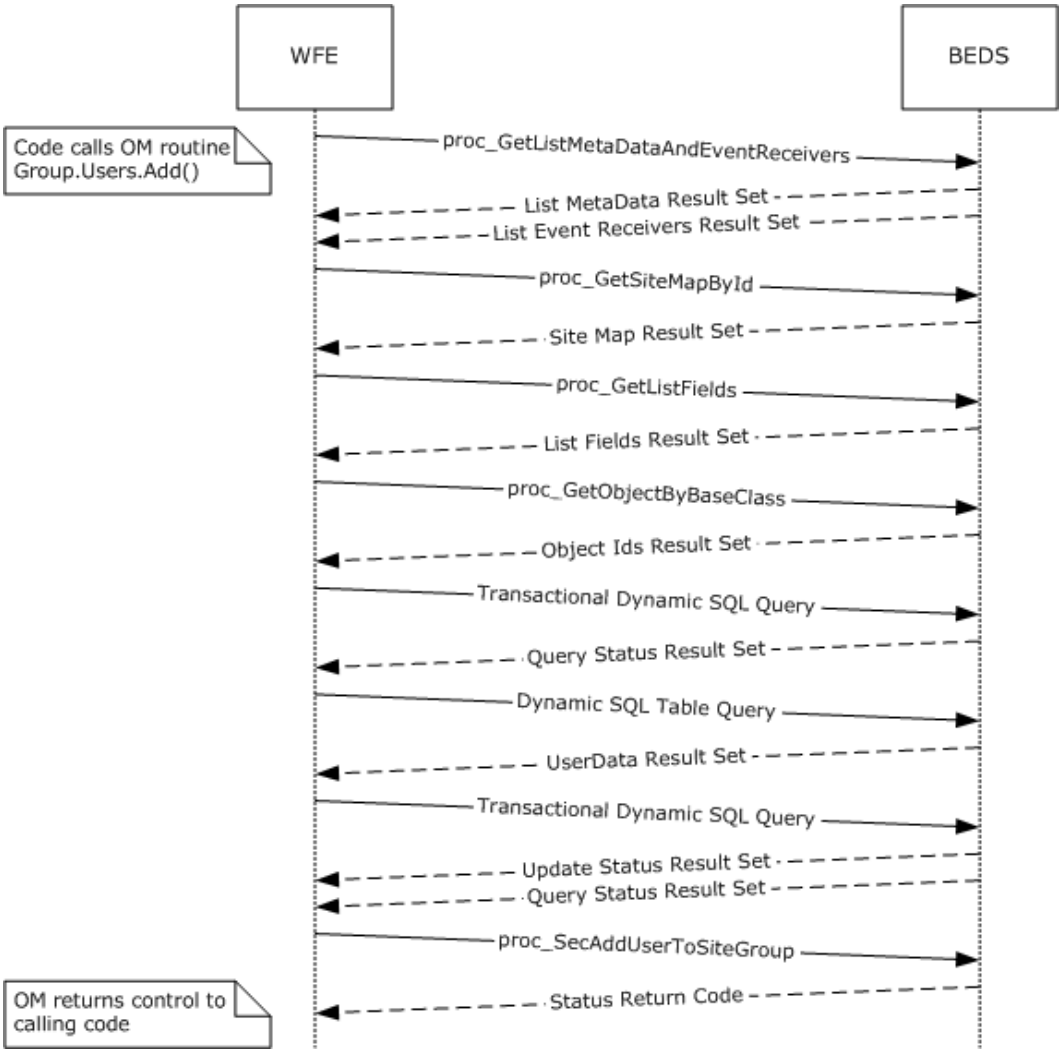


Figure 3: Group Add User To Site Group OM

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 (2)** registered in a site

collection. It does this by calling the stored procedure **proc_GetListMetaDataAndEventReceivers** (section 3.1.5.33) using TDS.

2. The BEDS returns two result sets, which include the **List Metadata** (section 2.2.4.14) and **Event Receivers** (section 2.2.4.11) for the specified user information list.

3. The WFE determines the site map for the site collection by calling the configuration database stored procedure **proc_getSiteMapById** (section 3.1.5.39) using TDS.

4. The BEDS returns a single **Site Map By Id Result Set** (section 3.1.5.39.1), 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** (section 3.1.5.32) stored procedure using TDS.

6. The BEDS returns a single **Fields Information Result Set** (section 3.1.5.32.1), 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 stored procedure **proc_getObjectsByBaseClass** (section 3.1.5.35) using TDS.

8. The BEDS returns a single **Object ID Result Set** (section 2.2.4.18), 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:

   - The query begins a new SQL transaction.
   - The query attempts to add the user to the site collection using the stored procedure **proc_SecAddUser** (section 3.1.5.51).
   - The query checks if the user exists in the site collection's user information list.
   - 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).
   - 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 2.2.6.8), 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:

    - The query begins a new SQL transaction.
    - The query checks if the user exists in the user information list for the site collection.

- 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**.
- 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 3.1.5.127).
- 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 3.1.5.52) 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.

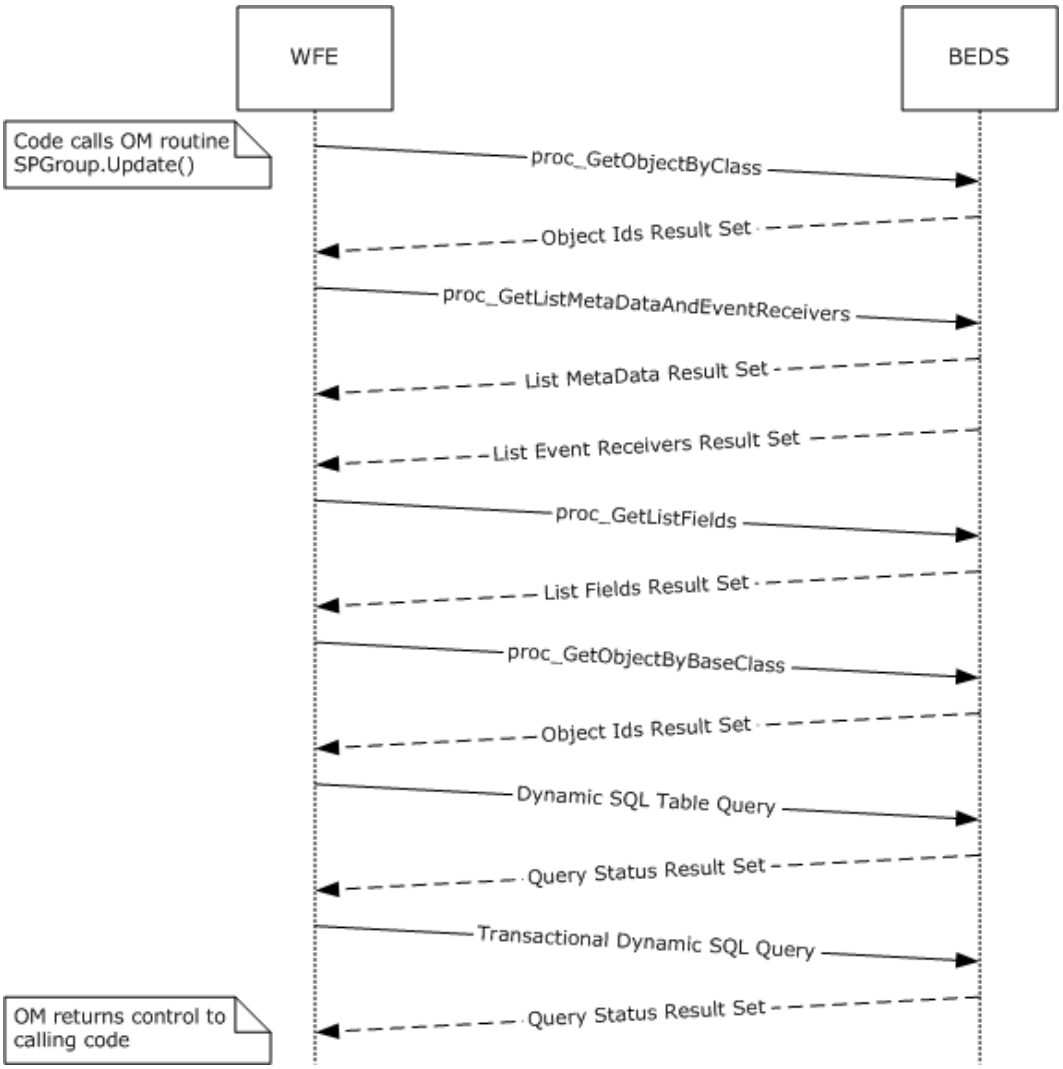


Figure 4: Group Update Site Group Properties OM

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 (2)**.

The following actions happen:

1. The WFE determines if the site (2) has the directory management service enabled with a call to the Configuration Database stored procedure **proc_GetObjectsByClass** (section 3.1.5.36) using TDS.
2. The BEDS returns a single **Object ID Result Set** (section 2.2.4.18) 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 (2)** registered in a site collection. It does this by calling the stored procedure **proc_GetListMetaDataAndEventReceivers** using TDS.

4. The BEDS returns two result sets, which include the **List Metadata** (section 2.2.4.14) and **Event Receivers** (section 2.2.4.11) 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 using TDS.

6. The BEDS returns a single **Field Information Result Set**, 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 stored procedure **proc_getObjectsByBaseClass** 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:

    - The query begins a new SQL transaction.
    - The query attempts to load the site group information using the stored procedure **proc_SecSetSiteGroupProperties** (section 3.1.5.115).
    - If the site group is not in the list of site **members**, the query invokes **proc_AddListItem** with the updated information.
    - 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 3.1.5.127).
    - 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.

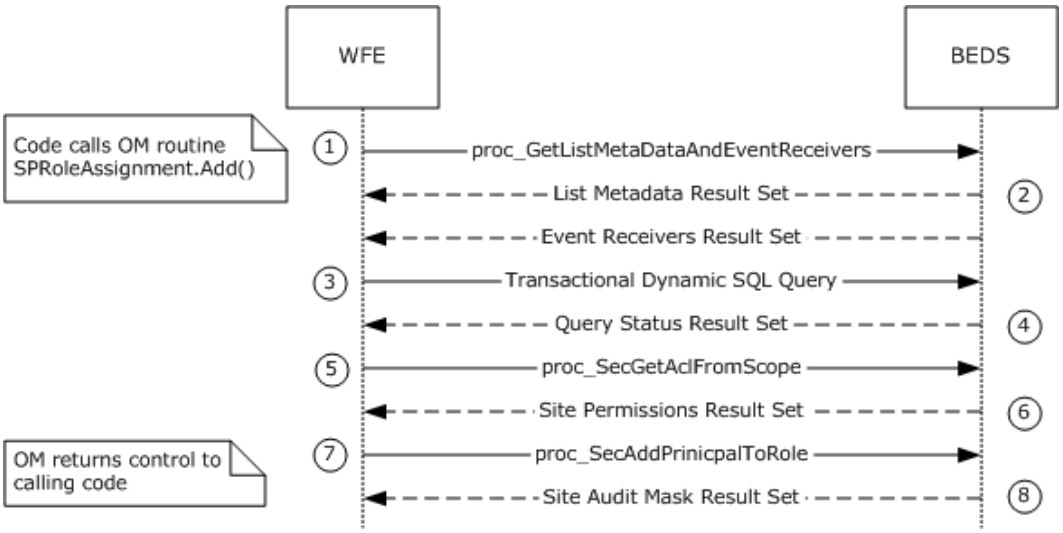


Figure 5: Add User to Document Library via Object Model

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, 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 3.1.5.33) using TDS.

2. The BEDS returns two result sets. The **List Metadata Result Set** returns a single row of data with the metadata for the list. The second result set is the **Event Receivers Result Set** (section 2.2.4.11). 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:

    - The query begins a new SQL transaction.
    - The query attempts to add the user to the site collection's user information list using the stored procedure **proc_SecAddUser** (section 3.1.5.51).
    - The query checks if the user exists in the site collection's user information list.
    - 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**.
    - 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 3.1.5.62) using TDS.

6. The BEDS returns an ACL and Permissions Result Set (section 2.2.4.1), 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 3.1.5.49) using TDS.

8. The BEDS returns a **Site Audit Mask Result Set** (section 2.2.4.22), 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 (2)**, rather than inheriting them from a parent.

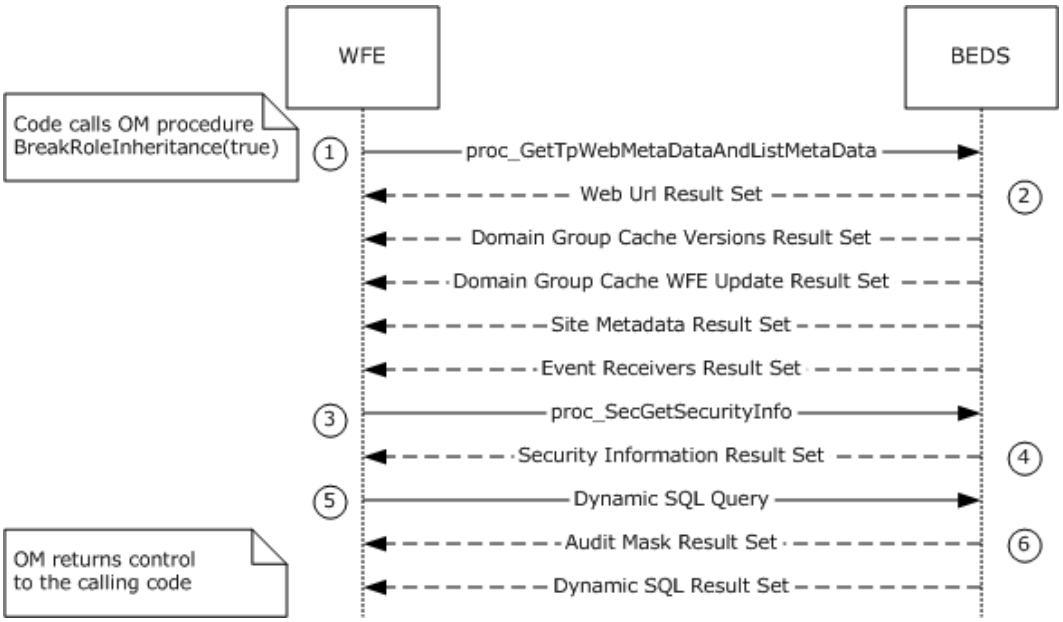


Figure 6: Break Web Inheritance OM

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 (2).

The following actions happen:

1. The WFE first retrieves metadata for the requested child site (2). It does this by calling the **proc_GetTpWebMetaDataAndListMetaData** (section 3.1.5.40) stored procedure using TDS.

2. The BEDS returns the following five result sets:

    - **Web URL Result Set** (section 3.1.5.40.1). This contains the store-relative form URL of the root of the requested child site (2).

- **Domain Group Cache Versions Result Set** (section 2.2.4.4). This contains information about the version numbers associated with the External Group Map Cache for the requested site (2).
- **Domain Group Cache WFE Update Result Set** (section 2.2.4.5). This returns the binary data needed to refresh the external group Map Cache.
- **Site Metadata Result Set** (section 2.2.4.24). This contains metadata for the requested site (2).
- **Event Receivers Result Set** (section 2.2.4.11). This contains information about the event receivers defined for the requested site (2).

3. The WFE then retrieves security permissions information about the requested site (2). It does this by calling the **proc_SecGetSecurityInfo** (section 3.1.5.83) stored procedure using TDS.
4. The BEDS returns the **Security Information Result Set** (section 3.1.5.83.1), which consists of information about security permissions about the requested site (2).
5. The WFE then builds a Dynamic SQL Query to convert the requested site (2) to use unique permissions (as opposed to inheriting those permission from the Parent site). It does this by calling the **proc_SecChangeToUniqueScope** (section 3.1.5.56) stored procedure using TDS.
6. The BEDS returns the following two result sets:
   - **SiteAudit Mask Result Set** (section 2.2.4.22), containing information about the Audit Flags (section 2.2.2.1) set for the requested child site (2).
   - A Dynamic SQL Result Set, containing the Scope Identifier (section 2.2.1.1.8) of the new scope generated for the child site (2).

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 using lower-level protocols is assumed.

4.6.1 Retrieving the Farm Id

The example begins by calling **proc_GetObjectsByClass** (section 3.1.5.36) with the **Farm Class ID**.

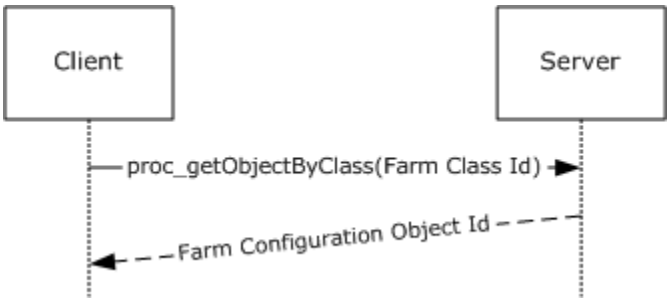


Figure 7: Retrieving the Farm ID

This call returns a result set including the **Configuration Object ID** 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 2.2.5.1) 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**.

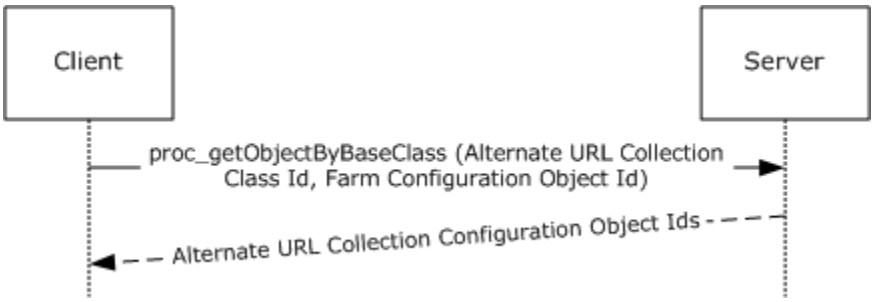


Figure 8: Web Service Lookup Operations

Because the implementation defined the returned Farm Configuration Object to be the Parent of all Alternate URL Collections, this call returns the **Configuration Object Ids** of all Alternate URL Collections stored in the Configuration Database.

4.6.3 Retrieving the Alternate URL Collections

The retrieved **Alternate URL Collection Configuration Object Ids** are then passed to **proc_getObject** to retrieve the full contents of the Alternate URL Collection Configuration Objects.

Figure 9: 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 (for example, "http://example.com:80"). If a match is found the configuration object 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**.

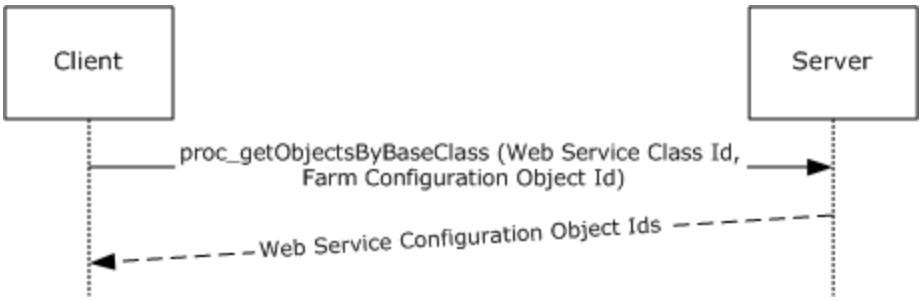


Figure 10: Web Service Lookup Operations

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.

4.6.6 Retrieving the Web Application Ids

In this step, the specified Web application **Class Id** is passed to **proc_getObjectsByBaseClass** with each of the retrieved **Web Service Configuration Object Ids**. This returns the Configuration Object Ids of all **Web applications** in the Configuration Database.

Figure 11: 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** to retrieve the full contents of the Web Application Configuration Objects.

Figure 12: 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 2.2.5.1.7.1) 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 contains 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 3.1.5.38), along with the **Request Web Application ID**.

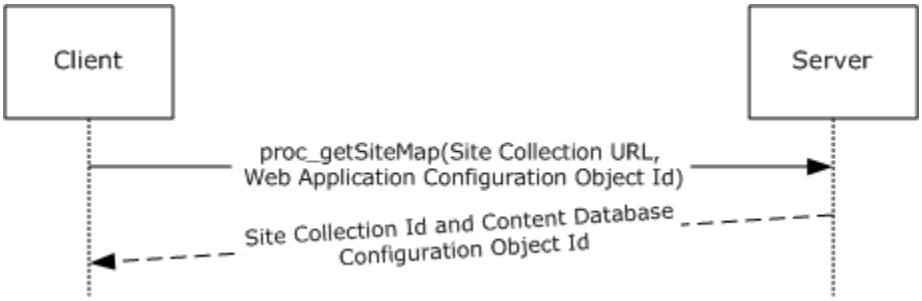


Figure 13: Site Collection Id Lookup

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, 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**, which returns the content database configuration object (section 2.2.5.1).

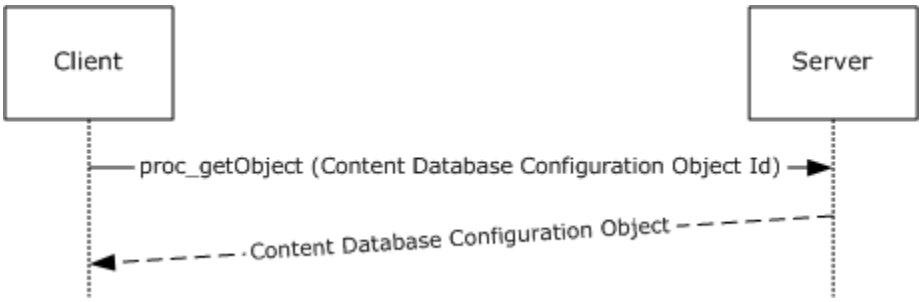


Figure 14: Name

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.

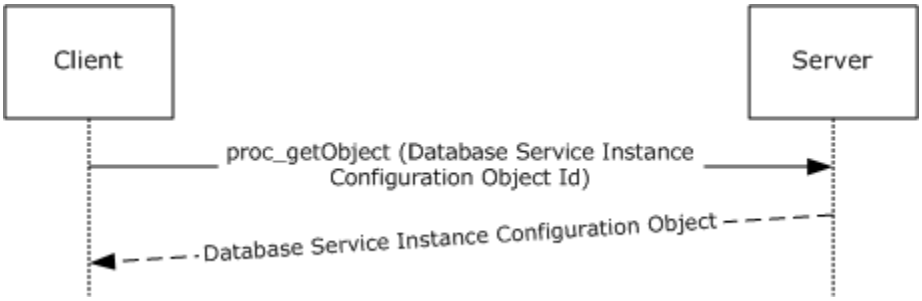


Figure 15: Database Server Instance

Server Address

The **ParentId** of the Database Service Instance Configuration Object is then passed to the one final call to **proc_getObject**, which a server configuration object. The name field of the server configuration object is used as the Server Address component of the connection string.

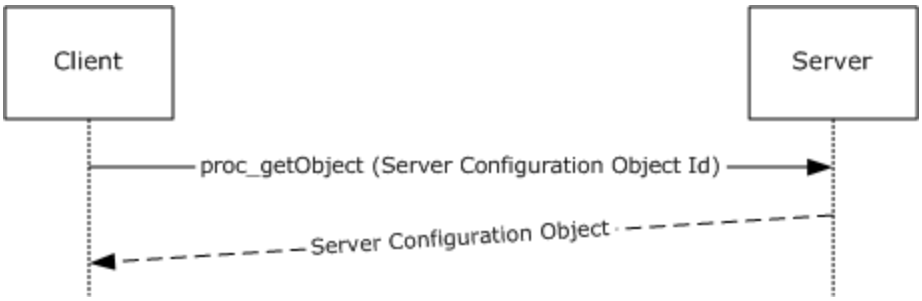


Figure 16: Server Address

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.

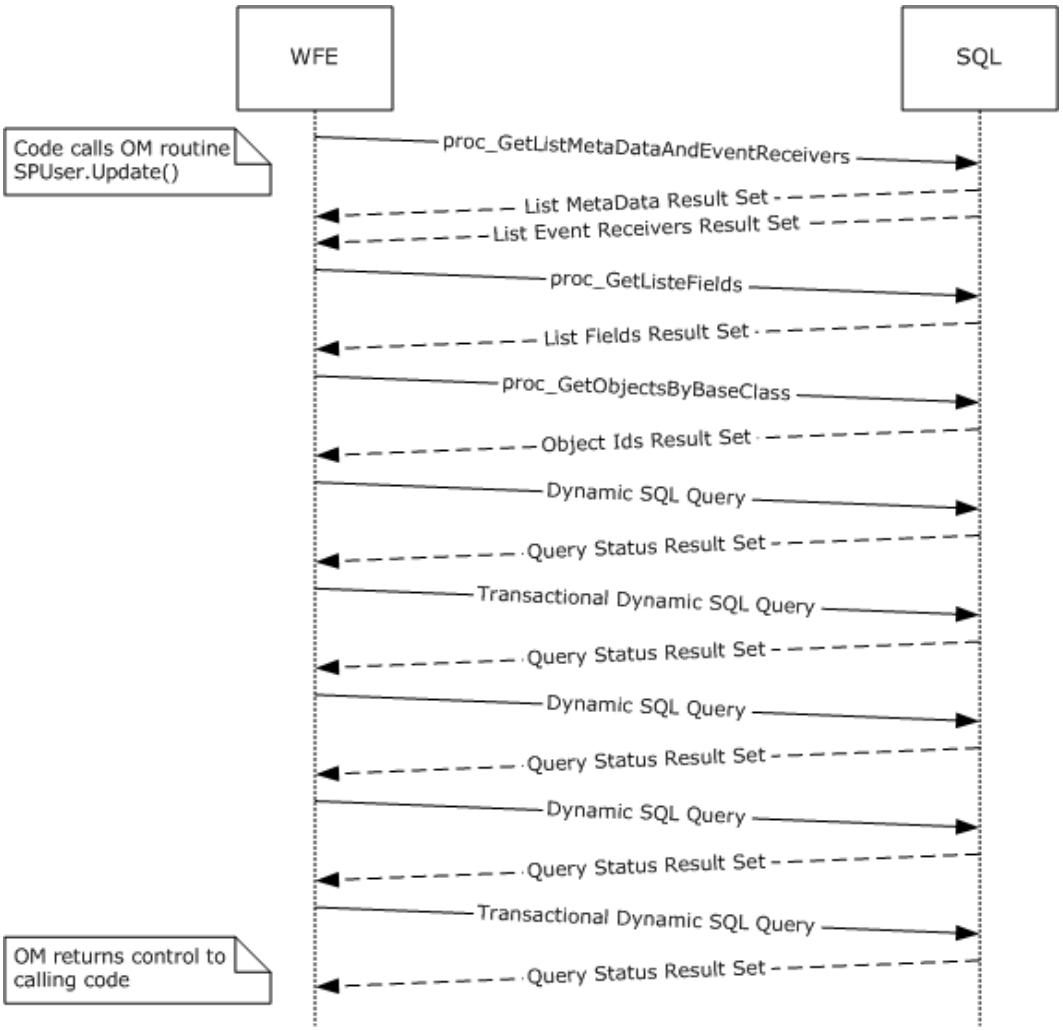


Figure 17: User Update User Properties OM

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 (2)**.

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 (2)** registered in a site collection. It does this by calling the stored procedure **proc_GetListMetaDataAndEventReceivers** (section 3.1.5.33) using TDS.
2. The BEDS returns two result sets, which include the list metadata (section 2.2.4.14) and event receivers 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** using TDS.

4. The BEDS returns a single fields information result set, 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 stored procedure **proc_getObjectsByBaseClass** 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 (2). This query is sent to the SQL server using TDS. On the SQL server the following actions occur:

    - The query begins a new SQL transaction.
    - The query attempts to update the user information using the stored procedure **proc_SecUpdateUser** (section 3.1.5.121).
    - The query attempts to update the user's properties in the site's user information list using the stored procedure **proc_UpdateListItem** (section 3.1.5.127).
    - 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 (2). This query is sent to the SQL server using TDS. On the SQL server the following actions occur:

    - The query begins a new SQL transaction.
    - The query attempts to update the site collection's user list data using the stored procedure **proc_UpdateListItem**.
    - 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 3.1.5.129).
    - 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.

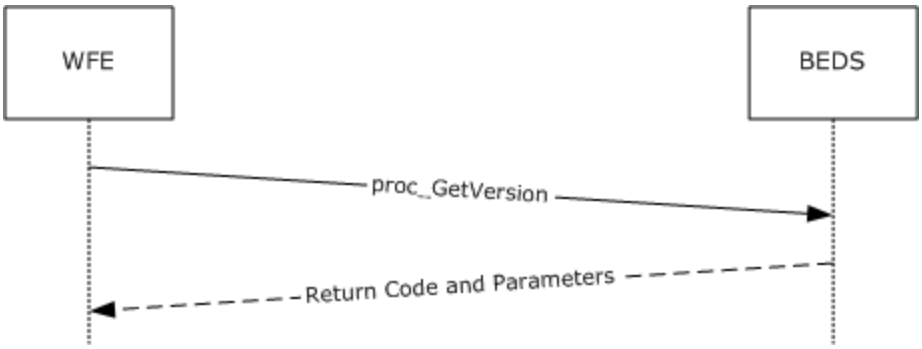


Figure 18: Version Negotiation

1. The WFE calls the **proc_GetVersion** (section 3.1.5.42) 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 '3.0.149.0'. The application has a pre-defined version of '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 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.

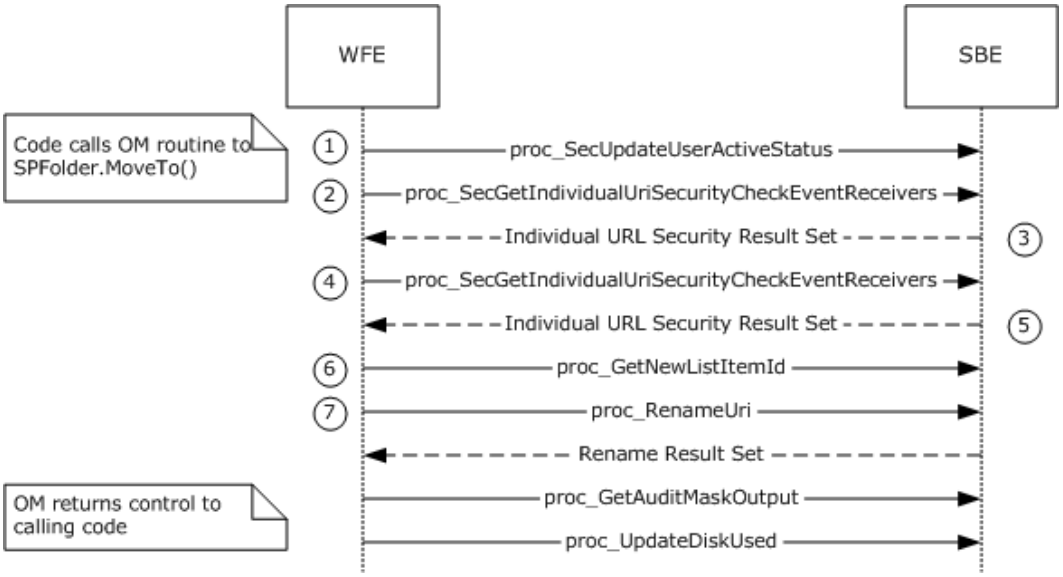


Figure 20: Folder: Move Folder OM

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** stored procedure.
2. The front-end Web server builds a dynamic query that invokes the **proc_SecGetIndividualUrlSecurityCheckEventReceivers** stored procedure against the source folder.
3. The back-end database server returns a return code of 0, and returns the following result sets:
   - **Individual URL Security Result Set**. 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 sets:
   - **Individual URL Security Result Set**. 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.
   - The query begins a new SQL transaction.
   - The query attempts to rename the URL for the folder using the **proc_RenameUrl** stored procedure. The back-end database server returns a return code of 0, and returns the **Rename Result Set**, which gives basic information about the old and new URLs for the renamed folder.
   - The query attempts to get **Audit Flags** information about the new folder using the **proc_GetAuditMaskOutput** stored procedure.
   - The query rolls back the SQL transaction if any of the previous procedures were not successful.
   - Assuming the previous procedures were successful, the query updates the disk spaced used for the **site (2)** 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	3.1.5.49
proc_SecAddRoleDef	3.1.5.50
proc_SecAddUser	3.1.5.51
proc_SecAddUserToSiteGroup	3.1.5.52
proc_SecAddWebMembership	3.1.5.53
proc_SecChangeToInheritedList	3.1.5.54
proc_SecChangeToInheritedWeb	3.1.5.55
proc_SecChangeToUniqueScope	3.1.5.56
proc_SecCheckDeletedAccounts	3.1.5.57
proc_SecCloneRoleDefinitions	3.1.5.58
proc_SecCreateSiteGroup	3.1.5.59
proc_SecDecCurrentUsersCount	3.1.5.60
proc_SecGetAccountStatus	3.1.5.61
proc_SecGetAclFromScope	3.1.5.62
proc_SecGetAllAclsForSite	3.1.5.63
proc_SecGetAllGroupsAndMembershipInfo	3.1.5.64
Proc_SecGetApplicationPrincipalAndUserToken	3.1.5.65
proc_SecGetCompleteWebRoleMemberList	3.1.5.66
proc_SecGetCurrentUsersCount	3.1.5.67
proc_SecGetDomainGroupMapData	3.1.5.68
proc_SecGetGroupById	3.1.5.69

Security Parameter	Section
proc_SecGetGroupOwner	3.1.5.70
proc_SecGetGroupSecurityScopes	3.1.5.71
proc_SecGetIndividualUrlSecurityCheckEventReceivers	3.1.5.72
Proc_SecGetItemsWithUniquePermissions	3.1.5.73
proc_SecGetPrincipalByEmail	3.1.5.74
proc_SecGetPrincipalById	3.1.5.75
Proc_SecGetPrincipalByIdEx	3.1.5.76
proc_SecGetPrincipalByLogin	3.1.5.77
proc_SecGetPrincipalByLogin20	3.1.5.78
proc_SecGetPrincipalDisplayInformation20	3.1.5.79
proc_SecGetRoleAssignments	3.1.5.80
proc_SecGetRoleBindingsForAllPrincipals	3.1.5.81
proc_SecGetRoleDefs	3.1.5.82
proc_SecGetSecurityInfo	3.1.5.83
proc_SecGetSiteAdmins	3.1.5.84
proc_SecGetSiteGroupById	3.1.5.85
proc_SecGetSiteGroupByTitle	3.1.5.86
proc_SecGetSiteGroupByTitle20	3.1.5.87
proc_SecGetUserAccountDirectoryPath	3.1.5.88
proc_SecGetUserPermissionOnGroup	3.1.5.89
Proc_SecGetWebsAndListsWithUniquePermissions	3.1.5.90
proc_SecListAllSiteMembers	3.1.5.91
proc_SecListAllWebMembers	3.1.5.92
proc_SecListGroupsInRole	3.1.5.93
proc_SecListScopeGroups	3.1.5.94
proc_SecListScopeUsers	3.1.5.95
proc_SecListSiteGroupMembership	3.1.5.96
proc_SecListSiteGroups	3.1.5.97
proc_SecListSiteGroupsContainingUser	3.1.5.98
proc_SecListSiteGroupsWhichUserOwns	3.1.5.99
proc_SecListUsersInRole	3.1.5.100
proc_SecMigrateUser	3.1.5.101

Security Parameter	Section
proc_SecReCalculateWebFGP	3.1.5.102
proc_SecRefreshToken	3.1.5.103
proc_SecRemoveGroup	3.1.5.104
proc_SecRemovePrincipalFromScope	3.1.5.106
proc_SecRemoveRoleDef	3.1.5.107
proc_SecRemoveUserFromScopeByLogin	3.1.5.108
proc_SecRemoveUserFromSite	3.1.5.109
proc_SecRemoveUserFromSiteGroup	3.1.5.110
proc_SecRemoveUserFromSiteGroupByLogin	3.1.5.111
proc_SecResetItemPerm	3.1.5.112
proc_SecResetWebToDefaultRoleDefinition	3.1.5.113
proc_SecResolvePrincipal	3.1.5.114
proc_SecSetSiteGroupProperties	3.1.5.115
proc_SecSetUserAccountDirectoryPath	3.1.5.116
proc_SecSetWebRequestAccess	3.1.5.117
proc_SecUpdateAnonymousPermMask	3.1.5.118
proc_SecUpdateDomainGroupMapData	3.1.5.119
proc_SecUpdateRoleDef	3.1.5.120
proc_SecUpdateUser	3.1.5.121
proc_SecUpdateUserActiveStatus	3.1.5.122

6 Appendix A: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include released service packs.

- Microsoft SharePoint Foundation 2013
- Microsoft SharePoint Server 2016

Exceptions, if any, are noted below. If a service pack or Quick Fix Engineering (QFE) number appears with the product version, behavior changed in that service pack or QFE. The new behavior also applies to subsequent service packs of the product unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

Unless otherwise specified, any statement of optional behavior in this specification that is prescribed using the terms "SHOULD" or "SHOULD NOT" implies product behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term "MAY" implies that the product does not follow the prescription.

<1> Section 2.2.1.1.10: For example, by default, SharePoint Foundation 2013 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 2013.

<3> Section 2.2.7.3.3.2: SharePoint Foundation 2013 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 2.2.7.3.3.2: SharePoint Foundation 2013 does include this attribute with a value of 0.

<5> Section 2.2.7.3.3.2: SharePoint Foundation 2013 sets the value MinusSign for this field.

<6> Section 3.1.5.42: SharePoint products use text-based SQL queries to get the version number which are equivalent in functionality to **proc_GetVersion**.

7 Change Tracking

This section identifies changes that were made to this document since the last release. Changes are classified as Major, Minor, or None.

The revision class **Major** means that the technical content in the document was significantly revised. Major changes affect protocol interoperability or implementation. Examples of major changes are:

- A document revision that incorporates changes to interoperability requirements.
- A document revision that captures changes to protocol functionality.

The revision class **Minor** means that the meaning of the technical content was clarified. Minor changes do not affect protocol interoperability or implementation. Examples of minor changes are updates to clarify ambiguity at the sentence, paragraph, or table level.

The revision class **None** means that no new technical changes were introduced. Minor editorial and formatting changes may have been made, but the relevant technical content is identical to the last released version.

The changes made to this document are listed in the following table. For more information, please contact dochelp@microsoft.com.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
2.2.1.2.3 Calendar Type	Added values 14 and 15 to the Calendar Type table.	N	New content added.

8 Index

N

O

P

R

S

T

U

V

W

X