[MS-ASDSP]:
Access Services Database Stored Procedures Protocol Specification

---

Intellectual Property Rights Notice for Open Specifications Documentation

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

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

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

Preliminary Documentation. This Open Specification provides documentation for past and current releases and/or for the pre-release (beta) version of this technology. This Open Specification is final documentation for past or current releases as specifically noted in the document, as applicable; it is preliminary documentation for the pre-release (beta) versions. Microsoft will release final documentation in connection with the commercial release of the updated or new version of this technology. As the documentation may change between this preliminary version and the final version of this technology, there are risks in relying on preliminary documentation. To the extent that you incur additional development obligations or any other costs as a result of relying on this preliminary documentation, you do so at your own risk.

Revision Summary

Date	Revision History	Revision Class	Comments
01/20/2012	0.1	New	Released new document.
04/11/2012	0.1	No change	No changes to the meaning, language, or formatting of the technical content.
07/16/2012	0.1	No change	No changes to the meaning, language, or formatting of the technical content.

Table of Contents

1 Introduction 6
1.1 Glossary 6
1.2 References 6
1.2.1 Normative References 7
1.2.2 Informative References 7
1.3 Overview 7
1.4 Relationship to Other Protocols 7
1.5 Prerequisites/Preconditions 8
1.6 Applicability Statement 8
1.7 Versioning and Capability Negotiation 8
1.8 Vendor-Extensible Fields 8
1.9 Standards Assignments 8

2 Messages 9
2.1 Transport 9
2.2 Common Data Types 9
2.2.1 Simple Data Types and Enumerations 9
2.2.1.1 ObjectIdentityTable 9
2.2.1.2 ObjectNameList 9
2.2.1.3 ObjectTypeNumber 9
2.2.1.4 QueryColumnsTable 10
2.2.1.5 TimeZoneDefinitions 10
2.2.1.6 TimeZoneRules 11
2.2.2 Bit Fields and Flag Structures 13
2.2.3 Binary Structures 13
2.2.4 Result Sets 14
2.2.4.1 GetExternalLinksAndObjectSchema.ResultSet0 14
2.2.4.2 GetExternalLinksAndObjectSchema.ResultSet1 14
2.2.4.3 GetObjects.ResultSet0 15
2.2.4.4 GetObjectSchema.ResultSet0 16
2.2.4.5 ObjectDefinitionSelect.ResultSet0 17
2.2.4.6 GetUserTableSchema.ResultSet0 17
2.2.4.7 GetUserTableSchema.ResultSet1 18
2.2.4.8 GetUserTableSchema.ResultSet2 19
2.2.4.9 QueryObject.ResultSet0 20
2.2.4.10 QueryObject.ResultSet1 21
2.2.4.11 GetObjectSchema.Sproc.ResultSet0 21
2.2.4.12 GetObjectSchema.Sproc.ResultSet1 21
2.2.4.13 GetObjectSchema.View.ResultSet0 22
2.2.4.14 GetObjectSchema.View.ResultSet1 22
2.2.4.15 QueryObjectUpdateOrderBy.ResultSet0 23
2.2.4.16 QueryObjectUpdateOrderBy.ResultSet1 24
2.2.5 Tables and Views 24
2.2.5.1 ApplicationProperties 25
2.2.5.2 ColumnProperties 25
2.2.5.3 GetTableColumns 25
2.2.5.4 ObjectDependencies 26
2.2.5.5 Objects 27
2.2.5.6 ObjectSelect 27
2.2.5.7 ObjectStorage 29

2.2.5.8 QueryColumns 29
2.2.5.9 TimeZoneDefinitionBase 30
2.2.5.10 TimeZoneRuleBase 31
2.2.5.11 Trace 33
2.2.5.12 GetDeletedObjects 34
2.2.5.13 GetDependentObjects 34
2.2.5.14 GetDependentQueries 34
2.2.5.15 GetRemotingSchema 35
2.2.5.16 GetSupportingObjects 36
2.2.5.17 GetUpdatedObjects 36
2.2.6 XML Structures 37
2.2.6.1 Namespaces 38
2.2.6.2 Simple Types 38
2.2.6.3 Complex Types 38
2.2.6.4 Elements 38
2.2.6.5 Attributes 38
2.2.6.6 Groups 38
2.2.6.7 Attribute Groups 38

3 Protocol Details 39
3.1 Common Details 39
3.2 Server Details 39
3.2.1 Abstract Data Model 39
3.2.2 Timers 39
3.2.3 Initialization 39
3.2.4 Higher-Layer Triggered Events 39
3.2.5 Message Processing Events and Sequencing Rules 39
3.2.5.1 ApplicationPropertiesDelete 39
3.2.5.2 ApplicationPropertiesInsert 40
3.2.5.3 ApplicationPropertiesUpdate 40
3.2.5.4 ApplicationPropertiesUpdateIf 41
3.2.5.5 ColumnPropertiesColumnRename 41
3.2.5.6 ColumnPropertiesDelete 42
3.2.5.7 ColumnPropertiesInsert 42
3.2.5.8 ColumnPropertiesUpdate 43
3.2.5.9 GetExternalLinksAndObjectSchema 44
3.2.5.10 GetObjects 45
3.2.5.11 GetObjectSchema 45
3.2.5.12 GetUserTableSchema 46
3.2.5.13 HandleError 47
3.2.5.14 LogActionTrace 48
3.2.5.15 ObjectDefinitionSelect 49
3.2.5.16 ObjectsDelete 49
3.2.5.17 ObjectsDeleteByObjectName 50
3.2.5.18 ObjectsInsert 50
3.2.5.19 ObjectStorageInsert 51
3.2.5.20 ObjectStorageUpdate 52
3.2.5.21 ObjectsUpdate 54
3.2.5.22 ObjectsUpdateProperties 55
3.2.5.23 PopulateTimeZoneData 55
3.2.5.24 QueryColumnsDelete 55
3.2.5.25 QueryColumnsInsert 56
3.2.5.26 QueryObjectInsert 56

3.2.5.27 QueryObjectUpdate 57
3.2.5.28 RaiseError 58
3.2.5.29 DatabaseCollationInfoSelect 59
3.2.5.30 ObjectsUpdateOrderBy 59
3.2.5.31 QueryObjectUpdateOrderBy 60
3.2.6 Timer Events 60
3.2.7 Other Local Events 60
3.3 Client Details 60
3.3.1 Abstract Data Model 60
3.3.2 Timers 60
3.3.3 Initialization 60
3.3.4 Higher-Layer Triggered Events 61
3.3.5 Message Processing Events and Sequencing Rules 61
3.3.6 Timer Events 61
3.3.7 Other Local Events 61

4 Protocol Examples 62
4.1 Create Table 62
4.2 Update Table 63
4.3 Delete Table 64

5 Security 65
5.1 Security Considerations for Implementers 65
5.2 Index of Security Parameters 65

6 Appendix A: Product Behavior 66

7 Change Tracking 67

8 Index 68

1 Introduction

The Access Services Database Stored Procedures Protocol enables a Web-based database application to manage its metadata on a server. This protocol provides a means of maintaining information about tables, forms, and any other objects and information that define the database. This protocol also provides the ability to create any new metadata when a new database application is provisioned.

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

1.1 Glossary

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

Coordinated Universal Time (UTC)
Security Support Provider Interface (SSPI)

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

back-end database server
column
computed field
database application
descending order
display name
expression
front-end Web server
MIME type
primary key
result set
return code
row
stored procedure
table-valued function
table-valued parameter
time zone

The following terms are specific to this document:

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

1.2 References

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

1.2.1 Normative References

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

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

[MS-AXL2] Microsoft Corporation, "Access Application Transfer Data Structure Version 2".

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

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

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, http://www.rfc-editor.org/rfc/rfc2119.txt

1.2.2 Informative References

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

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

1.3 Overview

This protocol defines tables, **table-valued functions**, and **stored procedures** that create and maintain the metadata for Web-based **database applications**. Protocol clients can add and remove information about the application and about objects in the application. A typical scenario for using this protocol is the addition of new metadata to a set of well-defined tables about a database application, such as when an object is added to the application.

1.4 Relationship to Other Protocols

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

This is shown in the following layered diagram:

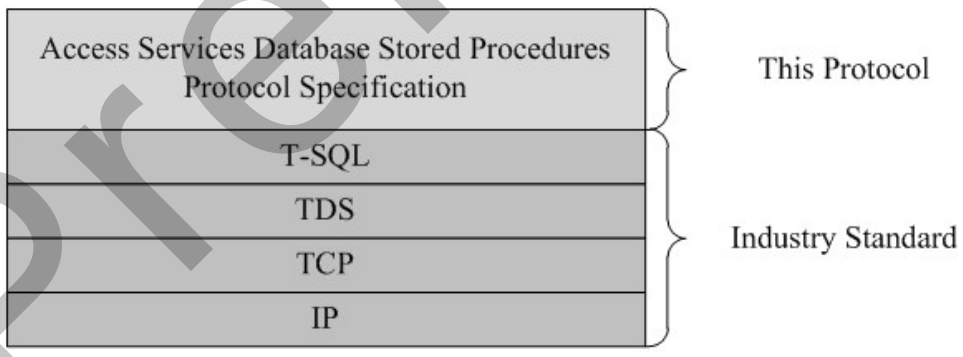

Figure 1: This protocol in relation to other protocols

1.5 Prerequisites/Preconditions

This protocol operates between a protocol client and a protocol server on which the back-end databases are stored. The protocol client is expected to know the location and connection information for the databases.

This protocol requires that the protocol client has the appropriate permissions to call the stored procedures in the back-end databases.

1.6 Applicability Statement

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

1.7 Versioning and Capability Negotiation

Security and Authentication Methods: This protocol supports the following authentication methods: **SSPI** and SQL Authentication. These authentication methods are defined in [MS-TDS].

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

The Tabular Data Stream Protocol ([MS-TDS]) MUST be used to call the stored procedures, fetch data from SQL tables, and return **result sets** and **return codes**.

2.2 Common Data Types

2.2.1 Simple Data Types and Enumerations

2.2.1.1 ObjectIdentityTable

A **table-valued parameter** that specifies a list of objects in a database application and their versions.

The **ObjectIdentityTable** type is defined using T-SQL syntax as follows:

```
ID int NOT NULL,
Version datetime2(6) NOT NULL
```

ID: Specifies the identifier of an object.

Version: Specifies a value that represents the version of the object.

2.2.1.2 ObjectNameList

A table-valued parameter that specifies a list of names of objects in the database application.

The **ObjectNameList** type is defined using T-SQL syntax as follows:

```
ObjectName nvarchar(128) NOT NULL
```

ObjectName: Specifies the name of an object in the database application.

2.2.1.3 ObjectTypeNumber

A number that specifies a type of object. MUST be one of the following values.

Value	Meaning
100	Table, query, or data macro
101	External data link
102	Form
103	UI macro
105	Image
111	Embedded expression

2.2.1.4 QueryColumnsTable

A table-valued parameter that specifies data for the **QueryColumns** table (section 2.2.5.8).

The **QueryColumnsTable** type is defined using T-SQL syntax as follows:

```
QueryName nvarchar(128) NULL,
QueryColumn nvarchar(128) NULL,
Definition nvarchar(256) NULL,
SimpleDefinition nvarchar(256) NULL,
DependentOn nvarchar(4) NULL,
IsUpdatable bit NULL,
Key nvarchar(4) NULL,
FKeyDetails nvarchar(256) NULL,
BaseTable nvarchar(128) NULL,
BaseColumn nvarchar(128) NULL,
BaseAlias nvarchar(128) NULL,
ParentTable nvarchar(128) NULL,
SelectAll bit NULL
```

QueryName: Specifies a value for the **QueryName column (1)** of the **QueryColumns** table.

QueryColumn: Specifies a value for the **QueryColumn** column (1) of the **QueryColumns** table.

Definition: Specifies a value for the **Definition** column (1) of the **QueryColumns** table.

SimpleDefinition: Specifies a value for the **SimpleDefinition** column (1) of the **QueryColumns** table.

DependentOn: Specifies a value for the **DependentOn** column (1) of the **QueryColumns** table.

IsUpdatable: Specifies a value for the **IsUpdatable** column (1) of the **QueryColumns** table.

Key: Specifies a value for the **Key** column (1) of the **QueryColumns** table.

FKeyDetails: Specifies a value for the **FKeyDetails** column (1) of the **QueryColumns** table.

BaseTable: Specifies a value for the **BaseTable** column (1) of the **QueryColumns** table.

BaseColumn: Specifies a value for the **BaseColumn** column (1) of the **QueryColumns** table.

BaseAlias: Specifies a value for the **BaseAlias** column (1) of the **QueryColumns** table.

ParentTable: Specifies a value for the **ParentTable** column (1) of the **QueryColumns** table.

SelectAll: Specifies a value for the **SelectAll** column (1) of the **QueryColumns** table.

2.2.1.5 TimeZoneDefinitions

A table-valued parameter that specifies data for the **TimeZoneDefinitionBase** table (section 2.2.5.9).

The **TimeZoneDefinitions** type is defined using T-SQL syntax as follows:

```
ModifiedOn datetime2(7) NULL,
TimeZoneCode int NOT NULL,
OrganizationId uniqueidentifier NULL,
```

```
TimeZoneDefinitionId uniqueidentifier NOT NULL,
CreatedOn datetime2(7) NULL,
Bias int NULL,
DaylightName nvarchar(100) NULL,
CreatedBy uniqueidentifier NULL,
UserInterfaceName nvarchar(100) NOT NULL,
StandardName nvarchar(100) NOT NULL,
ModifiedBy uniqueidentifier NULL,
CreatedOnBehalfBy uniqueidentifier NULL,
ModifiedOnBehalfBy uniqueidentifier NULL
```

ModifiedOn: Specifies a value for the **ModifiedOn** column (1) of the **TimeZoneDefinitionBase** table.

TimeZoneCode: Specifies a value for the **TimeZoneCode** column (1) of the **TimeZoneDefinitionBase** table.

OrganizationId: Specifies a value for the **OrganizationId** column (1) of the **TimeZoneDefinitionBase** table.

TimeZoneDefinitionId: Specifies a value for the **TimeZoneDefinitionId** column (1) of the **TimeZoneDefinitionBase** table.

CreatedOn: Specifies a value for the **CreatedOn** column (1) of the **TimeZoneDefinitionBase** table.

Bias: Specifies a value for the **Bias** column (1) of the **TimeZoneDefinitionBase** table.

DaylightName: Specifies a value for the **DaylightName** column (1) of the **TimeZoneDefinitionBase** table.

CreatedBy: Specifies a value for the **CreatedBy** column (1) of the **TimeZoneDefinitionBase** table.

UserInterfaceName: Specifies a value for the **UserInferfaceName** column (1) of the **TimeZoneDefinitionBase** table.

StandardName: Specifies a value for the **StandardName** column (1) of the **TimeZoneDefinitionBase** table.

ModifiedBy: Specifies a value for the **ModifiedBy** column (1) of the **TimeZoneDefinitionBase** table.

CreatedOnBehalfBy: Specifies a value for the **CreatedOnBehalfBy** column (1) of the **TimeZoneDefinitionBase** table.

ModifiedOnBehalfBy: Specifies a value for the **ModifiedOnBehalfBy** column (1) of the **TimeZoneDefinitionBase** table.

2.2.1.6 TimeZoneRules

A table-valued parameter that specifies data for the **TimeZoneRuleBase** table (section 2.2.5.10).

The **TimeZoneRules** type is defined using T-SQL syntax as follows:

```
ModifiedBy uniqueidentifier NULL,
```

```
StandardDay int NOT NULL,
ModifiedOn datetime2(7) NULL,
StandardMinute int NOT NULL,
StandardBias int NOT NULL,
StandardYear int NOT NULL,
DaylightMonth int NOT NULL,
StandardDayOfWeek int NOT NULL,
DaylightSecond int NOT NULL,
Bias int NOT NULL,
TimeZoneRuleVersionNumber int NOT NULL,
DaylightBias int NOT NULL,
StandardMonth int NOT NULL,
EffectiveDateTime datetime2(7) NOT NULL,
CreatedBy uniqueidentifier NULL,
DaylightHour int NOT NULL,
StandardHour int NOT NULL,
CreatedOn datetime2(7) NULL,
DaylightYear int NOT NULL,
StandardSecond int NOT NULL,
DaylightMinute int NOT NULL,
TimeZoneDefinitionId uniqueidentifier NOT NULL,
DaylightDayOfWeek int NOT NULL,
TimeZoneRuleId uniqueidentifier NOT NULL,
DaylightDay int NOT NULL,
OrganizationId uniqueidentifier NULL,
ModifiedOnBehalfBy uniqueidentifier NULL,
CreatedOnBehalfBy uniqueidentifier NULL
```

ModifiedBy: Specifies a value for the **ModifiedBy** column (1) of the **TimeZoneRuleBase** table.

StandardDay: Specifies a value for the **StandardDay** column (1) of the **TimeZoneRuleBase** table.

ModifiedOn: Specifies a value for the **ModifiedOn** column (1) of the **TimeZoneRuleBase** table.

StandardMinute: Specifies a value for the **StandardMinute** column (1) of the **TimeZoneRuleBase** table.

StandardBias: Specifies a value for the **StandardBias** column (1) of the **TimeZoneRuleBase** table.

StandardYear: Specifies a value for the **StandardYear** column (1) of the **TimeZoneRuleBase** table.

DaylightMonth: Specifies a value for the **DaylightMonth** column (1) of the **TimeZoneRuleBase** table.

StandardDayOfWeek: Specifies a value for the **StandardDayOfWeek** column (1) of the **TimeZoneRuleBase** table.

DaylightSecond: Specifies a value for the **DaylightSecond** column (1) of the **TimeZoneRuleBase** table.

Bias: Specifies a value for the **Bias** column (1) of the **TimeZoneRuleBase** table.

TimeZoneRuleVersionNumber: Specifies a value for the **TimeZoneRuleVersionNumber** column (1) of the **TimeZoneRuleBase** table.

DaylightBias: Specifies a value for the **DaylightBias** column (1) of the **TimeZoneRuleBase** table.

StandardMonth: Specifies a value for the **StandardMonth** column (1) of the **TimeZoneRuleBase** table.

EffectiveDateTime: Specifies a value for the **EffectiveDateTime** column (1) of the **TimeZoneRuleBase** table.

CreatedBy: Specifies a value for the **CreatedBy** column (1) of the **TimeZoneRuleBase** table.

DaylightHour: Specifies a value for the **DaylightHour** column (1) of the **TimeZoneRuleBase** table.

StandardHour: Specifies a value for the **StandardHour** column (1) of the **TimeZoneRuleBase** table.

CreatedOn: Specifies a value for the **CreatedOn** column (1) of the **TimeZoneRuleBase** table.

DaylightYear: Specifies a value for the **DaylightYear** column (1) of the **TimeZoneRuleBase** table.

StandardSecond: Specifies a value for the **StandardSecond** column (1) of the **TimeZoneRuleBase** table.

DaylightMinute: Specifies a value for the **DaylightMinute** column (1) of the **TimeZoneRuleBase** table.

TimeZoneDefinitionId: Specifies a value for the **TimeZoneDefinitionId** column (1) of the **TimeZoneRuleBase** table.

DaylightDayOfWeek: Specifies a value for the **DaylightDayOfWeek** column (1) of the **TimeZoneRuleBase** table.

TimeZoneRuleId: Specifies a value for the **TimeZoneRuleId** column (1) of the **TimeZoneRuleBase** table.

DaylightDay: Specifies a value for the **DaylightDay** column (1) of the **TimeZoneRuleBase** table.

OrganizationId: Specifies a value for the **OrganizationId** column (1) of the **TimeZoneRuleBase** table.

ModifiedOnBehalfBy: Specifies a value for the **ModifiedOnBehalfBy** column (1) of the **TimeZoneRuleBase** table.

CreatedOnBehalfBy: Specifies a value for the **CreatedOnBehalfBy** column (1) of the **TimeZoneRuleBase** table.

2.2.2 Bit Fields and Flag Structures

No common bit field or flag structures are defined in this protocol.

2.2.3 Binary Structures

No common binary structures are defined in this protocol.

2.2.4 Result Sets

This protocol specifies the following result sets.

2.2.4.1 GetExternalLinksAndObjectSchema.ResultSet0

Specifies all objects that represent external links in a database application requested by **GetExternalLinksAndObjectSchema** (section 3.2.5.9), where each **row (1)** in the result set specifies an object.

```
ObjectName nvarchar(128),
```

ObjectName: Specifies the name of an object.

2.2.4.2 GetExternalLinksAndObjectSchema.ResultSet1

Specifies the schema of a table that represents an external link in a database application requested by **GetExternalLinksAndObjectSchema** (section 3.2.5.9), where each row (1) in the result set specifies one column (1) in the table.

```
Name nvarchar(128),
Type nvarchar(128),
MaxLength smallint,
Precision tinyint,
Scale tinyint,
IsNullable bit,
IsIdentity bit,
IsComputed bit,
DefaultValue nvarchar(max),
IsPrimaryKey bit,
AccessProperties nvarchar(max),
```

Name: Specifies the name of the column (1).

Type: Specifies the back-end database server type of the column (1).

MaxLength: Specifies the maximum size, in bytes, of the column (1). A value of -1 specifies that the column (1) does not have a maximum size.

Precision: Specifies the precision of the column (1). MUST be NULL if the column (1) does not specify a precision.

Scale: Specifies the scale of the column (1). MUST be NULL if the column (1) does not specify a scale.

IsNullable: Specifies whether the column (1) can contain NULL values.

IsIdentity: Specifies whether the column (1) contains automatically generated unique values.

IsComputed: Specifies whether the values of the column (1) are automatically calculated based on a formula.

DefaultValue: Specifies the default value of the column (1).

IsPrimaryKey: Specifies whether the column (1) is a **primary key**.

AccessProperties: Specifies properties of the column (1).

2.2.4.3 GetObjects.ResultSet0

Specifies metadata about user-created objects in the database requested by **GetObjects** (section 3.2.5.10), where each row (1) in the result set specifies one object.

```
ID int,
ObjectName nvarchar(128),
ObjectTypeNumber int,
Description nvarchar(350),
Definition nvarchar(max),
OrderBy nvarchar(max),
LastModified datetime,
CreatedBy nvarchar(255),
ModifiedBy nvarchar(255),
ParentId int,
Attachment varbinary(max),
FilePath nvarchar(450),
ContentType nvarchar(50),
Contents varbinary(max),
TypeDescription nvarchar(2),
LastModifiedInSQL datetime,
SqlObjectId int,
```

ID: Specifies the unique identifier of the object.

ObjectName: Specifies the name of the object.

ObjectTypeNumber: An **ObjectTypeNumber** (section 2.2.1.3) that specifies the type of the object.

Description: Specifies a description of the object.

Definition: Specifies an XML definition of the object.

OrderBy: Specifies an XML definition of the ordering of data in the object.

LastModified: Specifies the most recent time that the object was updated.

CreatedBy: Specifies the user who created the object.

ModifiedBy: Specifies the user who last modified the object.

ParentId: Specifies the identifier, from the **ID** column of the **Objects** table (section 2.2.5.5), of the object in the database that is considered the parent of the object. If the object has no parent, then **ParentId** MUST be NULL.

Attachment: Specifies the binary attachment of the object.

FilePath: If not NULL, specifies the relative location of files pertaining to the object.

ContentType: If not NULL, specifies the **MIME type** of the object.

Contents: Specifies binary information about the object.

TypeDescription: Specifies a description of the type of the object. MUST be either NULL or one of the values specified by the **type** column of the **sys.objects** view ([MSDN-TSQL-Ref]).

LastModifiedInSQL: Specifies the date and time when the definition of the object was most recently modified. MUST be the value in the **LastModifiedInSQL** column (1) in the **ObjectStorage** table for objects with an **ObjectTypeNumber** of 102 or 103. MUST be determined by the back-end database server for objects with an **ObjectTypeNumber** of 100. MUST be NULL for all other objects.

SqlObjectId: Specifies the back-end database server object identifier for objects with an **ObjectTypeNumber** of 100 or 101. MUST be NULL for other objects.

2.2.4.4 GetObjectSchema.ResultSet0

Specifies metadata about the columns (1) of a table in a database application, requested by the **GetObjectSchema** stored procedure (section 3.2.5.11), where each row (1) in the result set specifies the metadata of one column (1).

MUST contain the same data as the **GetTableColumns** table-valued function (section 2.2.5.3) when its parameters are as follows:

- The value of the *@accessObjectId* parameter and the *@sqlObjectId* parameter are respectively equal to the values in the **ID** column (1) and the **SqlObjectId** column (1) of the **ObjectSelect** view (section 2.2.5.6), and the value in the **Name** column (1) of the **ObjectSelect** view is equal to the value in *@objectName* parameter in the call to **GetObjectSchema** stored procedure.

```
Name nvarchar(128),
Type nvarchar(128),
MaxLength smallint,
Precision tinyint,
Scale tinyint,
IsNullable bit,
IsIdentity bit,
IsComputed bit,
DefaultValue nvarchar(max),
IsPrimaryKey bit,
AccessProperties nvarchar(max),
```

Name: Specifies the value of the **Name** column (1) of the **GetTableColumns** table-valued function.

Type: Specifies the value of the **Type** column (1) of the **GetTableColumns** table-valued function.

MaxLength: Specifies the value of the **MaxLength** column (1) of the **GetTableColumns** table-valued function.

Precision: Specifies the value of the **Precision** column (1) of the **GetTableColumns** table-valued function.

Scale: Specifies the value of the **Scale** column (1) of the **GetTableColumns** table-valued function.

IsNullable: Specifies the value of the **IsNullable** column (1) of the **GetTableColumns** table-valued function.

IsIdentity: Specifies the value of the **IsIdentity** column (1) of the **GetTableColumns** table-valued function.

IsComputed: Specifies the value of the **IsComputed** column (1) of the **GetTableColumns** table-valued function.

DefaultValue: Specifies the value of the **DefaultValue** column (1) of the **GetTableColumns** table-valued function.

IsPrimaryKey: Specifies the value of the **IsPrimaryKey** column (1) of the **GetTableColumns** table-valued function.

AccessProperties: Specifies the value of the **AccessProperties** column (1) of the **GetTableColumns** table-valued function.

2.2.4.5 ObjectDefinitionSelect.ResultSet0

Specifies definitions for objects in a database application requested by **ObjectDefinitionSelect** (section 3.2.5.15), where each row (1) in the result set specifies the definition of one object.

The **ObjectDefinitionSelect.ResultSet0** result set is defined using T-SQL syntax as follows:

```
Definition nvarchar(max),
```

Definition: Specifies the definition of an object in the **Objects** table (section 2.2.5.5).

2.2.4.6 GetUserTableSchema.ResultSet0

Specifies information about a table in a database application requested by **GetUserTableSchema** (section 3.2.5.12). MUST contain exactly one row (1). That row (1) MUST contain data from the row (1) of the **ObjectSelect** view (section 2.2.5.6) where the following are true:

- The value of the **ObjectTypeNumber** column (1) is 100.
- The value of the **ObjectName** column (1) is equal to the value of the *@tableName* parameter passed to **GetUserTableSchema**.

The **GetUserTableSchema.ResultSet0** result set is defined using T-SQL syntax as follows:

```
ID int,
ObjectName nvarchar(128),
ObjectTypeNumber int,
Description nvarchar(350),
Definition nvarchar(max),
OrderBy nvarchar(max),
LastModified datetime,
CreatedBy nvarchar(255),
ModifiedBy nvarchar(255),
ParentId int,
Attachment varbinary(max),
FilePath nvarchar(450),
ContentType nvarchar(50),
Contents varbinary(max),
TypeDescription nvarchar(2),
LastModifiedInSQL datetime,
SqlObjectId int,
```

ID: Specifies the value of the **ID** column (1) of the **ObjectSelect** table.

ObjectName: Specifies the value of the **ObjectName** column (1) of the **ObjectSelect** table. MUST be equal to the value of the **@tableName** parameter passed to **GetUserTableSchema**.

ObjectTypeNumber: Specifies the value of the **ObjectTypeNumber** column (1) of the **ObjectSelect** table. MUST be 100.

Description: Specifies the value of the **Description** column (1) of the **ObjectSelect** table.

Definition: Specifies the value of the **Definition** column (1) of the **ObjectSelect** table.

OrderBy: Specifies the value of the **OrderBy** column (1) of the **ObjectSelect** table.

LastModified: Specifies the value of the **LastModified** column (1) of the **ObjectSelect** table.

CreatedBy: Specifies the value of the **CreatedBy** column (1) of the **ObjectSelect** table.

ModifiedBy: Specifies the value of the **ModifiedBy** column (1) of the **ObjectSelect** table.

ParentId: Specifies the value of the **ParentId** column (1) of the **ObjectSelect** table.

Attachment: Specifies the value of the **Attachment** column (1) of the **ObjectSelect** table.

FilePath: Specifies the value of the **FilePath** column (1) of the **ObjectSelect** table.

ContentType: Specifies the value of the **ContentType** column (1) of the **ObjectSelect** table.

Contents: Specifies the value of the **Contents** column (1) of the **ObjectSelect** table.

TypeDescription: Specifies the value of the **TypeDescription** column (1) of the **ObjectSelect** table.

LastModifiedInSQL: Specifies the value of the **LastModifiedInSQL** column (1) of the **ObjectSelect** table.

SqlObjectId: Specifies the value of the **SqlObjectId** column (1) of the **ObjectSelect** table.

2.2.4.7 GetUserTableSchema.ResultSet1

Specifies information about the columns (1) of a table in a database application requested by **GetUserTableSchema** (section 3.2.5.12). MUST contain the same data as the **GetTableColumns** table-valued function (section 2.2.5.3) when its parameters are as follows:

- The value of the *@accessObjectId* parameter is equal to the value in the **ID** column (1) returned by the **GetUserTableSchema.ResultSet0** result set (section 2.2.4.6) in the same call to **GetUserTableSchema**.
- The value of the *@sqlObjectId* parameter is equal to the value in the **SqlObjectId** column (1) returned by the **GetUserTableSchema.ResultSet0** result set (section 2.2.4.6) in the same call to **GetUserTableSchema**.

The **GetUserTableSchema.ResultSet1** result set is defined using T-SQL syntax as follows:

```
Name nvarchar(128),
Type nvarchar(128),
MaxLength smallint,
Precision tinyint,
Scale tinyint,
IsNullable bit,
```

```
IsIdentity bit,
IsComputed bit,
DefaultValue nvarchar(max),
IsPrimaryKey bit,
AccessProperties nvarchar(max),
```

Name: Specifies the value of the **Name** column (1) of the **GetTableColumns** table-valued function.

Type: Specifies the value of the **Type** column (1) of the **GetTableColumns** table-valued function.

MaxLength: Specifies the value of the **MaxLength** column (1) of the **GetTableColumns** table-valued function.

Precision: Specifies the value of the **Precision** column (1) of the **GetTableColumns** table-valued function.

Scale: Specifies the value of the **Scale** column (1) of the **GetTableColumns** table-valued function.

IsNullable: Specifies the value of the **IsNullable** column (1) of the **GetTableColumns** table-valued function.

IsIdentity: Specifies the value of the **IsIdentity** column (1) of the **GetTableColumns** table-valued function.

IsComputed: Specifies the value of the **IsComputed** column (1) of the **GetTableColumns** table-valued function.

DefaultValue: Specifies the value of the **DefaultValue** column (1) of the **GetTableColumns** table-valued function.

IsPrimaryKey: Specifies the value of the **IsPrimaryKey** column (1) of the **GetTableColumns** table-valued function.

AccessProperties: Specifies the value of the **AccessProperties** column (1) of the **GetTableColumns** table-valued function.

2.2.4.8 GetUserTableSchema.ResultSet2

Specifies information about indexes of a table in a database application requested by **GetUserTableSchema** (section 3.2.5.12), excluding the primary key index. Each row (1) in the result set specifies an index on the table specified by the value of the *@tableName* parameter passed to **GetUserTableSchema**.

The **GetUserTableSchema.ResultSet2** result set is defined using T-SQL syntax as follows:

```
IndexName nvarchar(128),
IsUniqueIndex bit,
ColumnName nvarchar(128),
IsDescendingKey bit,
```

IndexName: Specifies the name of the index.

IsUniqueIndex: Specifies whether the index requires all values in the column (1) specified by **ColumnName** to be unique.

ColumnName: Specifies column (1) of the table specified by the value of the *@tableName* parameter passed to **GetUserTableSchema** that is used as the key to the index. MUST match one of the values in the **Name** column (1) returned by the **GetUserTableSchema.ResultSet1** result set (section 2.2.4.7) in the same call to **GetUserTableSchema**.

IsDescendingKey: Specifies whether the index sorts the column (1) specified by **ColumnName** in **descending order**.

2.2.4.9 QueryObject.ResultSet0

Specifies metadata about each column (1) that is projected in each query in the database requested by **QueryObjectInsert** (section 3.2.5.26) and **QueryObjectUpdate** (section 3.2.5.27), where each row (1) in the result set specifies one column (1) in a query.

```
QueryName nvarchar(128),
QueryColumn nvarchar(128),
Definition nvarchar(256),
SimpleDefinition nvarchar(256),
DependentOn nvarchar(4),
IsUpdatable bit,
Key nvarchar(4),
FKeyDetails nvarchar(256),
BaseTable nvarchar(128),
BaseColumn nvarchar(128),
BaseAlias nvarchar(128),
ParentTable nvarchar(128),
SelectAll bit,
```

QueryName: The name of the query in which the column (1) is projected.

QueryColumn: The name of the column (1) in the query. MUST be the value of alias if the column (1) has been aliased in the query definition. MUST be the name of the projected column (1) otherwise.

Definition: Specifies a definition of the column (1).

SimpleDefinition: Specifies an alias to the **Definition**.

DependentOn: Specifies the value of **SimpleDefinition** of all columns on which this column (1) is dependent.

IsUpdatable: Specifies whether values in the column (1) in the underlying table can be changed.

Key: Specifies whether the column (1) in the underlying table is part of the primary key or a foreign key. If not NULL, MUST be "PKEY" or "FKEY".

FKeyDetails: Specifies the **Definition** of all the columns (1) that are the primary keys of tables to which the foreign key is a lookup. MUST be empty if the **Key** is not "FKEY".

BaseTable: The name of the underlying table from which the query column (1) is projected.

BaseColumn: The name of the column (1) in **BaseTable**.

BaseAlias: Specifies the alias of the **BaseTable** if it has been aliased in the query definition.

ParentTable: MUST be empty.

SelectAll: MUST be empty.

2.2.4.10 QueryObject.ResultSet1

Specifies all columns that are a primary key or that have a unique constraint set on them, where each row (1) in the result set represents a column (1) in a table.

```
TABLE_NAME nvarchar(128),
COLUMN_NAME nvarchar(128),
```

TABLE_NAME: The name of a table in the database.

COLUMN_NAME: The name of a column (1) in the table specified by **TABLE_NAME**, which is part of the primary key in that table or has a unique constraint set on it.

2.2.4.11 GetObjectSchema.Sproc.ResultSet0

Specifies metadata about the columns (1) of an object in a database application, requested by the **GetObjectSchema** stored procedure (section 3.2.5.11), where each row (1) in the result set specifies the metadata of one column (1).

```
Name nvarchar(128),
Type nvarchar(128),
MaxLength smallint,
IsNullable bit,
IsIdentity bit,
IsComputed bit,
DefaultValue nvarchar(max),
```

Name: Specifies the name of the column (1).

Type: Specifies the **data type** ([MSDN-TSQL-Ref] Data Types) of the column (1).

MaxLength: Specifies the maximum size, in bytes, of the column (1). A value of -1 specifies that the column (1) does not have a maximum size.

IsNullable: Specifies whether the column (1) can contain NULL values.

IsIdentity: Specifies whether the column (1) contains unique values.

IsComputed: Specifies whether the column (1) is a **computed field**.

DefaultValue: Specifies the default value of the column (1).

2.2.4.12 GetObjectSchema.Sproc.ResultSet1

Specifies metadata about the parameters of a table-valued function or a stored procedure in a database application, requested by the **GetObjectSchema** stored procedure (section 3.2.5.11), where each row (1) in the result set specifies the metadata of one parameter.

```
Name nvarchar(128),
Type nvarchar(128),
MaxLength smallint,
IsOutput bit,
```

Name: Specifies the name of the parameter.

Type: Specifies the **data type** ([MSDN-TSQL-Ref] Data Types) of the parameter.

MaxLength: Specifies the maximum size, in bytes, of the parameter. A value of -1 specifies that the parameter does not have a maximum size.

IsOutput: Specifies whether the parameter is an output parameter or not. A value of 1 specifies that the parameter is an output parameter and a value of 0 specifies otherwise.

2.2.4.13 GetObjectSchema.View.ResultSet0

Specifies metadata about the columns (1) of a query in a database application, requested by the **GetObjectSchema** stored procedure (section 3.2.5.11), where each row (1) in the result set specifies the metadata of one column (1).

```
Name nvarchar(128),
Type nvarchar(128),
MaxLength smallint,
IsNullable bit,
IsIdentity bit,
IsComputed bit,
DefaultValue nvarchar(max),
```

Name: Specifies the name of the column (1).

Type: Specifies the **data type** ([MSDN-TSQL-Ref] Data Types) of the column (1).

MaxLength: Specifies the maximum size, in bytes, of the column (1). A value of -1 specifies that the column (1) does not have a maximum size.

IsNullable: Specifies whether the column (1) can contain NULL values.

IsIdentity: Specifies whether the column (1) contains unique values.

IsComputed: Specifies whether the column (1) is a computed field.

DefaultValue: Specifies the default value of the column (1).

2.2.4.14 GetObjectSchema.View.ResultSet1

Specifies metadata about the columns (1) of a query in a database application, requested by the **GetObjectSchema** stored procedure (section 3.2.5.11), where each row (1) in the result set specifies the metadata of one column (1).

The values in the corresponding row (1) of the result set for each column (1) in the query MUST contain the same data in the **QueryColumns** table (section 2.2.5.8) or the **ColumnProperties** table (section 2.2.5.2) for that column (1) of the query, where a column (1) in a query is mapped across the **QueryColumns** table, the **Objects** table (section 2.2.5.5), and the **ColumnProperties** table as follows:

- The values of the **BaseTable** column (1) and the **BaseColumn** column (1) from the **QueryColumns** table respectively match the **ObjectName** column (1) from the **Objects** table and the **ColumnName** column (1) from the **ColumnProperties** table, where the value of the **ID** column (1) from the **Objects** table of that corresponding row (1) matches that of the **ObjectId** column (1) from the **ColumnProperties** table.

```
QueryName nvarchar(128),
QueryColumn nvarchar(128),
Definition nvarchar(256),
SimpleDefinition nvarchar(256),
DependentOn nvarchar(4),
IsUpdatable bit,
Key nvarchar(4),
FKeyDetails nvarchar(256),
BaseTable nvarchar(128),
BaseColumn nvarchar(128),
BaseAlias nvarchar(128),
ParentTable nvarchar(128),
SelectAll bit,
AccessProperties nvarchar(max),
```

QueryName: Specifies the value of the **QueryName** column (1) of the **QueryColumns** table.

QueryColumn: Specifies the value of the **QueryColumn** column (1) of the **QueryColumns** table.

Definition: Specifies the value of the **Definition** column (1) of the **QueryColumns** table.

SimpleDefinition: Specifies the value of the **SimpleDefinition** column (1) of the **QueryColumns** table.

DependentOn: Specifies the value of the **DependentOn** column (1) of the **QueryColumns** table.

IsUpdatable: Specifies the value of the **IsUpdatable** column (1) of the **QueryColumns** table.

Key: Specifies the value of the **Key** column (1) of the **QueryColumns** table

FKeyDetails: Specifies the value of the **FKeyDetails** column (1) of the **QueryColumns** table.

BaseTable: Specifies the value of the **BaseTable** column (1) of the **QueryColumns** table.

BaseColumn: Specifies the value of the **BaseColumn** column (1) of the **QueryColumns** table.

BaseAlias: Specifies the value of the **BaseAlias** column (1) of the **QueryColumns** table.

ParentTable: Specifies the value of the **ParentTable** column (1) of the **QueryColumns** table.

SelectAll: Specifies the value of the **SelectAll** column (1) of the **QueryColumns** table.

AccessProperties: Specifies the value of the **Properties** column (1) of the **ColumnProperties** table.

2.2.4.15 QueryObjectUpdateOrderBy.ResultSet0

Specifies metadata about each column (1) that is projected in each query in the database, as requested by **QueryObjectUpdateOrderBy** (section 3.2.5.31). Each row (1) in the result set specifies one column (1) in a query.

```
QueryName nvarchar(128),
QueryColumn nvarchar(128),
Definition nvarchar(256),
SimpleDefinition nvarchar(256),
DependentOn nvarchar(4),
IsUpdatable bit,
```

```
Key nvarchar(4),
FKeyDetails nvarchar(256),
BaseTable nvarchar(128),
BaseColumn nvarchar(128),
BaseAlias nvarchar(128),
ParentTable nvarchar(128),
SelectAll bit,
```

QueryName: The name of the query in which the column (1) is projected.

QueryColumn: The name of the column (1) in the query. MUST be the value of alias if the column (1) has been aliased in the query definition. MUST be the name of the projected column (1) otherwise.

Definition: Specifies a definition of the column (1).

SimpleDefinition: Specifies an alias to the **Definition**.

DependentOn: Specifies the value of **SimpleDefinition** of all columns on which this column (1) is dependent.

IsUpdatable: Specifies whether values in the column (1) in the underlying table can be changed.

Key: Specifies whether the column (1) in the underlying table is part of the primary key or a foreign key. If not NULL, MUST be "PKEY" or "FKEY".

FKeyDetails: Specifies the **Definition** of all the columns (1) that are the primary keys of tables to which the foreign key is a lookup. MUST be empty if the **Key** is not "FKEY".

BaseTable: The name of the underlying table from which the query column (1) is projected.

BaseColumn: The name of the column (1) in **BaseTable**.

BaseAlias: Specifies the alias of the **BaseTable** if it has been aliased in the query definition.

ParentTable: MUST be empty.

SelectAll: MUST be empty.

2.2.4.16 QueryObjectUpdateOrderBy.ResultSet1

Specifies all columns (1) that are a primary key or that have a unique constraint set on them, as requested by **QueryObjectUpdateOrderBy** (section 3.2.5.31). Each row (1) in the result set represents one column (1) in a table.

```
TABLE_NAME nvarchar(128),
COLUMN_NAME nvarchar(128),
```

TABLE_NAME: The name of a table in the database application.

COLUMN_NAME: The name of a column (1) in the table specified by **TABLE_NAME**, which is part of the primary key in that table or has a unique constraint set on it.

2.2.5 Tables and Views

This protocol specifies the following tables and views.

2.2.5.1 ApplicationProperties

The **ApplicationProperties** table contains string-based name/value pairs of properties that store application settings and configuration data.

The **ApplicationProperties** table is defined using T-SQL syntax as follows:

```
ID int NOT NULL,
PropertyName nvarchar(128) NOT NULL,
PropertyValue nvarchar(max) NOT NULL,
```

ID: The identifier of the name/value pair. MUST be unique among all rows (1) in the table.

PropertyName: The name of a property.

PropertyValue: The value of the property specified by **PropertyName**.

2.2.5.2 ColumnProperties

The **ColumnProperties** table contains values for the fields used in a database application. The properties are associated with fields in user tables of a database application. Properties are stored as blocks of text that are stored in the table without any manipulation.

The ColumnProperties table is defined using T-SQL syntax as follows:

```
ObjectId int NOT NULL,
ColumnName nvarchar(128) NOT NULL,
Properties nvarchar(max) NULL,
ExtendedAttributes nvarchar(max) NULL,
```

ObjectId: The identifier of the table that contains **ColumnName**.

ColumnName: The name of a field in a user table in the database, which is specified by **ObjectId**.

Properties: Properties of the column that is specified by the combination of **ObjectId** and **ColumnName**.

ExtendedAttributes: Properties of the column that is specified by the combination of **ObjectId** and **ColumnName**.

2.2.5.3 GetTableColumns

Specifies metadata about all columns (1) in a table in a database application. The table object is identified by the table identifier in the database application, as specified by *@accessObjectID*, and its back-end database server object identifier, as specified by *@sqlObjectId*. Each row (1) of the result set specifies metadata of one column (1) in the table.

CREATE FUNCTION GetTableColumns(@accessObjectId, @sqlObjectId)

```
Name sysname NULL,
Type sysname NOT NULL,
MaxLength smallint NOT NULL,
Precision tinyint NOT NULL,
Scale tinyint NOT NULL,
IsNullable bit NULL,
```

```
IsIdentity bit NOT NULL,
IsComputed bit NOT NULL,
DefaultValue nvarchar(max) NULL,
IsPrimaryKey bit NULL,
AccessProperties nvarchar(max) NULL,
```

@accessObjectId: Specifies the identifier of the table object whose column (1) metadata is being queried.

@sqlObjectId: Specifies the back-end database server object identifier of the table object whose column (1) metadata is being queried.

Name: Specifies the name of the column (1).

Type: Specifies the **data type** ([MSDN-TSQL-Ref] Data Types) of the column (1).

MaxLength: Specifies the maximum size, in bytes, of the column. A value of -1 specifies that the column does not have a maximum size.

Precision: If the value of **Type** is the string 'datetime2', specifies the **scale** ([MSDN-TSQL-Ref] Data Types) of the column (1). Otherwise specifies the **precision** ([MSDN-TSQL-Ref] Data Types) of the column (1).

Scale: Specifies the **scale** ([MSDN-TSQL-Ref] Data Types) of the column (1).

IsNullable: Specifies whether the column (1) can contain NULL values.

IsIdentity: Specifies whether the column (1) contains unique values.

IsComputed: Specifies whether the column (1) is a computed field.

DefaultValue: Specifies the default value of the column (1).

IsPrimaryKey: Specifies whether the column (1) is a primary key.

AccessProperties: Specifies other properties of the column (1), as specified by the **Properties** column (1) in the **ColumnProperties** table (section 2.2.5.2).

2.2.5.4 ObjectDependencies

The **ObjectDependencies** table contains information about objects from the **Objects** table (section 2.2.5.5) that are dependent on other objects and the names of the objects that support them.

```
ID int NOT NULL,
ObjectID int NOT NULL,
SupportingObjectName nvarchar(128) NOT NULL,
```

ID: The identifier of the object dependency. MUST be unique among all rows (1) in the table.

ObjectID: Specifies the **ID** column (1) from the **Objects** table of an object that is dependent on another object.

SupportingObjectName: Specifies the name of the object that supports the object referenced by **ObjectID**.

2.2.5.5 Objects

The **Objects** table contains metadata about all the objects in the database application. Each row (1) specifies metadata about an object in the database application.

```
ID int NOT NULL,
ObjectName nvarchar(128) NOT NULL,
ObjectTypeNumber int NOT NULL,
Description nvarchar(350) NULL,
Definition nvarchar(max) NULL,
OrderBy nvarchar(max) NULL,
LastModified datetime2 NOT NULL,
CreatedBy nvarchar(255) NULL,
ModifiedBy nvarchar(255) NULL,
ParentId int NULL,
Attachment varbinary(max) NULL,
Properties nvarchar(max) NULL,
```

ID: Specifies an identifier that uniquely identifies the object within the database application. MUST be unique among all rows (1) in the table.

ObjectName: Specifies the name of the object.

ObjectTypeNumber: Specifies the **ObjectTypeNumber** (section 2.2.1.3) of the object.

Description: Specifies a description of the object.

Definition: Specifies an XML representation of the definition of the object.

OrderBy: Specifies an XML representation of the ordering of the columns (1) in the object. MUST be NULL for objects that are not a view.

LastModified: Specifies the date and time when the schema of the object was most recently modified.

CreatedBy: Specifies the user that created the object.

ModifiedBy: Specifies the user that last modified the object after the object was created.

ParentId: For objects that have a parent object, specifies the **ID** of the parent object in the database application. MUST be NULL if the object has no parent.

Attachment: Specifies a binary attachment for the object.

Properties: Specifies other properties of the object.

2.2.5.6 ObjectSelect

The **ObjectSelect** view specifies user-defined and system-defined metadata about all the objects specified in the **Objects** table (section 2.2.5.5), where each row (1) specifies metadata about one object.

```
ID int NOT NULL,
ObjectName nvarchar(128) NOT NULL,
ObjectTypeNumber int NOT NULL,
Description nvarchar(350) NULL,
Definition nvarchar(max) NULL,
```

```
OrderBy nvarchar(max) NULL,
LastModified datetime2 NOT NULL,
CreatedBy nvarchar(255) NULL,
ModifiedBy nvarchar(255) NULL,
ParentId int NULL,
Attachment varbinary(max) NULL,
FilePath nvarchar(450) NULL,
ContentType nvarchar(50) NULL,
Contents varbinary(max) NULL,
TypeDescription char(2) NULL,
LastModifiedInSQL datetime2 NULL,
SqlObjectId int NULL,
```

ID: Specifies the identifier of the object in the database application, as specified by the **ID** column (1) in the **Objects** table.

ObjectName: Specifies the name of the object, as specified by the **ObjectName** column (1) in the **Objects** table.

ObjectTypeNumber: Specifies the **ObjectTypeNumber** (section 2.2.1.3) of the object, as specified by the **ObjectTypeNumber** column (1) in the **Objects** table.

Description: Specifies the description of the object, as specified by the **Description** column (1) in the **Objects** table.

Definition: Specifies the definition of the object as specified by the **Definition** column (1) in the **Objects** table.

OrderBy: Specifies the ordering of the columns (1) in the object, as specified by the **OrderBy** column (1) in the **Objects** table.

LastModified: Specifies the date and time when the schema of the object was most recently modified, as specified by the **LastModified** column (1) in the **Objects** table.

CreatedBy: Specifies the user that created the object, as specified by the **CreatedBy** column (1) in the **Objects** table.

ModifiedBy: Specifies the user that last modified the object after the object was created, as specified by the **ModifiedBy** column (1) in the **Objects** table.

ParentId: Specifies the **ID** of the parent of the object, as specified by the **ParentId** column (1) in the **Objects** table.

Attachment: Specifies the binary attachment of the object, as specified by the **Attachment** column (1) in the **Objects** table.

FilePath: Specifies the file path of the object, as specified by the **FilePath** column (1) in the **ObjectStorage** table. MUST be NULL if the object does not have metadata in the **ObjectStorage** table (section 2.2.5.7).

ContentType: Specifies the content type of the object, as specified by the **ContentType** column (1) in the **ObjectStorage** table. MUST be NULL if the object does not have metadata in the **ObjectStorage** table.

Contents: Specifies the contents of the object in binary, as specified by the **Contents** column (1) in the **ObjectStorage** table. MUST be NULL if the object does not have metadata in the **ObjectStorage** table.

TypeDescription: Specifies the back-end database server type description for objects with **ObjectTypeNumber** as 100. MUST be NULL for other values of **ObjectTypeNumber**.

LastModifiedInSQL: Specifies the date and time when the definition of the object was most recently modified, as specified by the **LastModifiedInSQL** column (1) in the **ObjectStorage** table, for objects with an **ObjectTypeNumber** of 102 or 103, and as determined by the back-end database server for objects with an **ObjectTypeNumber** of 100. MUST be NULL for all other objects.

SqlObjectId: Specifies the back-end database server object identifier for objects with **ObjectTypeNumber** as 100 or 101. MUST be NULL for other values of **ObjectTypeNumber**.

2.2.5.7 ObjectStorage

The **ObjectStorage** table specifies metadata for Forms (objects with **ObjectTypeNumber** (section 2.2.1.3) as 102) and UI Macros (objects with **ObjectTypeNumber** as 103) in the database application, in addition to the metadata specified in the **Objects** table (section 2.2.5.5). Each row (1) specifies metadata about one such object.

```
ObjectID int NOT NULL,
FilePath nvarchar(450) NOT NULL,
ContentType nvarchar(50) NOT NULL,
Contents varbinary(max) NOT NULL,
LastModifiedInSQL datetime2 NOT NULL,
```

ObjectID: Specifies the identifier of the object in the database application, as specified by the **ID** column (1) in the **Objects** table.

FilePath: Specifies the file path of the object.

ContentType: Specifies the content type of the object.

Contents: Specifies the contents of the object in binary.

LastModifiedInSQL: Specifies the date and time when the definition of the object was most recently modified, as noted by the back-end database server.

2.2.5.8 QueryColumns

Specifies metadata about each column (1) that is projected in each query in the database, where each row (1) in the table specifies one column (1) in a query.

```
QueryName nvarchar(128) NULL,
QueryColumn nvarchar(128) NULL,
Definition nvarchar(256) NULL,
SimpleDefinition nvarchar(256) NULL,
DependentOn nvarchar(4) NULL,
IsUpdatable bit NULL,
Key nvarchar(4) NULL,
FKeyDetails nvarchar(256) NULL,
BaseTable nvarchar(128) NULL,
BaseColumn nvarchar(128) NULL,
BaseAlias nvarchar(128) NULL,
ParentTable nvarchar(128) NULL,
SelectAll bit NULL,
```

QueryName: The name of the query in which the column (1) is projected.

QueryColumn: The name of the column (1) in the query. MUST be the value of alias if the column (1) has been aliased in the query definition. MUST be the name of the projected column (1) otherwise.

Definition: Definition of the column (1) in the query. MUST be the name of the column (1) if it is projected from another data source or MUST be the string representation of the expression if it is an expression.

SimpleDefinition: An alias to the **Definition**. If the **Definition** is a column (1) name the **SimpleDefinition** has a unique string value starting with @ character. This value MUST be unique for each column (1) projected in a query. If the **Definition** is an expression the **SimpleDefinition** is the string representation of the expression where each of the columns (1) used in this expression is replaced with its corresponding **SimpleDefinition** value.

DependentOn: Specifies the **SimpleDefinition** of all the query columns (1) which this query column (1) is dependent on. If there are multiple values they are separated by comma.

IsUpdatable: Specifies whether values in the column (1) in the underlying table can be changed.

Key: Specifies whether this column (1) is part of the primary key or a foreign key. MUST be either "PKEY" or "FKEY".

FKeyDetails: Specifies the **Definition** of all the columns (1) that are the primary keys to tables which the foreign key is a lookup. If there are multiple values they are separated by comma. If **Key** is not "FKEY" this value MUST be empty.

BaseTable: The name of the underlying table or query from which the query column (1) is projected.

BaseColumn: The name of the column (1) in **BaseTable**.

BaseAlias: Specifies the alias of **BaseTable** if it has been aliased in the query definition.

ParentTable: MUST be empty.

SelectAll: MUST be empty.

2.2.5.9 TimeZoneDefinitionBase

The **TimeZoneDefinitionBase** table contains information about the **time zone** of the database application. This table MUST contain exactly one row (1).

```
ModifiedOn datetime2 NULL,
TimeZoneCode int NOT NULL,
OrganizationId uniqueidentifier NULL,
TimeZoneDefinitionId uniqueidentifier NOT NULL,
CreatedOn datetime2 NULL,
Bias int NULL,
DaylightName nvarchar(100) NULL,
CreatedBy uniqueidentifier NULL,
UserInterfaceName nvarchar(100) NOT NULL,
StandardName nvarchar(100) NOT NULL,
ModifiedBy uniqueidentifier NULL,
CreatedOnBehalfBy uniqueidentifier NULL,
ModifiedOnBehalfBy uniqueidentifier NULL,
```

ModifiedOn: Specifies the date and time that the row (1) was last modified.

TimeZoneCode: MUST be 0.

OrganizationId: MUST be NULL.

TimeZoneDefinitionId: Specifies an identifier for the time zone. MUST be equal to the **TimeZoneDefinitionId** column (1) in all rows (1) of the **TimeZoneRuleBase** (section 2.2.5.10) table.

CreatedOn: Specifies the date and time that the row (1) was added to the table.

Bias: MUST be NULL.

DaylightName: Specifies a name for the time zone used during daylight saving time.

CreatedBy: MUST be NULL.

UserInterfaceName: Specifies a **display name** for the time zone.

StandardName: Specifies a name for the time zone used during standard time.

ModifiedBy: MUST be NULL.

CreatedOnBehalfBy: MUST be NULL.

ModifiedOnBehalfBy: MUST be NULL.

2.2.5.10 TimeZoneRuleBase

The **TimeZoneRuleBase** table contains the daylight saving time rules for a time zone. Each row (1) in the table specifies a single rule which specifies the period of time the rule is in effect, the points in time at which daylight saving time starts and ends each year, and the offsets from **Coordinated Universal Time (UTC)** in effect during standard time and daylight saving time.

```
ModifiedBy uniqueidentifier NULL,
StandardDay int NOT NULL,
ModifiedOn datetime2 NULL,
StandardMinute int NOT NULL,
StandardBias int NOT NULL,
StandardYear int NOT NULL,
DaylightMonth int NOT NULL,
StandardDayOfWeek int NOT NULL,
DaylightSecond int NOT NULL,
Bias int NOT NULL,
TimeZoneRuleVersionNumber int NOT NULL,
DaylightBias int NOT NULL,
StandardMonth int NOT NULL,
EffectiveDateTime datetime2 NOT NULL,
CreatedBy uniqueidentifier NULL,
DaylightHour int NOT NULL,
StandardHour int NOT NULL,
CreatedOn datetime2 NULL,
DaylightYear int NOT NULL,
StandardSecond int NOT NULL,
DaylightMinute int NOT NULL,
TimeZoneDefinitionId uniqueidentifier NOT NULL,
DaylightDayOfWeek int NOT NULL,
```

```
TimeZoneRuleId uniqueidentifier NOT NULL,
DaylightDay int NOT NULL,
OrganizationId uniqueidentifier NULL,
ModifiedOnBehalfBy uniqueidentifier NULL,
CreatedOnBehalfBy uniqueidentifier NULL,
```

ModifiedBy: MUST be NULL.

StandardDay: Specifies the week of the month in which standard time begins. MUST be between 1 and 5. A value of 5 specifies the last week of the month.

ModifiedOn: Specifies the date and time that the row (1) was last modified.

StandardMinute: Specifies the minutes portion of the time of day at which standard time begins. MUST be between 0 and 59.

StandardBias: MUST be 0.

StandardYear: MUST be 0.

DaylightMonth: Specifies the month in which daylight saving time begins. MUST be between 1 and 12, where 1 specifies January.

StandardDayOfWeek: Specifies the day of the week on which standard time begins. MUST be between 0 and 6, where 0 specifies Sunday.

DaylightSecond: Specifies the seconds portion of the time of day at which daylight saving time begins. MUST be between 0 and 59.

Bias: Specifies the number of minutes to subtract from Coordinated Universal Time (UTC) to obtain the local time in this time zone.

TimeZoneRuleVersionNumber: MUST be 0.

DaylightBias: Specifies the number of minutes to subtract from standard time to obtain daylight saving time.

StandardMonth: Specifies the month in which standard time begins. MUST be between 1 and 12, where 1 specifies January.

EffectiveDateTime: Specifies the date and time that the rule takes effect.

CreatedBy: MUST be NULL.

DaylightHour: Specifies the hours portion of the time of day at which daylight saving time begins. MUST be between 0 and 23.

StandardHour: Specifies the hours portion of the time of day at which standard time begins. MUST be between 0 and 23.

CreatedOn: Specifies the date and time that the row (1) was added to the table.

DaylightYear: MUST be 0.

StandardSecond: Specifies the seconds portion of the time of day at which standard time begins. MUST be between 0 and 59.

DaylightMinute: Specifies the minutes portion of the time of day at which daylight saving time begins. MUST be between 0 and 59.

TimeZoneDefinitionId: Specifies the identifier for the time zone this rule applies to. MUST be equal to the **TimeZoneDefinitionId** column (1) of the only row (1) in the **TimeZoneDefinitionBase** (section 2.2.5.9) table.

DaylightDayOfWeek: Specifies the day of the week on which daylight saving time begins. MUST be between 0 and 6, where 0 specifies Sunday.

TimeZoneRuleId: Specifies an identifier for the rule.

DaylightDay: Specifies the week of the month in which daylight saving time begins. MUST be between 1 and 5. A value of 5 specifies the last week of the month.

OrganizationId: MUST be NULL.

ModifiedOnBehalfBy: MUST be NULL.

CreatedOnBehalfBy: MUST be NULL.

2.2.5.11 Trace

The **Trace** table contains event logs about running data macros ([MS-AXL2] section 2.1.2.2) in a database application. If the string 'DataMacroTracing' exists in the **PropertyName** column in a row (1) in the **ApplicationProperties** table (section 2.2.5.1), an entry will be logged into the **Trace** table for each data macro action ([MS-AXL2] section 2.2.5.1) being executed, and for any error encountered. In the **Objects** table (section 2.2.5.5), there MUST be an entry for the **Trace** table with ID column (1) equal to 1.

The **Trace** table is defined using T-SQL syntax as follows:

```
ID int NOT NULL,
MacroName nvarchar(128) NULL,
ActionName nvarchar(128) NULL,
Operand nvarchar(max) NULL,
Output nvarchar(max) NULL,
TargetRow nvarchar(max) NULL,
Timestamp datetime2 NOT NULL,
RuntimeErrorMessage nvarchar(max) NULL,
```

ID: The identifier of the trace entry. MUST be unique among all rows (1) in the table.

MacroName: The name of the data macro being executed. For embedded data macros, the hosting table name MUST also be included.

ActionName: The name of the data macro action being executed.

Operand: The operand of the data macro action.

Output: The output of the data macro action.

TargetRow: The information about the row (1) being targeted by the data macro action.

Timestamp: The execution time stamp of the data macro action.

RuntimeErrorMessage: The error message of the runtime error.

2.2.5.12 GetDeletedObjects

Specifies the subset of identifiers from a provided list that do not have entries in the **Objects** table (section 2.2.5.5).

The GetDeletedObjects table-valued function is defined using T-SQL syntax as follows:

CREATE FUNCTION GetDeletedObjects(@ClientObjects)

```
ID int NOT NULL,
```

@ClientObjects: An **ObjectIdentityTable** (section 2.2.1.1) that specifies identifiers and versions of objects.

ID: An identifier from the list provided in *@ClientObjects* that does not exist in the **Objects** table.

2.2.5.13 GetDependentObjects

Specifies objects in a database application from the **Objects** table (section 2.2.5.5) that depend on the object provided. Object A is dependent on Object B if any of the following are true:

- There is a row (1) in the **ObjectDependencies** table (section 2.2.5.4) with the value in the **SupportingObjectName** column (1) equal to the name of Object B and the **ObjectID** column(1) equal to the identifier in the **Objects** table of Object A.
- There is a row (1) in the **ObjectDependencies** table with the value in the **SupportingObjectName** column (1) equal to the name of Object B and the **ObjectID** column(1) equal to the identifier in the **Objects** table of an object that Object A depends on.

The **GetDependentObjects** table-valued function is defined using T-SQL syntax as follows:

CREATE FUNCTION GetDependentObjects(@accessObjectName)

```
ID int NULL,
DependentID int NULL,
NestingLevel int NULL,
```

@accessObjectName: Specifies the name of an object in a database application.

ID: Specifies the **ID** column (1) from the **Objects** table of the object specified by **@accessObjectName**.

DependentID: Specifies the **ID** column (1) from the **Objects** table of an object that is dependent on the object specified by **@accessObjectName**.

NestingLevel: Specifies the level of dependency.

2.2.5.14 GetDependentQueries

Specifies objects in a database application from the **Objects** table (section 2.2.5.5) that depend on the object provided. **ObjectTypeNumber** value (section 2.2.1.3) in **Objects** table corresponding to each of these objects MUST be 100 and the type column in sys.objects table [MSDN-TSQL-Ref] corresponding to each of these objects MUST be either "V" or "TF".

Object A is dependent on Object B if any of the following are true:

- There is a row (1) in the **ObjectDependencies** table (section 2.2.5.4) with the value in the **SupportingObjectName** column (1) equal to the name of Object B and the **ObjectID** column(1) equal to the identifier in the **Objects** table of Object A.
- There is a row (1) in the **ObjectDependencies** table with the value in the **SupportingObjectName** column (1) equal to the name of Object B and the **ObjectID** column(1) equal to the identifier in the **Objects** table of an object that Object A depends on.

The **GetDependentQueries** table-valued function is defined using T-SQL syntax as follows:

CREATE FUNCTION GetDependentQueries(@accessObjectName)

```
ID int NOT NULL,
NestingLevel int NULL,
```

@accessObjectName: Specifies the name of an object in a database application.

ID: Specifies the **ID** column (1) from the **Objects** table of an object that is dependent on the object specified by **@accessObjectName**.

NestingLevel: Specifies the level of dependency.

2.2.5.15 GetRemotingSchema

Specifies objects in a database application from the **Objects** table (section 2.2.5.5) that qualifies both of the following conditions:

- The Object supports one of the objects specified in **@accessObjectNames**. For more information on objects supporting, see section 2.2.5.16.
- The Object itself or any of its supporting objects is a linked table ([MS-AXL2] section 2.1.1.6).

The **GetRemotingSchema** table-valued function is defined using T-SQL syntax as follows:

CREATE FUNCTION GetRemotingSchema(@accessObjectNames)

```
ID int NOT NULL,
ObjectName nvarchar(128) NOT NULL,
Definition nvarchar(max) NULL,
LinkedTable int NOT NULL,
```

@accessObjectNames: An **ObjectNameList** (section 2.2.1.2) that specifies names of objects in a database application.

ID: Specifies the **ID** column (1) from the **Objects** table of an object that qualifies the conditions.

ObjectName: Specifies the **ObjectName** column (1) from the **Objects** table of an object that qualifies the conditions.

Definition: Specifies the **Definition** column (1) from the **Objects** table of an object that qualifies the conditions.

LinkedTable: Specifies whether an object that qualifies the conditions is a linked table. If the value is 1, then the object is a linked table, otherwise it is not.

2.2.5.16 GetSupportingObjects

Specifies objects in a database application from the **Objects** table (section 2.2.5.5) that supports objects in a list of objects provided. Object A is specified to support Object B if any of the following are true:

- Object A is the same object as Object B, and there is an entry for them in the **Objects** table.
- There is a row (1) in the **ObjectDependencies** table (section 2.2.5.4) with the value in the **SupportingObjectName** column (1) equal to the name of Object A and the **ObjectID** column (1) equal to the identifier in the **Objects** table of Object B.
- There is a row (1) in the **ObjectDependencies** table with the value in the **SupportingObjectName** column (1) equal to the name of Object A and the **ObjectID** column (1) equal to the identifier in the **Objects** table of an object that supports Object B.

The **GetSupportingObjects** table-valued function is defined using T-SQL syntax as follows:

CREATE FUNCTION GetSupportingObjects(@accessObjectNames)

```
ID int NULL,
ParentID int NULL,
```

@accessObjectNames: An **ObjectNameList** (section 2.2.1.2) that specifies names of objects in a database application.

ID: Specifies the **ID** column (1) from the **Objects** table of an object that supports the object referenced by **ParentID**.

ParentID: Specifies the **ID** column (1) from the **Objects** table of one of the objects specified by *@accessObjectNames*.

2.2.5.17 GetUpdatedObjects

Specifies objects in a database application, which, given a list of identifiers and versions of objects, are either not in the list or have different versions than those in the list. Each row (1) of the result set specifies data corresponding to a row (1) from the **ObjectSelect** table (section 2.2.5.6) where either:

- The value of the **ID** column (1) of the **ObjectSelect** table does not match any value of the **ID** column (1) in the *@ClientObjects* parameter.
- The value of the **ModifiedBy** column (1) of the **ObjectSelect** table does not match the value of the **Version** column (1) in the row (1) of *@ClientObjects* where the value of the **ID** column (1) of the **ObjectSelect** table matches the value of the **ID** column (1) of *@ClientObjects*.

The **GetUpdatedObjects** table-valued function is defined using T-SQL syntax as follows:

CREATE FUNCTION GetUpdatedObjects(@ClientObjects)

```
ClientHas bit NOT NULL,
ID int NOT NULL,
ObjectName nvarchar(128) NOT NULL,
ObjectTypeNumber int NOT NULL,
Description nvarchar(350) NULL,
Definition nvarchar(max) NULL,
```

```
OrderBy nvarchar(max) NULL,
LastModified datetime2 NOT NULL,
CreatedBy nvarchar(255) NULL,
ModifiedBy nvarchar(255) NULL,
ParentID int NULL,
Attachment varbinary(max) NULL,
FilePath nvarchar(450) NULL,
ContentType nvarchar(50) NULL,
Contents varbinary(max) NULL,
TypeDescription char(2) NULL,
LastModifiedInSQL datetime2 NULL,
SQLObjectID int NULL,
```

@ClientObjects: An **ObjectIdentityTable** (section 2.2.1.1) that specifies identifiers and versions of objects.

ClientHas: Specifies whether the value of **ID** matches one of the values from the **ID** column (1) of **@ClientObjects**.

ID: Specifies a value of the **ID** column (1) of the **ObjectSelect** table.

ObjectName: Specifies a value of the **ObjectName** column (1) of the **ObjectSelect** table.

ObjectTypeNumber: Specifies a value of the **ObjectTypeNumber** column (1) of the **ObjectSelect** table.

Description: Specifies a value of the **Description** column (1) of the **ObjectSelect** table.

Definition: Specifies a value of the **Definition** column (1) of the **ObjectSelect** table.

OrderBy: Specifies a value of the **OrderBy** column (1) of the **ObjectSelect** table.

LastModified: Specifies a value of the **LastModified** column (1) of the **ObjectSelect** table.

CreatedBy: Specifies a value of the **CreatedBy** column (1) of the **ObjectSelect** table.

ModifiedBy: Specifies a value of the **ModifiedBy** column (1) of the **ObjectSelect** table.

ParentID: Specifies a value of the **ModifiedBy** column (1) of the **ObjectSelect** table.

Attachment: Specifies a value of the **Attachment** column (1) of the **ObjectSelect** table.

FilePath: Specifies a value of the **FilePath** column (1) of the **ObjectSelect** table.

ContentType: Specifies a value of the **ContentType** column (1) of the **ObjectSelect** table.

Contents: Specifies a value of the **Contents** column (1) of the **ObjectSelect** table.

TypeDescription: Specifies a value of the **TypeDescription** column (1) of the **ObjectSelect** table.

LastModifiedInSQL: Specifies a value of the **LastModifiedInSQL** column (1) of the **ObjectSelect** table.

SQLObjectID: Specifies a value of the **SQLObjectID** column (1) of the **ObjectSelect** table.

2.2.6 XML Structures

This specification does not define any common XML structures.

2.2.6.1 Namespaces

This specification does not define any common XML schema namespaces.

2.2.6.2 Simple Types

This specification does not define any common XML schema simple type definitions.

2.2.6.3 Complex Types

This specification does not define any common XML schema complex type definitions.

2.2.6.4 Elements

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

2.2.6.5 Attributes

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

2.2.6.6 Groups

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

2.2.6.7 Attribute Groups

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

3 Protocol Details

3.1 Common Details

None.

3.2 Server Details

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 specification does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this specification.

The protocol server MUST maintain metadata about objects in the database application and the application itself. The server MUST keep a mapping of dependent objects; for example, a query is dependent on the tables from which it projects columns (1). For objects that store tabular data, such as tables and queries, the protocol server MUST maintain additional metadata about each column (1) in the object. The server MUST keep a mapping of dependency information between columns in the tabular data as well; for example, a query column that projects a calculation is dependent on the columns that are used in the **expression**.

3.2.2 Timers

None.

3.2.3 Initialization

None.

3.2.4 Higher-Layer Triggered Events

None.

3.2.5 Message Processing Events and Sequencing Rules

This section provides information on the stored procedures that are part of a database application.

3.2.5.1 ApplicationPropertiesDelete

The **ApplicationPropertiesDelete** stored procedure removes rows (1) from the **ApplicationProperties** table (section 2.2.5.1). The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE ApplicationPropertiesDelete (
@name nvarchar(128)
,@value nvarchar(max)
);
```

@name: The name of a property. If the value of *@name* is NULL or does not match any values in the **PropertyName** column, then no operation is performed.

@value: The value of the property specified by *@name*. If *@value* is not NULL, the row (1) where both *@name* and *@value* match the contents of **ProperyName** and **PropertyValue**, respectively, MUST be deleted. If *@value* is NULL, all rows (1) in which the contents of the **PropertyName** column match *@name* MUST be deleted.

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

Value	Description
@@rowcount	The number of rows (1) that were affected by running this procedure.

Result Sets: MUST NOT return any result sets.

3.2.5.2 ApplicationPropertiesInsert

The **ApplicationPropertiesInsert** stored procedure inserts a row (1) into the **ApplicationProperties** table (section 2.2.5.1). The T-SQL syntax for the stored procedure is as follows:

```
PROCEDURE ApplicationPropertiesInsert (
@name nvarchar(128)
,@value nvarchar(max)
);
```

@name: The name of a property to enter in a new row (1) in the **ApplicationsProperties** table in the PropertyName column (1).

@value: The value of the property specified by @name to enter in a new row (1) in the **ApplicationsProperties** table in the PropertyValue column (1).

Return Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.2.5.3 ApplicationPropertiesUpdate

The **ApplicationPropertiesUpdate** stored procedure updates the contents of the **ApplicationProperties** table (section 2.2.5.1).

If *@name* is NULL, then the table MUST NOT be updated. If *@value* is NULL, then all rows (1) in the table in which the value of **PropertyName** is the same as *@name* MUST be deleted.

If neither *@name* nor *@propertyValue* is NULL, then the value of **PropertyValue** in all rows (1) of the table in which *@name* is the same as the value of **PropertyName** MUST be set to *@value*. If there are no matching rows (1) in the table, then a new row (1) MUST be inserted into the table in which **ID** is unique within the table, **PropertyName** is set to *@name*, and **PropertyValue** is set to *@value*.

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

```
PROCEDURE ApplicationPropertiesUpdate (
@name nvarchar(128)
,@value nvarchar(max)
);
```

@name: The name of a property.

@value: The value of the property specified by *@name*.

Return Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.2.5.4 ApplicationPropertiesUpdateIf

The **ApplicationPropertiesUpdateIf** stored procedure updates the contents of **PropertyValue** in a row (1) in the **ApplicationProperties** table (section 2.2.5.1).

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

```
PROCEDURE ApplicationPropertiesUpdateIf (
@name nvarchar(128)
,@originalValue nvarchar(max)
,@newValue nvarchar(max)
);
```

@name: The name of the property to update. If *@name* is NULL or if there are no rows (1) in the **ApplicationProperties** table in which *@name* matches the value of the **PropertyName** column (1), then the table MUST NOT be updated.

@originalValue: The existing value of the property identified by *@name*. If this value does not match the value of **PropertyValue** in the row (1) or rows (1) of the **ApplicationProperties** table in which *@name* matches the value of **PropertyName**, then the table MUST NOT be updated. Otherwise, the value of **PropertyValue** in each matching row (1) MUST be set to *@newValue*.

@newValue: The new value of the property.

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

Value	Description
@@rowcount	Returns the number of rows (1) in the **ApplicationProperties** table that were updated by the procedure.

Result Sets: MUST NOT return any result sets.

3.2.5.5 ColumnPropertiesColumnRename

The **ColumnPropertiesColumnRename** stored procedure updates the name of a field in the **ColumnProperties** table (section 2.2.5.2). The value of **ColumnName** in the **ColumnProperties** table MUST be updated to *@newColumnName* in the row (1) in which *@objectId* and *@oldColumnName* match the values of the columns **ObjectId** and **ColumnName**, respectively. If there is no matching row (1), then no action is taken.

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

```
PROCEDURE ColumnPropertiesColumnRename (
@objectId int
,@oldColumnName nvarchar(128)
,@newColumnName nvarchar(128)
```

```
);
```

@objectId: The identifier of the table that contains the column specified by *@oldColumnName*.

@oldColumnName: The current name of the field in the table.

@newColumnName: The new name of the field to write to the table.

Return Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.2.5.6 ColumnPropertiesDelete

The **ColumnPropertiesDelete** stored procedure removes a row (1) in the **ColumnProperties** table (section 2.2.5.2) when it finds a match for *@objectId* and *@columnName*. If either *@objectId* or *@columnName* is NULL or if no match is found, then no change is made. Otherwise, the row (1) in **ColumnProperties** in which *@objectId* and *@columnName* match the values of the **ObjectId** and **ColumnName** columns, respectively, MUST be deleted.

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

```
PROCEDURE ColumnPropertiesDelete (
@objectId int
,@objectName nvarchar(128)
,@columnName nvarchar(128)
);
```

@objectId: The identifier of the table that contains *@columnName*.

@objectName: MUST NOT be used.

@columnName: The name of a field in the table identified by *@objectId*.

Return Values: An integer that MUST be 0.

Result Sets: MUST NOT return any result sets.

3.2.5.7 ColumnPropertiesInsert

The **ColumnPropertiesInsert** stored procedure inserts column properties into the **ColumnProperties** table (section 2.2.5.2).

A row (1) MUST be inserted into **ColumnProperties** and the contents of *@objectId*, *@columnName, @properties* and *@extendedAttributes* MUST be set in that row (1).

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

```
PROCEDURE ColumnPropertiesInsert (
@objectId int
,@objectName nvarchar(128)
,@columnName nvarchar(128)
,@properties nvarchar(max)
,@extendedAttributes nvarchar(max)
```

```
);
```

@objectId: The identifier of the table in the **Objects** table (section 2.2.5.5) that contains the column (1) named *@columnName*. MUST NOT be NULL.

@objectName: MUST be ignored.

@columnName: The name of a column (1) in the table identified by *@objectId*. MUST NOT be NULL.

@properties: Properties of the column (1).

@extendedAttributes: Properties of the column (1).

Return Values: An integer that MUST be 0.

Result Sets: MUST NOT return any result sets.

3.2.5.8 ColumnPropertiesUpdate

The **ColumnPropertiesUpdate** stored procedure updates the **ColumnProperties** table (section 2.2.5.2).

If *@properties* and *@extendedAttributes* are both NULL, then the row (1) in **ColumnProperties** in which the contents of *@objectId* and *@columnName* match values in the **ObjectId** and **ColumnName** columns, respectively, MUST be deleted.

If either *@properties* or *@extendedAttributes* is not NULL, then the row (1) in **ColumnProperties** in which the contents of *@objectId* and *@columnName* match values in the **ObjectId** and **ColumnName** columns, respectively, MUST be updated with the new values of both *@properties* and *@extendedAttributes*. If there is no match for *@objectId* and *@columnName* in the table, then a row (1) MUST be inserted and the contents of *@objectId*, *@columnName, @properties* and *@extendedAttributes* MUST be set in that row (1).

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

```
PROCEDURE ColumnPropertiesUpdate (
@objectId int
,@objectName nvarchar(128)
,@columnName nvarchar(128)
,@properties nvarchar(max)
,@extendedAttributes nvarchar(max)
);
```

@objectId: The identifier of the table that contains *@columnName*. MUST NOT be null.

@objectName: MUST be ignored.

@columnName: The name of a field whose properties will be added, updated, or deleted in a row (1) in the table. MUST NOT be null.

@properties: A set of properties of a field to be added or updated in a row (1) in the table.

@extendedAttributes: A set of properties of a field to be added or updated in a row (1) in the table.

Return Values: An integer that MUST be 0.

Result Sets: MUST NOT return any result sets.

3.2.5.9 GetExternalLinksAndObjectSchema

The **GetExternalLinksAndObjectSchema** stored procedure gets a list of all objects that represent external links in a database application, as well as schema information for database application objects. The results depend on the type of the object.

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

```
PROCEDURE GetExternalLinksAndObjectSchema (
@objectName nvarchar(128)
,@typeDescription nvarchar(2) OUTPUT
,@orderBy nvarchar(max) OUTPUT
);
```

@objectName: The name of the object for which to get information. MUST NOT be NULL.

@typeDescription: A description of the type of the object, which MUST be returned as an output parameter.

Value	Description
objectselect.typedescription	MUST be one of the values specified by the **type** parameter of the **sys.objects** view ([MSDN-TSQL-Ref]).

@orderBy: Specifies the sort order.

Value	Description
objectselect.orderby	MUST be NULL if *@typeDescription* is not "V"; otherwise, MUST be the value of the **OrderBy** column in the **Objects** table (section 2.2.5.5) in the row (1) that specifies *@objectName*.

Error code values:

Value	Description
cannot find object.	If *@objectName* does not specify an object with an **ObjectTypeNumber** (section 2.2.1.3) of 100 or 101, then an error MUST be raised to the user, as specified by **RaiseError** (section 3.2.5.28), with an error number of 50003.

Return Values: An integer which MUST be 0.

Result Sets:

This stored procedure MUST return a GetExternalLinksAndObjectSchema.ResultSet0

If *@typeDescription* is "U":

This stored procedure MUST return a GetExternalLinksAndObjectSchema.ResultSet1

If *@typeDescription* is "V":

This stored procedure MUST return a GetObjectSchema.View.ResultSet0

This stored procedure MUST return a GetObjectSchema.View.ResultSet1

If *@typeDescription* is "TF", "IF" or "P":

This stored procedure MUST return a GetObjectSchema.Sproc.ResultSet0

This stored procedure MUST return a GetObjectSchema.Sproc.ResultSet1

3.2.5.10 GetObjects

The **GetObjects** stored procedure returns metadata about user-created objects in the database. If *@objectId* is not NULL, the result set MUST include the object for which *@objectId* matches the value of **ID** in the **ObjectSelect** (section 2.2.5.6) view and all its child objects (that is, all objects in which *@objectId* matches the value of **ParentId**). If *@objectID* is NULL, the result set MUST contain all objects.

The result set is ordered by **ParentId** and then **ID**.

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

```
PROCEDURE GetObjects (
@objectId int
);
```

@objectId: Specifies the parent object to return.

Return Values: An integer that MUST be 0.

Result Sets:

This stored procedure MUST return a GetObjects.ResultSet0

3.2.5.11 GetObjectSchema

The **GetObjectSchema** stored procedure gets schema information for database application objects. The results depend on the type of the object.

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

```
PROCEDURE GetObjectSchema (
@objectName nvarchar(128)
,@typeDescription nvarchar(2) OUTPUT
,@orderBy nvarchar(max) OUTPUT
);
```

@objectName: The name of an object in the database.

@typeDescription: A description of the type of the object, which MUST be returned as an output parameter.

Value	Description
objectselect.typedescription	MUST be one of the values specified by the **type** parameter of the

Value	Description
	sys.objects view ([MSDN-TSQL-Ref]).

@orderBy: Specifies the sort order.

Value	Description
objectselect.orderby	MUST be NULL if *@typeDescription* is not "V"; otherwise, MUST be the value of the **OrderBy** column in the **Objects** table (section 2.2.5.5) in the row (1) that specifies *@objectName*.

Error code values:

Value	Description
cannot find object.	If *@objectName* does not specify an object with an **ObjectTypeNumber** (section 2.2.1.3) of 100 or 101, then an error MUST be raised to the user, as specified by **RaiseError** (section 3.2.5.28), with an error number of 50003.

Return Values: An integer that MUST be 0.

Result Sets:

If *@typeDescription* is "U", this stored procedure MUST return: GetObjectSchema.ResultSet0

If *@typeDescription* is "V":

This stored procedure MUST return a GetObjectSchema.View.ResultSet0

This stored procedure MUST return a GetObjectSchema.View.ResultSet1

If *@typeDescription* is "TF", "IF" or "P":

This stored procedure MUST return a GetObjectSchema.Sproc.ResultSet0

This stored procedure MUST return a GetObjectSchema.Sproc.ResultSet1

3.2.5.12 GetUserTableSchema

The **GetUserTableSchema** stored procedure returns column information for a user-created table in the database application.

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

```
PROCEDURE GetUserTableSchema (
@tableName nvarchar(128)
);
```

@tableName: The name of the table in the database application for which to get information.

Error code values:

Value	Description
table does not exist	*@tableName* was NULL or did not specify a user-created table in the database. The error has a severity of 16 and a state of 1.

Return Values: An integer which MUST be 0.

Result Sets:

If the procedure did not raise an error, the following result sets MUST be returned.

This stored procedure MUST return a GetUserTableSchema.ResultSet0

This stored procedure MUST return a GetUserTableSchema.ResultSet1

This stored procedure MUST return a GetUserTableSchema.ResultSet2

3.2.5.13 HandleError

The **HandleError** stored procedure is called when an error occurs while a data macro is running. For more information about data macros, see [MS-AXL2] section 2.1.2.2.

The database MUST preserve all the entries in the **Trace** table (section 2.2.5.11) that had an **ID** greater than *@_dm_lastLoggingID* when this procedure was called. If *@_dm_initialTranCount* is greater than 0, then the current transaction MUST be rolled back to *@_dm_savePoint*; otherwise, the entire transaction MUST be rolled back.

If *@errorNumber* is 50000, an error MUST be raised to the caller with a **msg_str** of *@errorMessage*, a **severity** of 16 and a **state** of *@errorState* (RAISEERROR [MSDN-TSQL-Ref]). If *@errorNumber* is not 50000, then an error MUST be raised to the caller as specified by **RaiseError** (section 3.2.5.28), where the *@errorNumber* and *@errorMessage* values of this function are handled in the same way as the parameters of the same name in **RaiseError**.

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

```
PROCEDURE HandleError (
@_dm_initialTranCount int
,@_dm_lastLoggingID int
,@_dm_savePoint varchar(36)
,@_dm_macroRunning nvarchar(128)
,@_dm_actionRunning nvarchar(128)
,@errorNumber int
,@errorMessage nvarchar(4000)
,@errorState int
,@_dm_traceOn bit
);
```

@_dm_initialTranCount: Specifies the initial transaction count when the caller of this procedure was called.

@_dm_lastLoggingID: Specifies the identifier of the most recent record in the **Trace** table before this procedure was called. Used to mark the boundary between existing **Trace** entries and the new Trace entries that will be rolled back.

@_dm_savePoint: Specifies the transaction save point to roll back to if *@_dm_initialTranCount* is greater than 0.

@_dm_macroRunning: Specifies the name of the data macro that was running when the error occurred.

@_dm_actionRunning: Specifies the action of the data macro that was running when the error occurred. For more information on data macro actions, see [MS-AXL2] section 2.2.5.1.

@errorNumber: The SQL exception number. A value of 50000 specifies that the error was intentionally raised by the data macro. For more information on the **RaiseError** data macro action, see [MS-AXL2] section 2.2.5.1.10.

@errorMessage: The error message that will be returned to the caller.

@errorState: The SQL exception state.

@_dm_traceOn: Specifies whether the procedure logs the error to the **Trace** table. If the value is 1 and *@errorNumber* is not 50000, *@_dm_macroRunning* MUST be logged to the **MacroName** column, *@_dm_actionRunning* MUST be logged to the **ActionName** column, and *@errorMessage* MUST be logged to the **RuntimeErrorMessage** column. Otherwise, the error MUST NOT be logged to the **Trace** table.

Error code values:

Value	Description
@errormessage	The error message that is returned to the caller of the procedure.

Return Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.2.5.14 LogActionTrace

The **LogActionTrace** stored procedure inserts a row (1) into the **Trace** table (section 2.2.5.11) that indicates a failure during execution of a **data macro** ([MS-AXL2] section 2.1.2.2). The procedure ignores any error during the logging to prevent any logging errors from propagating and interrupting the data macro execution error in progress.

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

```
PROCEDURE LogActionTrace (
@macroName nvarchar(128)
,@actionName nvarchar(128)
,@operand nvarchar(4000)
,@output nvarchar(max)
,@targetRow nvarchar(4000)
,@errorMsg nvarchar(4000)
);
```

@macroName: The name of the running macro.

@actionName: The name of the running action.

@operand: The operand of the running action.

@output: The output of the running action.

@targetRow: Captures the value of fields that were involved on the targeting row (1).

@errorMsg: The SQL exception error message.

Return Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.2.5.15 ObjectDefinitionSelect

The **ObjectDefinitionSelect** stored procedure returns the definition of an object in the **Objects** table (section 2.2.5.5). If either *@objectName* or *@objectTypeNumber* is NULL, then the result set will be empty. Otherwise, the result set MUST contain the contents of the **Definition** column (1) in the **Objects** table for the row (1) in which *@objectName* matches the value in the **ObjectName** column (1) and *@objectTypeNumber* matches the value in the **ObjectTypeNumber** column (1).

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

```
PROCEDURE ObjectDefinitionSelect (
@objectName nvarchar(128)
,@objectTypeNumber int
);
```

@objectName: The name of an object in the **Objects** table.

@objectTypeNumber: An **ObjectTypeNumber** (section 2.2.1.3) that specifies the type of the object.

Return Values: An integer which MUST be 0.

Result Sets:

This stored procedure MUST return a ObjectDefinitionSelect.ResultSet0

3.2.5.16 ObjectsDelete

The **ObjectsDelete** stored procedure deletes row (1) from the **Objects** table (section 2.2.5.5).

If *@id* matches a value of **ID** in the **Objects** table, the row (1) in which the value matches MUST be deleted.

If *@id* is not found or, in the row (1) that matches, the value of the **LastModified** column (1) does not match the contents of *@lastModified*, then the table MUST NOT be updated and an error MUST be raised to the caller as specified by **RaiseError** (section 3.2.5.28), where the *@errorNumber* is 50003 and *@errorMessage* is text specifying a delete conflict.

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

```
PROCEDURE ObjectsDelete (
@id int
,@lastModified datetime2
);
```

@id: The unique identifier of an object in the **Objects** table.

@lastModified: A timestamp for the last modification of an object.

Return Values: An integer that MUST be 0.

Result Sets: MUST NOT return any result sets.

3.2.5.17 ObjectsDeleteByObjectName

The **ObjectsDeleteByObjectName** stored procedure deletes rows (1) from the **Objects** table (section 2.2.5.5). All rows (1) in the table in which the value in the **ObjectName** column matches *@objectName* MUST be deleted. If there are no rows (1) with matching values, then the table MUST NOT be updated.

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

```
PROCEDURE ObjectsDeleteByObjectName (
@objectName nvarchar(128)
);
```

@objectName: The name of an object in the **Objects** table.

Return Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.2.5.18 ObjectsInsert

The **ObjectsInsert** stored procedure inserts an object into the **Objects** table (section 2.2.5.5) and its dependencies into the **ObjectDependencies** table (section 2.2.5.4). The identifier of the newly inserted object is returned in the *@id* output parameter.

For objects with an *@objectTypeNumber* of "100", the initial value of the **LastModified** column MUST be the last modified date of the object in the database. For all other objects, the initial value of the **LastModified** column MUST be the current time when the procedure is called.

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

```
PROCEDURE ObjectsInsert (
@objectName nvarchar(128)
,@objectTypeNumber int
,@definition nvarchar(max)
,@description nvarchar(350)
,@parentId int
,@createdBy nvarchar(255)
,@orderBy nvarchar(max) = null
,@supportingObjects ObjectNameList
,@id int OUTPUT
);
```

@objectName: The name of the object, which MUST be inserted into the **ObjectName** column. MUST NOT be NULL.

@objectTypeNumber: An **ObjectTypeNumber** (section 2.2.1.3) that specifies the type of the object, which MUST be inserted into the **ObjectTypeNumber** column.

@definition: The XML definition of the object, which MUST be inserted into the **Definition** column.

@description: A description of the object, which MUST be inserted into the **Description** column.

@parentId: The identifier of the parent object, which MUST be inserted into the **ParentId** column.

@createdBy: The name or identifier of the user who created the object, which MUST be inserted into the **CreatedBy** column.

@orderBy: An XML representation of the sorting of the object, which MUST be inserted into the **OrderBy** column.

@supportingObjects: An **ObjectNameList** (section 2.2.1.2) that specifies the dependent objects of the object being inserted. Each object in this list MUST be inserted into the **ObjectDependencies** table (section 2.2.5.4) unless doing so would create a circular dependency. If a circular dependency would be created, then objects MUST NOT be inserted and a circular dependency error MUST be raised. A circular dependency would be created for a dependent object if *@id* matches the value of the **ParentId** column in the row (1) in the **ObjectDependencies** table that represents the dependent object.

@id: The **ID** of the newly inserted object in the **Objects** table, which MUST be returned as an output parameter.

Value	Description
objects.scope_identity	The identifier of the newly inserted object.

Return Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.2.5.19 ObjectStorageInsert

The **ObjectStorageInsert** stored procedure inserts an object into the **Objects** table (section 2.2.5.5) and its dependencies into the **ObjectDependencies** table (section 2.2.5.4). The identifier of the newly inserted object is returned in the *@id* output parameter. The procedure also inserts additional information about the object into the **ObjectStorage** table (section 2.2.5.7).

For objects with an *@objectTypeNumber* of "100", the initial value of the **LastModified** column MUST be the last modified date of the object in the database. For all other objects, the initial value of the **LastModified** column MUST be the current time when the procedure is called.

The initial value of the **LastModifiedInSql** column in the **ObjectStorage** table MUST be "0001-01-01 00:00:00".

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

```
PROCEDURE ObjectStorageInsert (
@objectName nvarchar(128)
,@objectTypeNumber int
,@definition nvarchar(max)
,@description nvarchar(350)
,@parentId int
,@createdBy nvarchar(255)
,@filePath nvarchar(450)
,@contentType nvarchar(50)
,@contents varbinary(max)
```

```
,@supportingObjects ObjectNameList
,@id int OUTPUT
);
```

@objectName: The name of the object, which MUST be inserted into the **ObjectName** column of the **Objects** table. MUST NOT be NULL.

@objectTypeNumber: An **ObjectTypeNumber** (section 2.2.1.3) that specifies the type of the object, which MUST be inserted into the **ObjectTypeNumber** column of the **Objects** table.

@definition: The XML definition of the object, which MUST be inserted into the **Definition** column of the **Objects** table.

@description: A description of the object, which MUST be inserted into the **Description** column of the **Objects** table.

@parentId: The identifier of the parent object, which MUST be inserted into the **ParentId** column of the **Objects** table.

@createdBy: The name or identifier of the user who created the object, which MUST be inserted into the **CreatedBy** column of the **Objects** table.

@filePath: The relative location of the object in the database application, which MUST be inserted into the **FilePath** column of the **ObjectStorage** table. MUST NOT be NULL.

@contentType: The MIME type of the object, which MUST be inserted into the **ContentType** column of the **ObjectStorage** table. MUST NOT be NULL.

@contents: Binary data associated with the object, which MUST be inserted into the **Contents** column of the **ObjectStorage** table. MUST NOT be NULL.

@supportingObjects: An **ObjectNameList** (section 2.2.1.2) that specifies the dependent objects of the object being inserted. Each object in this list MUST be inserted into the **ObjectDependencies** table (section 2.2.5.4) unless doing so would create a circular dependency. If a circular dependency would be created, then objects MUST NOT be inserted and a circular dependency error MUST be raised. A circular dependency would be created for a dependent object if *@id* matches the value of the **ParentId** column in the row (1) in the **ObjectDependencies** table that represents the dependent object.

@id: The **ID** of the newly inserted object in the **Objects** table, which MUST be returned as an output parameter.

Return Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.2.5.20 ObjectStorageUpdate

The **ObjectStorageUpdate** stored procedure updates the row (1) in the **Objects** table (section 2.2.5.5) in which the value of **ID** matches *@id*. If there is no matching row (1), or if the value of **LastModified** in that row (1) does not match *@lastModified*, then an error MUST be raised to the caller as specified by **RaiseError** (section 3.2.5.28), where the *@errorNumber* is 50002 and *@errorMessage* is text specifying a save conflict. The procedure also updates the row (1) in the **ObjectStorage** table (section 2.2.5.7) in which the value of the **ObjectID** column matches *@id*.

For objects with an *@objectTypeNumber* of "100", the new value of the **LastModified** column MUST be the last modified date of the corresponding object in the database. For all other objects, the new value of the **LastModified** column MUST be the current time when the procedure is called.

For all objects, the new value of the **LastModifiedInSql** column in the **ObjectStorage** table is *@lastModified*.

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

```
PROCEDURE ObjectStorageUpdate (
@id int
,@objectName nvarchar(128)
,@lastModified datetime2
,@definition nvarchar(max)
,@description nvarchar(350)
,@modifiedBy nvarchar(255)
,@supportingObjects ObjectNameList
,@filePath nvarchar(450)
,@contents varbinary(max)
);
```

@id: The identifier of the object to modify in the **Objects** table.

@objectName: The name of the object, which MUST be set into the **ObjectName** column (1) of the **Objects** table. MUST NOT be NULL.

@lastModified: The timestamp for the last time the object was modified, which MUST be set into the **LastModified** column (1).

@definition: The XML definition of the object, which MUST be set into the **Definition** column (1) of the **Objects** table.

@description: A description of the object, which MUST be set into the **Description** column (1) of the **Objects** table.

@modifiedBy: The name or identifier of the user who modified the object, which MUST be set into the **ModifiedBy** column (1).

@supportingObjects: An **ObjectNameList** (section 2.2.1.2) that specifies the dependent objects of the object being inserted. This list of values MUST replace all rows (1) in the **ObjectDependencies** table (section 2.2.5.4) in which the value of **ObjectID** is the same as *@id*. Each object in this list MUST be inserted into the **ObjectDependencies** table unless doing so would create a circular dependency. If a circular dependency would be created, then objects MUST NOT be inserted and an error MUST be raised to the caller as specified by **RaiseError**, where the *@errorNumber* is 50002 and *@errorMessage* is text specifying a circular dependency. A circular dependency would be created for a dependent object if *@id* matches the value of the **ParentId** column in the row (1) in the **ObjectDependencies** table that represents the dependent object.

@filePath: The relative location of the object in the database application, which MUST be inserted into the **FilePath** column (1) of the **ObjectStorage** table. MUST NOT be NULL.

@contents: Binary data associated with the object, which MUST be inserted into the **Contents** column(1) of the **ObjectStorage** table. MUST NOT be NULL.

Return Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.2.5.21 ObjectsUpdate

The **ObjectsUpdate** stored procedure updates the row (1) in the **Objects** table (section 2.2.5.5) in which the value of **ID** matches *@id*. If there is no matching row (1), or if the value of **LastModified** in that row (1) does not match *@lastModified*, then an error MUST be raised to the caller as specified by **RaiseError** (section 3.2.5.28), where the *@errorNumber* is 50002 and *@errorMessage* is text specifying a save conflict.

For objects with an *@objectTypeNumber* of "100", the new value of the **LastModified** column MUST be the last modified date of the corresponding object in the database. For all other objects, the new value of the **LastModified** column MUST be the current time when the procedure is called.

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

```
PROCEDURE ObjectsUpdate (
@id int
,@objectName nvarchar(128)
,@lastModified datetime2
,@definition nvarchar(max)
,@description nvarchar(350)
,@modifiedBy nvarchar(255)
,@orderBy nvarchar(max) = null
,@supportingObjects ObjectNameList
);
```

@id: The identifier of the object to modify in the **Objects** table.

@objectName: The name of the object, which MUST be set into the **ObjectName** column (1). MUST NOT be NULL.

@lastModified: The timestamp for the last time the object was modified, which MUST be set into the **LastModified** column (1).

@definition: The XML definition of the object, which MUST be set into the **Definition** column (1).

@description: A description of the object, which MUST be set into the **Description** column (1).

@modifiedBy: The name or identifier of the user who modified the object, which MUST be set into the **ModifiedBy** column (1).

@orderBy: An XML representation of the sorting of the object, which MUST be set into the **OrderBy** column (1).

@supportingObjects: An **ObjectNameList** (section 2.2.1.2) that specifies the dependent objects of the object being inserted. This list of values MUST replace all rows (1) in the **ObjectDependencies** table (section 2.2.5.4) in which the value of **ObjectID** is the same as *@id*. Each object in this list MUST be inserted into the **ObjectDependencies** table unless doing so would create a circular dependency. If a circular dependency would be created, then objects MUST NOT be inserted and an error MUST be raised to the caller as specified by **RaiseError**, where the *@errorNumber* is 50002 and *@errorMessage* is text specifying a circular dependency. A circular dependency would be created for a dependent object if *@id* matches the value of the **ParentId** column in the row (1) in the **ObjectDependencies** table that represents the dependent object.

Return Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.2.5.22 ObjectsUpdateProperties

The **ObjectsUpdateProperties** stored procedure updates the properties associated with an object in the **Objects** table (section 2.2.5.5).

```
PROCEDURE ObjectsUpdateProperties (
@id int
,@properties nvarchar(max)
);
```

@id: An identifier for an object in the **Objects** table.

@properties: The **Properties** to update in the **Objects** table. Properties are overwritten with the new value.

Return Values: An integer that MUST be 0.

Result Sets: MUST NOT return any result sets.

3.2.5.23 PopulateTimeZoneData

The **PopulateTimeZoneData** stored procedure inserts time zone definitions and rules into the **TimeZoneDefinitionBase** table (section 2.2.5.9) and **TimeZoneRuleBase** table (section 2.2.5.10).

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

```
PROCEDURE PopulateTimeZoneData (
@Definitions TimeZoneDefinitions
,@Rules TimeZoneRules
);
```

@Definitions: A **TimeZoneDefinitions** (section 2.2.1.5). All rows (1) in *@Definitions* MUST be inserted into the **TimeZoneDefinitionBase** table.

@Rules: A **TimeZoneRules** (section 2.2.1.6). All rows (1) in *@Rules* MUST be inserted into the **TimeZoneRuleBase** table.

Return Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.2.5.24 QueryColumnsDelete

The **QueryColumnsDelete** stored procedure deletes a row (1) from the **QueryColumns** table (section 2.2.5.8).

```
PROCEDURE QueryColumnsDelete (
@objectName nvarchar(128)
);
```

@objectName: The name of a query. If *@objectName* is not NULL, then all rows (1) in the **QueryColumns** table in which the value of the **QueryName** column is the same as *@objectName*

MUST be deleted. If *@objectName* is NULL or if no rows (1) contain a matching value in the **QueryName** column, the table MUST NOT be updated.

Return Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.2.5.25 QueryColumnsInsert

The **QueryColumnsInsert** stored procedure inserts rows (1) provided as a table parameter into the **QueryColumns** table (section 2.2.5.8).

```
PROCEDURE QueryColumnsInsert (
@QueryColumnsData QueryColumnsTable
);
```

@QueryColumnsData: A **QueryColumnsTable** (section 2.2.1.4) that specifies information about query columns (1). Each row (1) in *@QueryColumnsData* MUST be inserted into the **QueryColumns** table.

Return Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.2.5.26 QueryObjectInsert

The **QueryObjectInsert** stored procedure inserts an object into the **Objects** table (section 2.2.5.5) and its dependencies into the **ObjectDependencies** table (section 2.2.5.4). The identifier of the newly inserted object is returned in the *@id* output parameter.

For objects with an *@objectTypeNumber* of "100", the initial value of the **LastModified** column MUST be the last modified date of the object in the database. For all other objects, the initial value of the **LastModified** column MUST be the current time when the procedure is called.

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

```
PROCEDURE QueryObjectInsert (
@objectName nvarchar(128)
,@objectTypeNumber int
,@definition nvarchar(max)
,@description nvarchar(350)
,@parentId int
,@createdBy nvarchar(255)
,@orderBy nvarchar(max) = null
,@supportingObjects ObjectNameList
,@id int OUTPUT
);
```

@objectName: The name of the object, which MUST be inserted into the **ObjectName** column (1). MUST NOT be NULL.

@objectTypeNumber: An **ObjectTypeNumber** (section 2.2.1.3) that specifies the type of the object, which MUST be inserted into the **ObjectTypeNumber** column (1).

@definition: The XML definition of the object, which MUST be inserted into the **Definition** column (1).

@description: A description of the object, which MUST be inserted into the **Description** column (1).

@parentId: The identifier of the parent object, which MUST be inserted into the **ParentId** column (1).

@createdBy: The name or identifier of the user who created the object, which MUST be inserted into the **CreatedBy** column (1).

@orderBy: An XML representation of the sorting of the object, which MUST be inserted into the **OrderBy** column (1).

@supportingObjects: An **ObjectNameList** (section 2.2.1.2) that specifies the dependent objects of the object being inserted. Each object in this list MUST be inserted into the **ObjectDependencies** table (section 2.2.5.4) unless doing so would create a circular dependency. If a circular dependency would be created, then objects MUST NOT be inserted and an error MUST be raised to the caller as specified by **RaiseError** (section 3.2.5.28), where the *@errorNumber* is 50001 and *@errorMessage* is text specifying a circular dependency. A circular dependency would be created for a dependent object if *@id* matches the value of the **ParentId** column in the row (1) in the **ObjectDependencies** table that represents the dependent object.

@id: The **ID** of the newly inserted object in the **Objects** table, which MUST be returned as an output parameter.

Return Values: An integer which MUST be 0.

Result Sets:

This stored procedure MUST return a QueryObject.ResultSet0

This stored procedure MUST return a QueryObject.ResultSet1

3.2.5.27 QueryObjectUpdate

The **QueryObjectUpdate** stored procedure updates the row (1) in the **Objects** table (section 2.2.5.5) in which the value of **ID** matches *@id*. If there is no matching row (1), or if the value of **LastModified** in that row (1) does not match *@lastModified*, then an error MUST be raised to the caller as specified by **RaiseError** (section 3.2.5.28), where the *@errorNumber* is 50002 and *@errorMessage* is text specifying a save conflict.

For objects with an *@objectTypeNumber* of "100", the new value of the **LastModified** column MUST be the last modified date of the corresponding object in the database. For all other objects, the new value of the **LastModified** column MUST be the current time when the procedure is called.

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

```
PROCEDURE QueryObjectUpdate (
@id int
,@objectName nvarchar(128)
,@lastModified datetime2
,@definition nvarchar(max)
,@description nvarchar(350)
,@modifiedBy nvarchar(255)
,@supportingObjects ObjectNameList
```

```
,@orderBy nvarchar(max) = null
);
```

@id: The identifier of the object to modify in the **Objects** table.

@objectName: The name of the object, which MUST be set into the **ObjectName** column (1). MUST NOT be NULL.

@lastModified: The timestamp for the last time the object was modified.

@definition: The XML definition of the object, which MUST be set into the **Definition** column (1).

@description: A description of the object, which MUST be set into the **Description** column (1).

@modifiedBy: The name or identifier of the user who modified the object, which MUST be set into the **ModifiedBy** column (1).

@supportingObjects: An **ObjectNameList** (section 2.2.1.2) that specifies the dependent objects of the object being inserted. Each object in this list MUST be inserted into the **ObjectDependencies** table (section 2.2.5.4) unless doing so would create a circular dependency. If a circular dependency would be created, then objects MUST NOT be inserted and an error MUST be raised to the caller as specified by **RaiseError** (section 3.2.5.28), where the *@errorNumber* is 50001 and *@errorMessage* is text specifying a circular dependency. A circular dependency would be created for a dependent object if *@id* matches the value of the **ParentId** column in the row (1) in the **ObjectDependencies** table that represents the dependent object.

@orderBy: An XML representation of the sorting of the object, which MUST be set into the **OrderBy** column (1).

Return Values: An integer which MUST be 0.

Result Sets:

This stored procedure MUST return a QueryObject.ResultSet0

This stored procedure MUST return a QueryObject.ResultSet1

3.2.5.28 RaiseError

The **RaiseError** stored procedure raises an error.

The severity of the error MUST be 16 and the state of the error MUST be 255. The error message MUST be the string '<?xml version="1.0" encoding="utf-8"?><Error><Number>%d</Number><Description>%s</Description></Error>', where "%d" is replaced with *@errorNumber* and "%s" is replaced with *@errorMessage*.

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

```
PROCEDURE RaiseError (
@errorNumber int
,@errorMessage nvarchar(4000)
);
```

@errorNumber: The error number to raise.

@errorMessage: The message to include in the raised error.

Return Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.2.5.29 DatabaseCollationInfoSelect

The **DatabaseCollationInfoSelect** stored procedure returns the comparison style and locale identifier of the database.

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

```
PROCEDURE DatabaseCollationInfoSelect (
@ComparisonStyle int OUTPUT
,@LCID int OUTPUT
);
```

@ComparisonStyle: The comparison style of the database, returned as an output parameter.

Value	Description
convert	The comparison style of the database. MUST be a value specified by the **DATABASEPROPERTYEX** Metadata Function ([MSDN-TSQL-Ref]) with the **property** parameter of "ComparisonStyle".

@LCID: The locale identifier of the database, returned as an output parameter.

Value	Description
convert	The locale identifier of the database. MUST be a value specified by the **DATABASEPROPERTYEX** Metadata Function ([MSDN-TSQL-Ref]) with the **property** parameter of "LCID".

Return Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.2.5.30 ObjectsUpdateOrderBy

The **ObjectsUpdateOrderBy** stored procedure updates the value of **OrderBy** in the **Objects** table (section 2.2.5.5).

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

```
PROCEDURE ObjectsUpdateOrderBy (
@id int
,@orderBy nvarchar(max)
);
```

@id: Specifies the identifier of an object in the **Objects** table. If *@id* is NULL or does not match the value of **ID** in any row (1) of the table, then the table MUST NOT be updated. Otherwise, the value of the **OrderBy** column in the row (1) in which *@id* matches **ID** MUST be set to *@orderBy*.

@orderBy: An XML representation of the sorting of the object.

Return Values: An integer which MUST be 0.

Result Sets: MUST NOT return any result sets.

3.2.5.31 QueryObjectUpdateOrderBy

The **QueryObjectUpdateOrderBy stored procedure** updates the value of **OrderBy** in the **Objects** table (section 2.2.5.5).

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

```
PROCEDURE QueryObjectUpdateOrderBy (
@id int
,@orderBy nvarchar(max) = null
);
```

@id: Specifies the identifier of an object in the **Objects** table. If *@id* is NULL or does not match the value of **ID** in any row (1) of the table, then the table MUST NOT be updated. Otherwise, the value of the **OrderBy** column in the row (1) in which *@id* matches **ID** MUST be set to *@orderBy*.

@orderBy: An XML representation of the sorting of the object.

Return Values: An integer which MUST be 0.

Result Sets:

This stored procedure MUST return a QueryObjectUpdateOrderBy.ResultSet0

This stored procedure MUST return a QueryObjectUpdateOrderBy.ResultSet1

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

3.3 Client Details

3.3.1 Abstract Data Model

None.

3.3.2 Timers

None.

3.3.3 Initialization

None.

3.3.4 Higher-Layer Triggered Events

None.

3.3.5 Message Processing Events and Sequencing Rules

None.

3.3.6 Timer Events

None.

3.3.7 Other Local Events

None.

4 Protocol Examples

The following examples show the basic lifecycle of a table in the database.

4.1 Create Table

In order to create a table in the database, the table is created using SQL commands to define the table, using the CREATE TABLE statement along with calls to the Access stored procedures that define the XML for the table and store information on column properties for the new table.

The table created in this action is named "Contacts" and contains three fields:

```
FirstName: nvarchar(220) NULL
LastName: nvarchar(220) NULL
Email: nvarchar(220) NULL
```

The protocol starts after the table has already been created in the database. The object definition is inserted into the **Objects** table (section 2.2.5.5) through a call to **ObjectsInsert** (section 3.2.5.18). Consider the following SQL request.

```
declare @p7 Access.ObjectNameList
declare @p8 int
set @p8=10

exec [Access].[ObjectsInsert]
@objectName=N'Contacts',
@objectTypeNumber=100,
@createdBy=N'User Name',
@description=N'',
@parentId=NULL,
@definition=N'<Schema Namespace="Access.Store"
xmlns:axl="http://schemas.microsoft.com/office/accessservices/2010/12/application"
xmlns="http://schemas.microsoft.com/ado/2008/09/edm"><EntityContainer
Name="AccessStoreContainer"><EntitySet Name="Contacts" EntityType="Access.Store.Contacts"
/></EntityContainer><EntityType Name="Contacts"><Key><PropertyRef Name="ID" /></Key><Property
Name="ID" Type="Int32" axl:ObjectId="ID" Nullable="false"
axl:StoreGeneratedPattern="Identity" /><Property Name="FirstName" Type="String"
axl:ObjectId="FirstName" Unicode="true" axl:TextType="SingleLine" MaxLength="220" /><Property
Name="LastName" Type="String" axl:ObjectId="LastName" Unicode="true"
axl:TextType="SingleLine" MaxLength="220" /><Property Name="Email" Type="String"
axl:ObjectId="Email" Unicode="true" axl:TextType="SingleLine" MaxLength="220"
/><axl:EventDataMacro><axl:DataMacro Event="AfterInsert"
/></axl:EventDataMacro><axl:EventDataMacro><axl:DataMacro Event="AfterUpdate"
/></axl:EventDataMacro><axl:EventDataMacro><axl:DataMacro Event="AfterDelete"
/></axl:EventDataMacro></EntityType></Schema>',
@supportingObjects=@p7,
@id=@p8 output
```

The protocol returns with result code "0" and the new IDENTITY value for the object is output in the *@id* parameter

Once the object definition is stored in the **Objects** table, the column properties for each column in the table are defined. Column properties are inserted into the **ColumnProperties** table (section 2.2.5.2) by calling **ColumnPropertiesInsert** (section 3.2.5.7). The *@id* parameter for **ColumnPropertiesInsert** takes the value returned in the *@id OUTPUT* parameter of **ObjectsInsert**.

```
EXEC [Access].[ColumnPropertiesInsert]
@objectId=10,
@objectName=N'dbo.Contacts',
@columnName=N'FirstName',
@properties=N'<axl:ColumnProperties axl:TextType="SingleLine"
xmlns:axl="http://schemas.microsoft.com/office/accessservices/2010/12/application" />',
@extendedAttributes=NULL

EXEC [Access].[ColumnPropertiesInsert]
@objectId=10, @objectName=N'dbo.Contacts',
@columnName=N'LastName',
@properties=N'<axl:ColumnProperties axl:TextType="SingleLine"
xmlns:axl="http://schemas.microsoft.com/office/accessservices/2010/12/application" />',
@extendedAttributes=NULL

EXEC [Access].[ColumnPropertiesInsert]
@objectId=10,
@objectName=N'dbo.Contacts',
@columnName=N'Email',
@properties=N'<axl:ColumnProperties axl:TextType="SingleLine"
xmlns:axl="http://schemas.microsoft.com/office/accessservices/2010/12/application" />',
@extendedAttributes=NULL
```

The protocol returns with result code "0" for each call to **ColumnPropertiesInsert**.

4.2 Update Table

In this example, a new column is added to the table created in section 4.1. The new field has the following characteristics:

```
Phone: nvarchar(220) NULL
```

In order to update the table definition, the client first submits changes to the table schema using the SQL ALTER TABLE statement.

The current definition of the object can then be retrieved by calling **GetObjects** (section 3.2.5.10), passing in the ID of the object in the **Objects** table (section 2.2.5.5).

```
exec [Access].[GetObjects] @objectId=10
```

The values in the result set for this call are stored for use in subsequent steps.

The new column information is added to the **ColumnProperties** table (section 2.2.5.2) by calling **ColumnPropertiesInsert** (section 3.2.5.7) for the new column only, as shown in the following snippet.

```
EXEC [Access].[ColumnPropertiesInsert]
@objectId=10,
@objectName=N'dbo.Contacts',
@columnName=N'Phone',
@properties=N'<axl:ColumnProperties axl:TextType="SingleLine"
xmlns:axl="http://schemas.microsoft.com/office/accessservices/2010/12/application" />',
@extendedAttributes=NULL
```

The client then modifies the definition of the table CSDL by calling **ObjectsUpdate** (section 3.2.5.21). The value of the *@lastModified* parameter is the value returned in the **LastModified** column of the **GetObjects** call (section 3.2.5.10). This value is used as a concurrency check to ensure that the object has not been modified by another user between modification requests.

```
declare @p7 Access.ObjectNameList
exec [Access].[ObjectsUpdate]
@id=10,
@objectName=N'Contacts',
@lastModified='2011-11-11 09:27:01.1330000',
@modifiedBy=N'User Name',
@description=N'',
@definition=N'<Schema Namespace="Access.Store"
xmlns:axl="http://schemas.microsoft.com/office/accessservices/2010/12/application"
xmlns="http://schemas.microsoft.com/ado/2008/09/edm"><EntityContainer
Name="AccessStoreContainer"><EntitySet Name="Contacts" EntityType="Access.Store.Contacts"
/></EntityContainer><EntityType Name="Contacts"><Key><PropertyRef Name="ID" /></Key><Property
Name="ID" Type="Int32" axl:ObjectId="ID" Nullable="false"
axl:StoreGeneratedPattern="Identity" /><Property Name="FirstName" Type="String"
axl:ObjectId="FirstName" Unicode="true" axl:TextType="SingleLine" MaxLength="220" /><Property
Name="LastName" Type="String" axl:ObjectId="LastName" Unicode="true"
axl:TextType="SingleLine" MaxLength="220" /><Property Name="Email" Type="String"
axl:ObjectId="Email" Unicode="true" axl:TextType="SingleLine" MaxLength="220" /><Property
Name="Phone" Type="String" axl:ObjectId="Phone" Unicode="true" axl:TextType="SingleLine"
MaxLength="220" /><axl:EventDataMacro><axl:DataMacro Event="AfterInsert"
/></axl:EventDataMacro><axl:EventDataMacro><axl:DataMacro Event="AfterUpdate"
/></axl:EventDataMacro><axl:EventDataMacro><axl:DataMacro Event="AfterDelete"
/></axl:EventDataMacro></EntityType></Schema>',
@supportingObjects=@p7
```

The protocol returns with result code "0" and the table definition is updated.

4.3 Delete Table

In this example, the table defined in section 4.1 and section 4.2 is removed from the database.

Before deleting the table in the database, the current definition of the object must be retrieved by calling **GetObjects** (section 3.2.5.10), passing in the **ID** of the object from the **Objects** table (section 2.2.5.5).

```
exec [Access].[GetObjects] @objectId=10
```

The values in the result set for this call are stored for use in subsequent steps.

The client is now able to delete the object from the database using the SQL DROP TABLE command.

Finally, the object definition is deleted from the database by calling **ObjectsDelete** (section 3.2.5.16), passing in the ID of the object as the *@id* parameter, and the **LastModified** value to the *@lastModified* parameter.

```
exec [Access].[ObjectsDelete] @id=10,@lastModified='2011-11-11 09:27:34.6500000'
```

5 Security

5.1 Security Considerations for Implementers

Interactions with SQL are susceptible to tampering and other forms of security risks. Implementers are advised to sanitize input parameters for a stored procedure prior to calling the stored procedure.

5.2 Index of Security Parameters

Security Parameter	Section
Authentication protocol	1.7

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® Server 2013 Preview

Exceptions, if any, are noted below. If a service pack or Quick Fix Engineering (QFE) number appears with the product version, behavior changed in that service pack or QFE. The new behavior also applies to subsequent service packs of the product unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

Unless otherwise specified, any statement of optional behavior in this specification that is prescribed using the terms SHOULD or SHOULD NOT implies product behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term MAY implies that the product does not follow the prescription.

7 Change Tracking

No table of changes is available. The document is either new or has had no changes since its last release.

8 Index

A

Abstract data model
  client 60
  server 39
Applicability 8
ApplicationPropertiesDelete method 39
ApplicationPropertiesInsert method 40
ApplicationPropertiesUpdate method 40
ApplicationPropertiesUpdateIf method 41
Attribute groups - overview 38
Attributes - overview 38

B

Binary structures - overview 13
Bit fields - overview 13

C

Capability negotiation 8
Change tracking 67
Client
  abstract data model 60
  higher-layer triggered events 61
  initialization 60
  local events 61
  message processing 61
  sequencing rules 61
  timer events 61
  timers 60
ColumnPropertiesColumnRename method 41
ColumnPropertiesDelete method 42
ColumnPropertiesInsert method 42
ColumnPropertiesUpdate method 43
Complex types - overview 38

D

Data model - abstract
  client 60
  server 39
Data types
  ObjectIdentityTable simple type 9
  ObjectNameList simple type 9
  ObjectTypeNumber simple type 9
  QueryColumnsTable simple type 10
  TimeZoneDefinitions simple type 10
  TimeZoneRules simple type 11
Data types - simple
  ObjectIdentityTable 9
  ObjectNameList 9
  ObjectTypeNumber 9
  QueryColumnsTable 10
  TimeZoneDefinitions 10
  TimeZoneRules 11
DatabaseCollationInfoSelect method 59

E

Elements - overview 38
Events
  local - client 61
  local - server 60
  timer - client 61
  timer - server 60

F

Fields - vendor-extensible 8
Flag structures - overview 13

G

GetExternalLinksAndObjectSchema method 44
GetObjects method 45
GetObjectSchema method 45
GetUserTableSchema method 46
Glossary 6
Groups - overview 38

H

HandleError method 47
Higher-layer triggered events
  client 61
  server 39

I

Implementer - security considerations 65
Index of security parameters 65
Informative references 7
Initialization
  client 60
  server 39
Introduction 6

L

Local events
  client 61
  server 60
LogActionTrace method 48

M

Message processing
  client 61
  server 39
Messages
  attribute groups 38
  attributes 38
  binary structures 13
  bit fields 13
  complex types 38
  elements 38
  flag structures 13
  groups 38

namespaces 38
result sets (section 2.2.4 14, section 2.2.4 14)
simple types 38
table structures (section 2.2.5 24, section 2.2.5 24)
transport 9
view structures (section 2.2.5 24, section 2.2.5 24)
XML structures 37
Methods
ApplicationPropertiesDelete 39
ApplicationPropertiesInsert 40
ApplicationPropertiesUpdate 40
ApplicationPropertiesUpdateIf 41
ColumnPropertiesColumnRename 41
ColumnPropertiesDelete 42
ColumnPropertiesInsert 42
ColumnPropertiesUpdate 43
DatabaseCollationInfoSelect 59
GetExternalLinksAndObjectSchema 44
GetObjects 45
GetObjectSchema 45
GetUserTableSchema 46
HandleError 47
LogActionTrace 48
ObjectDefinitionSelect 49
ObjectsDelete 49
ObjectsDeleteByObjectName 50
ObjectsInsert 50
ObjectStorageInsert 51
ObjectStorageUpdate 52
ObjectsUpdate 54
ObjectsUpdateOrderBy 59
ObjectsUpdateProperties 55
PopulateTimeZoneData 55
QueryColumnsDelete 55
QueryColumnsInsert 56
QueryObjectInsert 56
QueryObjectUpdate 57
QueryObjectUpdateOrderBy 60
RaiseError 58

N

Namespaces 38
Normative references 7

O

ObjectDefinitionSelect method 49
ObjectIdentityTable simple type 9
ObjectNameList simple type 9
ObjectsDelete method 49
ObjectsDeleteByObjectName method 50
ObjectsInsert method 50
ObjectStorageInsert method 51
ObjectStorageUpdate method 52
ObjectsUpdate method 54
ObjectsUpdateOrderBy method 59
ObjectsUpdateProperties method 55
ObjectTypeNumber simple type 9
Overview (synopsis) 7

P

Parameters - security index 65
PopulateTimeZoneData method 55
Preconditions 8
Prerequisites 8
Product behavior 66

Q

QueryColumnsDelete method 55
QueryColumnsInsert method 56
QueryColumnsTable simple type 10
QueryObjectInsert method 56
QueryObjectUpdate method 57
QueryObjectUpdateOrderBy method 60

R

RaiseError method 58
References 6
informative 7
normative 7
Relationship to other protocols 7
Result sets
overview 14
Result sets - overview 14

S

Security
implementer considerations 65
parameter index 65
Sequencing rules
client 61
server 39
Server
abstract data model 39
ApplicationPropertiesDelete method 39
ApplicationPropertiesInsert method 40
ApplicationPropertiesUpdate method 40
ApplicationPropertiesUpdateIf method 41
ColumnPropertiesColumnRename method 41
ColumnPropertiesDelete method 42
ColumnPropertiesInsert method 42
ColumnPropertiesUpdate method 43
DatabaseCollationInfoSelect method 59
GetExternalLinksAndObjectSchema method 44
GetObjects method 45
GetObjectSchema method 45
GetUserTableSchema method 46
HandleError method 47
higher-layer triggered events 39
initialization 39
local events 60
LogActionTrace method 48
message processing 39
ObjectDefinitionSelect method 49
ObjectsDelete method 49
ObjectsDeleteByObjectName method 50
ObjectsInsert method 50
ObjectStorageInsert method 51

ObjectStorageUpdate method 52
ObjectsUpdate method 54
ObjectsUpdateOrderBy method 59
ObjectsUpdateProperties method 55
PopulateTimeZoneData method 55
QueryColumnsDelete method 55
QueryColumnsInsert method 56
QueryObjectInsert method 56
QueryObjectUpdate method 57
QueryObjectUpdateOrderBy method 60
RaiseError method 58
sequencing rules 39
timer events 60
timers 39
Simple data types
ObjectIdentityTable 9
ObjectNameList 9
ObjectTypeNumber 9
QueryColumnsTable 10
TimeZoneDefinitions 10
TimeZoneRules 11
Simple types - overview 38
Standards assignments 8
Structures
binary 13
table and view (section 2.2.5 24, section 2.2.5 24)
XML 37

T

Table structures - overview 24
Timer events
client 61
server 60
Timers
client 60
server 39
TimeZoneDefinitions simple type 10
TimeZoneRules simple type 11
Tracking changes 67
Transport 9
Triggered events - higher-layer
client 61
server 39
Types
complex 38
simple 38

V

Vendor-extensible fields 8
Versioning 8
View structures
overview 24
View structures - overview 24

X

XML structures 37