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**

<table>
<thead>
<tr>
<th>Date</th>
<th>Revision History</th>
<th>Revision Class</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>04/04/2008</td>
<td>0.1</td>
<td></td>
<td>Initial Availability</td>
</tr>
<tr>
<td>06/27/2008</td>
<td>1.0</td>
<td>Major</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>12/12/2008</td>
<td>1.01</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>03/18/2009</td>
<td>1.02</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>07/13/2009</td>
<td>1.03</td>
<td>Major</td>
<td>Changes made for template compliance</td>
</tr>
<tr>
<td>08/28/2009</td>
<td>1.04</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>11/06/2009</td>
<td>1.05</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>02/19/2010</td>
<td>2.0</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>03/31/2010</td>
<td>2.01</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>04/30/2010</td>
<td>2.02</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>06/07/2010</td>
<td>2.03</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>06/29/2010</td>
<td>2.04</td>
<td>Editorial</td>
<td>Changed language and formatting in the technical content.</td>
</tr>
<tr>
<td>07/23/2010</td>
<td>2.04</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>09/27/2010</td>
<td>2.04</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>11/15/2010</td>
<td>2.04</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>12/17/2010</td>
<td>2.05</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content.</td>
</tr>
<tr>
<td>03/18/2011</td>
<td>2.05</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>06/10/2011</td>
<td>2.05</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>01/20/2012</td>
<td>3.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>Date</td>
<td>Revision History</td>
<td>Revision Class</td>
<td>Comments</td>
</tr>
<tr>
<td>------------</td>
<td>------------------</td>
<td>----------------</td>
<td>------------------------------------------------------------</td>
</tr>
<tr>
<td>04/11/2012</td>
<td>3.0</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>07/16/2012</td>
<td>3.0</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
</tbody>
</table>
# Table of Contents

## 1 Introduction

1.1 Glossary ............................................................................................................... 9  
1.2 References ........................................................................................................... 10  
  1.2.1 Normative References .................................................................................. 10  
  1.2.2 Informative References .............................................................................. 10  
1.3 Protocol Overview (Synopsis) ............................................................................. 11  
1.4 Relationship to Other Protocols ......................................................................... 11  
1.5 Prerequisites/Preconditions ................................................................................ 12  
1.6 Applicability Statement ...................................................................................... 12  
1.7 Versioning and Capability Negotiation .............................................................. 12  
1.8 Vendor-Extensible Fields .................................................................................... 12  
1.9 Standards Assignments ...................................................................................... 12  

## 2 Messages

2.1 Transport ............................................................................................................ 13  
2.2 Common Data Types .......................................................................................... 13  
  2.2.1 Simple Data Types and Enumerations ......................................................... 13  
  2.2.2 Common Fields ............................................................................................ 13  
    2.2.2.1 Id ........................................................................................................... 13  
    2.2.2.2 Name ..................................................................................................... 13  
    2.2.2.3 IsCached ................................................................................................ 13  
    2.2.2.4 EstimatedInstanceCount ...................................................................... 13  
    2.2.2.5 MetadataObjectType ............................................................................. 13  
    2.2.2.6 Position ................................................................................................ 15  
    2.2.2.7 IsDisplayed ............................................................................................ 15  
    2.2.2.8 IsCollection .......................................................................................... 15  
    2.2.2.9 IsOpenedInNewWindow ........................................................................ 15  
    2.2.2.10 Icon ..................................................................................................... 15  
    2.2.2.11 URL ..................................................................................................... 15  
    2.2.2.12 Index ................................................................................................... 15  
    2.2.2.13 FilterDescriptorTypeName .................................................................. 15  
    2.2.2.14 IdentifierTypeName ............................................................................ 16  
    2.2.2.15 MethodInstanceType .......................................................................... 17  
    2.2.2.16 Direction ............................................................................................. 18  
    2.2.2.17 TypeReflectorTypeName ...................................................................... 18  
    2.2.2.18 TypeDescriptorTypeName ................................................................... 18  
    2.2.2.19 ConnectionManagerTypeName ............................................................ 18  
    2.2.2.20 SystemUtilityTypeName ..................................................................... 19  
    2.2.2.21 EntityInstanceTypeName .................................................................... 19  
    2.2.2.22 MetadataRights .................................................................................. 20  
    2.2.2.23 IsStatic ............................................................................................... 20  
    2.2.2.24 IsCollection ....................................................................................... 20  
    2.2.2.25 IsDisplayed ......................................................................................... 20  
    2.2.2.26 Position ............................................................................................... 20  
    2.2.2.27 MetadataObjectType .......................................................................... 20  
  2.2.3 Bit Fields and Flag Structures ....................................................................... 20  
  2.2.4 Binary Structures .......................................................................................... 20  
  2.2.5 Result Sets ..................................................................................................... 20  
    2.2.5.1 Action Result Set .................................................................................. 20  
    2.2.5.2 ActionParameter Result Set .................................................................. 21  
    2.2.5.3 Association Result Set .......................................................................... 22  
    2.2.5.4 Count Result Set .................................................................................. 22  
    2.2.5.5 DataClass Result Set ............................................................................ 23  
    2.2.5.6 Entity Result Set ................................................................................... 23  

---

[MS-BDCSP] — v20120630  
*Business Data Catalog Database Protocol Specification*  

Copyright © 2012 Microsoft Corporation.  

Release: July 16, 2012
3 Protocol Details ...................................................................................................... 30

3.1 Back End Database Server Details ..................................................................... 30
3.1.1 Abstract Data Model ....................................................................................... 30
3.1.2 Timers .......................................................................................................... 31
3.1.3 Initialization .................................................................................................. 31
3.1.4 Higher-Layer Triggered Events ...................................................................... 31
3.1.5 Message Processing Events and Sequencing Rules ........................................ 31
    3.1.5.1 proc_ar_AddOrInsertLocalizedStringForMetadataObjectId .................. 31
    3.1.5.2 proc_ar_AddOrInsertPropertyForMetadataObjectId ............................... 32
    3.1.5.3 proc_ar_BumpCacheInvalidationCounter ............................................. 33
    3.1.5.4 proc_ar_ClearAccessControlEntriesForMetadataObject ........................ 34
    3.1.5.5 proc_ar_CopyAccessControlEntriesForMetadataObjectId ................... 34
    3.1.5.6 proc_ar_CreateAction .......................................................................... 34
    3.1.5.7 proc_ar_CreateActionParameter .......................................................... 36
    3.1.5.8 proc_ar_CreateAssociation .................................................................. 37
    3.1.5.9 proc_ar_CreateEntity .......................................................................... 38
    3.1.5.10 proc_ar_CreateFilterDescriptor .......................................................... 39
    3.1.5.11 proc_ar_CreateIdentifier .................................................................... 40
    3.1.5.12 proc_ar_CreateMethod ......................................................................... 41
    3.1.5.13 proc_ar_CreateMethodInstance ............................................................ 42
    3.1.5.14 proc_ar_CreateParameter ..................................................................... 43
    3.1.5.15 proc_ar_CreateSystem ........................................................................ 44
    3.1.5.16 proc_ar_CreateSystemInstance ............................................................. 45
    3.1.5.17 proc_ar_CreateTypeDescriptor ............................................................. 46
    3.1.5.18 proc_ar_DeleteActionByld ................................................................. 47
    3.1.5.19 proc_ar_DeleteActionParameterByld .................................................... 48
    3.1.5.20 proc_ar_DeleteAssociationByld ............................................................. 49
    3.1.5.21 proc_ar_DeleteDefaultValue ............................................................... 50
    3.1.5.22 proc_ar_DeleteEntityByld ................................................................. 50
    3.1.5.23 proc_ar_DeleteFilterDescriptorByld .................................................... 51
    3.1.5.24 proc_ar_DeleteIdentifierByld .............................................................. 52
    3.1.5.25 proc_ar_DeleteLocalizedNameForMetadataObjectByLCID ............... 53
    3.1.5.26 proc_ar_DeleteLocalizedNamesByMetadataObjectId .......................... 53
    3.1.5.27 proc_ar_DeleteMethodByld ................................................................. 54
    3.1.5.28 proc_ar_DeleteMethodInstanceByld ..................................................... 55
    3.1.5.29 proc_ar_DeleteParameterByld .............................................................. 55
    3.1.5.30 proc_ar_DeletePropertiesByld ............................................................. 56
    3.1.5.31 proc_ar_DeletePropertyForMetadataObjectId ..................................... 57
    3.1.5.32 proc_ar_DeleteSystemByld ................................................................. 57
    3.1.5.33 proc_ar_DeleteSystemInstanceByld ..................................................... 58
    3.1.5.34 proc_ar_DeleteTypeDescriptorByld ..................................................... 59
    3.1.5.35 proc_ar_EsureApplicationRegistryExists ........................................... 59

2.2.6 Tables and Views ........................................................................................... 29
2.2.7 XML Structures .............................................................................................. 29
3.1.5.36 proc_ar_GetAccessControlEntriesForMetadataObject ............................................. 60
3.1.5.36.1 Access Control Entry Result Set ..................................................................... 60
3.1.5.37 proc_ar_GetActionById ................................................................................... 61
3.1.5.37.1 Action Result Set ......................................................................................... 61
3.1.5.38 proc_ar_GetActionParameterById .................................................................. 61
3.1.5.38.1 ActionParameter Result Set ......................................................................... 61
3.1.5.39 proc_ar_GetActionParametersForActionWithCount ......................................... 62
3.1.5.39.1 Count Result Set ......................................................................................... 62
3.1.5.39.2 ActionParameter Result Set ......................................................................... 62
3.1.5.40 proc_ar_GetActionsForEntityWithCount ........................................................ 62
3.1.5.40.1 Count Result Set ......................................................................................... 62
3.1.5.40.2 Action Result Set ......................................................................................... 62
3.1.5.41 proc_ar_GetAllLocalizedNamesForMetadataObjectWithCount .......................... 63
3.1.5.41.1 Count Result Set ......................................................................................... 63
3.1.5.41.2 Localized Name Result Set ........................................................................... 63
3.1.5.42 proc_ar_GetAllSystemInstancesLikeNameWithCount ........................................ 63
3.1.5.42.1 Count Result Set ......................................................................................... 64
3.1.5.42.2 System Instance Result Set .......................................................................... 64
3.1.5.43 proc_ar_GetAllSystemInstancesWithCount .................................................... 64
3.1.5.43.1 Count Result Set ......................................................................................... 64
3.1.5.43.2 System Instance Result Set ........................................................................... 64
3.1.5.44 proc_ar_GetAllSystemsWithCount ................................................................... 65
3.1.5.44.1 Count Result Set ......................................................................................... 65
3.1.5.44.2 System Result Set ......................................................................................... 65
3.1.5.45 proc_ar_GetAssociationById .......................................................................... 65
3.1.5.45.1 Association Result Set .................................................................................. 65
3.1.5.46 proc_ar_GetAssociationByType ........................................................................ 65
3.1.5.46.1 Association Result Set .................................................................................. 65
3.1.5.47 proc_ar_GetAssociationsForDataClassWithCount ............................................ 66
3.1.5.47.1 Count Result Set ......................................................................................... 66
3.1.5.47.2 Association Result Set .................................................................................. 66
3.1.5.48 proc_ar_GetAssociationsForEntityAndRoleWithCount ...................................... 67
3.1.5.48.1 Count Result Set ......................................................................................... 67
3.1.5.48.2 Association Result Set .................................................................................. 67
3.1.5.49 proc_ar_GetAssociationsForMethodWithCount ............................................... 67
3.1.5.49.1 Count Result Set ......................................................................................... 67
3.1.5.49.2 Association Result Set .................................................................................. 67
3.1.5.50 proc_ar_GetCacheInvalidationCountersWithCount ........................................... 68
3.1.5.50.1 Count Result Set ......................................................................................... 68
3.1.5.50.2 Cache Version Stamps Result Set ................................................................... 68
3.1.5.51 proc_ar_GetChildTypeDescriptorsForTypeDescriptorWithCount ....................... 69
3.1.5.51.1 Count Result Set ......................................................................................... 69
3.1.5.51.2 TypeDescriptor Result Set .......................................................................... 69
3.1.5.52 proc_ar_GetDataClassById ............................................................................. 69
3.1.5.52.1 DataClass Result Set .................................................................................... 69
3.1.5.53 proc_ar_GetDataClassesForSystemWithCount ................................................ 70
3.1.5.53.1 Count Result Set ......................................................................................... 70
3.1.5.53.2 DataClass Result Set .................................................................................... 70
3.1.5.54 proc_ar_GetDefaultValuesForTypeDescriptor ............................................... 70
3.1.5.54.1 DefaultValues Result Set ............................................................................. 71
3.1.5.55 proc_ar_GetDependentEntitiesForEntity ......................................................... 71
3.1.5.55.1 EntityId Result Set ....................................................................................... 72
3.1.5.56 proc_ar_GetEntitiesForAssociationAndRoleWithCount .................................. 72
3.1.5.61 proc_ar_GetFilterDescriptorsForMethodWithCount .................................................. 75
3.1.5.61.1 Count Result Set ........................................................................ 75
3.1.5.61.2 FilterDescriptor Result Set ........................................................................ 75
3.1.5.62 proc_ar_GetIdentifierById ................................................................. 75
3.1.5.62.1 Identifier Result Set ........................................................................ 75
3.1.5.63 proc_ar_GetIdentifiersForEntityWithCount ................................................. 76
3.1.5.63.1 Count Result Set ........................................................................ 76
3.1.5.63.2 Identifier Result Set ........................................................................ 76
3.1.5.64 proc_ar_GetMethodById ................................................................. 76
3.1.5.64.1 Method Result Set ........................................................................ 76
3.1.5.65 proc_ar_GetMethodInstanceById ................................................................. 77
3.1.5.65.1 MethodInstance Result Set ................................................................ 77
3.1.5.66 proc_ar_GetMethodInstancesForDataClassWithCount ...................................... 77
3.1.5.66.1 Count Result Set ........................................................................ 77
3.1.5.66.2 MethodInstance Result Set ................................................................ 77
3.1.5.67 proc_ar_GetMethodInstancesForMethodWithCount ............................................ 78
3.1.5.67.1 Count Result Set ........................................................................ 78
3.1.5.67.2 MethodInstance Result Set ................................................................ 78
3.1.5.68 proc_ar_GetMethodsForDataClassWithCount .................................................. 78
3.1.5.68.1 Count Result Set ........................................................................ 78
3.1.5.68.2 Method Result Set ........................................................................ 78
3.1.5.69 proc_ar_GetParameterById ................................................................. 79
3.1.5.69.1 Parameter Result Set ........................................................................ 79
3.1.5.70 proc_ar_GetParametersForMethodWithCount .................................................. 79
3.1.5.70.1 Count Result Set ........................................................................ 79
3.1.5.70.2 Parameter Result Set ........................................................................ 79
3.1.5.71 proc_ar_GetPropertiesForMetadataObject .................................................. 80
3.1.5.71.1 Property Result Set ......................................................................... 80
3.1.5.72 proc_ar_GetRootTypeDescriptorForParameter .................................................. 80
3.1.5.72.1 TypeDescriptor Result Set .................................................................. 81
3.1.5.73 proc_ar_GetSystemById ................................................................. 81
3.1.5.73.1 System Result Set ......................................................................... 81
3.1.5.74 proc_ar_GetSystemDataBySystemName .................................................... 81
3.1.5.74.1 System Data Result Set ..................................................................... 81
3.1.5.75 proc_ar_GetSystemInstanceById ................................................................. 82
3.1.5.75.1 System Instance Result Set .................................................................. 82
3.1.5.76 proc_ar_GetSystemInstancesForSystemWithCount ........................................... 82
3.1.5.76.1 Count Result Set ........................................................................ 83
3.1.5.76.2 System Instance Result Set .................................................................. 83
3.1.5.77 proc_ar_GetSystemsLikeNameWithCount .................................................... 83
3.1.5.77.1 Count Result Set ........................................................................ 83

[MS-BDCSP] — v20120630
Business Data Catalog Database Protocol Specification

Copyright © 2012 Microsoft Corporation.
Release: July 16, 2012
1 Introduction

This document specifies the Business Data Catalog Database Protocol. This protocol provides an interface for protocol clients to store and retrieve information about interfaces of line-of-business systems (LOB systems) and annotations of these interfaces.

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]:

- access control entry (ACE)
- GUID
- language code identifier (LCID)
- security identifier (SID)
- Security Support Provider Interface (SSPI)
- Unicode

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

- AccessChecker
- Action
- ActionParameter
- Association
- back-end database server
- business logic
- Business Logic Module
- ComparisonFilter
- DataClass
- Entity
- FilterDescriptor
- Finder
- front-end Web server
- GenericInvoker
- Identifier
- IdEnumerator
- LastIdFilter
- LimitFilter
- line-of-business (LOB) system
- LobSystem
- LobSystemInstance
- locale
- metadata model
- metadata store
- MetadataObject
- MetadataObjectId
- Method
- MethodInstance
- Parameter
- PasswordCredentialFilter
- Property

---

[MS-BDCSP] — v20120630
Business Data Catalog Database Protocol Specification

Copyright © 2012 Microsoft Corporation.

Release: July 16, 2012
RangeFilter
result set
return code
ReturnTypeDescriptor
root TypeDescriptor
Scalar
security principal
SpecificFinder
SsoTicketFilter
stored procedure
Transact-Structured Query Language (T-SQL)
TypeDescriptor
TypeReflector
Uniform Resource Locator (URL)
UserContextFilter
UsernameCredentialFilter
UserProfileFilter
ViewAccessor
WildcardFilter

The following terms are specific to this document:

**Business Logic Module Reference:** A string of characters that identifies a specific Business Logic Module.

**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](http://msdn2.microsoft.com/en-us/library/E4BD6494-06AD-4aed-9823-445E921C9624), as an additional source.


---

[MS-BDCSP] — v20120630
Business Data Catalog Database Protocol Specification

Copyright © 2012 Microsoft Corporation.

Release: July 16, 2012
1.2.2 Informative References


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

1.3 Protocol Overview (Synopsis)

Enterprises have a variety of data stored in various line-of-business (LOB) system. Typically, this data is accessible only through the proprietary programming interface of these software systems. It is desirable to be able to provide access to such data via a set of normalized interfaces so that users do not have to learn system-specific or adapter-specific programming patterns for each software system.

To facilitate this, it is possible to store descriptions of the programmatic interface of the LOB systems using data structures such as Methods, Parameters, and TypeDescriptors, along with information about the LOB systems themselves (such as the server name, connection string and how to authenticate), using data structures such as LobSystem and LobSystemInstance. Methods can be considered to live within an Entity abstraction, representing a business data type, such as customer or order. The LOB system interface definitions can then be transformed into stereotypical operations against Entities such as "Read-An-Entity-Instance-By-Id", "Read-Entity-Instances", and "Check-Entity-Instance-Permissions" by annotating the actual LOB system interface descriptions, with the annotations described by data structures such as MethodInstance, Identifier, FilterDescriptor, and Association. These data structures, collectively called MetadataObjects, can be grouped into related collections called metadata models that describe a single LOB system. Once a store of metadata models is made available, it can be utilized by a runtime engine to convert a stereotypical, normalized operation requested by an application that uses the protocol client into a LOB system-specific invocation programmatically.

This protocol allows a metadata client to create, read, update, and delete MetadataObjects in a metadata store. For write operations, the protocol server provides validation and diagnostic errors such that protocol clients can maintain the set of stored MetadataObjects in a state that satisfies certain semantic constraints for metadata models. These constraints are documented in detail in section 3.

1.4 Relationship to Other Protocols

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

```
+---------------------+   Business Data Catalog Database   +
|                     |                                 |
|   This Protocol     |                                 |
|                     |                                 |
|   Industry Standard |                                 |
|                     |                                 |
|   T-SQL             |                                 |
|   TDS               |                                 |
|   TCP               |                                 |
|   IP                |                                 |
```

Figure 1: This protocol in relation to other protocols
1.5 **Prerequisites/Preconditions**

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

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

1.6 **Applicability Statement**

There are typically two types of applications that can be built using the metadata client, though an application that combines these functions in a single implementation is also feasible:

**Metadata model designers:** Their primary purpose is to create or edit a metadata model. These applications typically offer some graphical design surface and connectivity to LOB systems of known types to enable mining of the LOB system public interface definition and creation of corresponding **MetadataObjects** in the protocol server store.

**Metadata model consumers:** Their primary purpose is to read a metadata model in the protocol server store and use the information therein to convert uniform, stereotypical operations into LOB system-specific interface invocations.

This protocol does not specify how the stored **MetadataObjects** can be used to do the conversion from a stereotypical client request into a system-specific invocation; it is merely a **MetadataObject** storage and retrieval protocol.

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 **Security Support Provider Interface (SSPI)** and SQL authentication with the protocol server role described in [MS-TDS].

1.8 **Vendor-Extensible Fields**

None.

1.9 **Standards Assignments**

None.
2 Messages

2.1 Transport

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

2.2 Common Data Types

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

2.2.1 Simple Data Types and Enumerations

None.

2.2.2 Common Fields

2.2.2.1 Id

Id: int NOT NULL. Identifies a MetadataObject uniquely within a metadata store.

2.2.2.2 Name

Name: nvarchar(255) NOT NULL. The programmatic name of a MetadataObject.

2.2.2.3 IsCached

IsCached: bit NOT NULL. A bit that specifies the frequency of the use of a MetadataObject. Protocol clients can use this as a recommendation as to whether to cache a MetadataObject in memory.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>The MetadataObject is infrequently used.</td>
</tr>
<tr>
<td>1</td>
<td>The MetadataObject is frequently used.</td>
</tr>
</tbody>
</table>

2.2.2.4 EstimatedInstanceCount

EstimatedInstanceCount: int NOT NULL. The maximum number of instances of an entity estimated to be returned from a physical LOB system represented by a LobSystemInstance.

2.2.2.5 MetadataObjectType

MetadataObjectType: nvarchar(255). The type of a MetadataObject. The value MUST be in the following table. If the length is greater than 250, then the characters after 250th character are ignored by the protocol server.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Microsoft.Office.Server.ApplicationRegistry.MetadataModel.AccessControlledMetadataObject</td>
<td>Identifies the set of MetadataObject types that have access control entry (ACE) associated with them.</td>
</tr>
<tr>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>Microsoft.Office.Server.ApplicationRegistry.MetadataModel.MetadataObject</td>
<td>Identifies the set of MetadataObject types that have Property and localized names associated with them.</td>
</tr>
</tbody>
</table>

2.2.2.6  Position

**Position**: tinyint NOT NULL. The order of an Action among the other Actions displayed in a user interface for an Entity.

2.2.2.7  IsDisplayed

**IsDisplayed**: bit NOT NULL. A bit that specifies whether an Action is displayed in the user interface presented to the user. The application that uses the protocol client typically uses this value as a guidance to represent the Action in the user interface.

2.2.2.8  IsCollection

**IsCollection**: bit NOT NULL. A bit that specifies whether a TypeDescriptor MUST be interpreted as a collection of native LOB System data structures.

2.2.2.9  IsOpenedInNewWindow

**IsOpenedInNewWindow**: bit NOT NULL. A bit that specifies whether the results of executing an Action are displayed in a new window in the user interface presented to the user. The application that uses the protocol client typically uses this value as guidance on creating new window when the Action is executed.

2.2.2.10  Icon

**Icon**: nvarchar(2080). The URL of the icon associated with an Action.

2.2.2.11  URL

**URL**: nvarchar(2080) NOT NULL. The URL associated with an Action.

2.2.2.12  Index

**Index**: tinyint NOT NULL. A value, indicating the position of an ActionParameter among the ActionParameters of the Action that contains the ActionParameter.

2.2.2.13  FilterDescriptorTypeName

**FilterDescriptorTypeName**: nvarchar(255) NOT NULL. The type of a FilterDescriptor. The value MUST be in the following table.
<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microsoft.Office.Server.ApplicationRegistry.Runtime.EqualsFilter</td>
<td>Indicates that a FilterDescriptor describes a ComparisonFilter with its comparator set to '=='.</td>
</tr>
</tbody>
</table>

### 2.2.2.14 IdentifierTypeName

**IdentifierTypeName**: nvarchar(255) NOT NULL. The type of the identifiers of instances of an Entity returned from an LOB system. The value MUST be in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>System.String</td>
<td>Defines the identifier of instances of the Entity to be strings of Unicode text.</td>
</tr>
<tr>
<td>System.Int16</td>
<td>Defines the identifier of instances of the Entity to be a number ranging from negative 32768 to positive 32767.</td>
</tr>
</tbody>
</table>
### Value Description

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>System.Int32</td>
<td>Defines the identifier of instances of the Entity to be a number ranging from negative 2,147,483,648 to positive 2,147,483,647.</td>
</tr>
<tr>
<td>System.Int64</td>
<td>Defines the identifier of instances of the Entity to be a number ranging from negative 9,223,372,036,854,775,808 to positive 9,223,372,036,854,775,807.</td>
</tr>
<tr>
<td>System.UInt16</td>
<td>Defines the identifier of instances of the Entity to be a number ranging from 0 to 65535.</td>
</tr>
<tr>
<td>System.UInt32</td>
<td>Defines the identifier of instances of the Entity to be a number ranging from 0 to 4,294,967,295.</td>
</tr>
<tr>
<td>System.UInt64</td>
<td>Defines the identifier of instances of the Entity to be a number ranging from 0 to 18,446,744,073,709,551,615.</td>
</tr>
<tr>
<td>System.DateTime</td>
<td>Defines the identifier of instances of the Entity to be a date and time ranging from 12:00:00 midnight, January 1, 1 Anno Domini (Common Era) to 11:59:59 P.M., December 31, 9999 Anno Domini (Common Era).</td>
</tr>
<tr>
<td>System.Single</td>
<td>Defines the identifier of instances of the Entity to be a single precision number ranging from negative 3.402823e38 to 3.402823e38.</td>
</tr>
<tr>
<td>System.Double</td>
<td>Defines the identifier of instances of the Entity to be a double precision number ranging from negative 1.79769313486232e308 to positive 1.79769313486232e308 as well as positive or negative zero, positive infinity, negative infinity and NaN.</td>
</tr>
<tr>
<td>System.Decimal</td>
<td>Defines the identifier of instances of the Entity to be a number ranging from negative 79,228,162,514,264,337,593,543,950,335 to positive 79,228,162,514,264,337,593,543,950,335.</td>
</tr>
<tr>
<td>System.Char</td>
<td>Defines the identifier of instances of the Entity to be a Unicode character.</td>
</tr>
<tr>
<td>System.Byte</td>
<td>Defines the identifier of instances of the Entity to be a number ranging from 0 to 255.</td>
</tr>
<tr>
<td>System.SByte</td>
<td>Defines the identifier of instances of the Entity to be a number ranging from negative 128 to positive 127.</td>
</tr>
<tr>
<td>Guid</td>
<td>Defines the identifier of instances of the Entity to be a GUID.</td>
</tr>
</tbody>
</table>

#### 2.2.2.15 MethodInstanceType

**MethodInstanceType**: tinyint NOT NULL. The type of a MethodInstance. The value MUST be in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Indicates that a MethodInstance should be interpreted as a Finder by the protocol client.</td>
</tr>
<tr>
<td>2</td>
<td>Indicates that a MethodInstance should be interpreted as a SpecificFinder by the protocol client.</td>
</tr>
<tr>
<td>3</td>
<td>Indicates that a MethodInstance should be interpreted as a ViewAccessor by the protocol client.</td>
</tr>
<tr>
<td>4</td>
<td>Indicates that a MethodInstance should be interpreted as a GenericInvoker by the protocol client.</td>
</tr>
<tr>
<td>5</td>
<td>Indicates that a MethodInstance should be interpreted as an IdEnumerator by the protocol client.</td>
</tr>
</tbody>
</table>
### 2.2.2.16 Direction

**Direction:** tinyint NOT NULL. The direction of the Parameter while calling the Method that contains the Parameter. The value MUST be in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Used for input purposes only.</td>
</tr>
<tr>
<td>2</td>
<td>Used for output purposes only.</td>
</tr>
<tr>
<td>3</td>
<td>Used for input before calling the LOB system and then for reading the output data once the call is complete.</td>
</tr>
<tr>
<td>4</td>
<td>Used to indicate that a parameter is the formal return parameter.</td>
</tr>
</tbody>
</table>

### 2.2.2.17 TypeReflectorTypeName

**TypeReflectorTypeName:** nvarchar(255) NOT NULL. Identifies the TypeReflector associated with a Parameter. The value MUST be in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
</table>

### 2.2.2.18 TypeDescriptorTypeName

**TypeDescriptorTypeName:** nvarchar(255) NOT NULL. The name of a unit of implementation-specific business logic that exists in an implementation-specific Business Logic Module.

### 2.2.2.19 ConnectionManagerTypeName

**ConnectionManagerTypeName:** nvarchar(255) NOT NULL. The name of the connection manager that will be used while connecting to this LobSystem. The value MUST be in the following table.
### 2.2.2.20 SystemUtilityTypeName

**SystemUtilityTypeName:** nvarchar(255) NOT NULL. The name of the Implementation-Specific logic module that will be used to execute the Methods in this LobSystem. The value MUST be in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
</table>

### 2.2.2.21 EntityInstanceTypeName

**EntityInstanceTypeName:** nvarchar(255) NOT NULL. The value MUST be in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
</table>
2.2.2.22 MetadataRights

**MetadataRights:** bigint NOT NULL. The permissions available to a *security principal* for a MetadataObject. The value MUST be in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00</td>
<td>No permissions.</td>
</tr>
<tr>
<td>0x01</td>
<td>Ability to call a <em>MethodInstance</em>.</td>
</tr>
<tr>
<td>0x02</td>
<td>Ability to change the attributes of a MetadataObject or its relationship to other MetadataObjects.</td>
</tr>
<tr>
<td>0x04</td>
<td>Ability to change the permissions associated with a MetadataObject.</td>
</tr>
<tr>
<td>A positive value</td>
<td>Implementation-specific abilities.</td>
</tr>
</tbody>
</table>

2.2.2.23 IsStatic

**IsStatic:** bit NOT NULL. A bit that specifies whether the Method is associated with an *EntityInstance*. The value MUST be in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>The Method operates in the context of a specific EntityInstance.</td>
</tr>
<tr>
<td>1</td>
<td>The Method operates out of the context of a specific EntityInstance.</td>
</tr>
</tbody>
</table>

2.2.3 Bit Fields and Flag Structures

None.

2.2.4 Binary Structures

None.

2.2.5 Result Sets

The following common result sets are used by this protocol.

2.2.5.1 Action Result Set

The *Action* result set contains information about actions. Each row in the result set contains all the attributes of a single *Action*.

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

```sql
Id             int,
EntityId       int,
Position       tinyint,
IsDisplayed    bit,
IsOpenedInNewWindow bit,
Icon           nvarchar(2080),
```
URL nvarchar(2080),
Name nvarchar(255),
IsCached bit,
Version int;

**Id:** The MetadataObjectId of the Action that is returned. The value MUST be an Id, as specified in section 2.2.2.1.

**EntityId:** The MetadataObjectId of the Entity that contains this Action. The value MUST be an Id.

**Position:** The order of this Action among the other Actions displayed in a user interface for this Entity. The value MUST be a Position, as specified in section 2.2.2.6.

**IsDisplayed:** A bit that provides a hint on whether this Action is displayed in the user interface presented to the user. The value MUST be an IsDisplayed, as specified in section 2.2.2.7.

**IsOpenedInNewWindow:** A bit that provides a hint on whether the results of executing this Action are displayed in a new window in the user interface presented to the user. The value MUST be an IsOpenedInNewWindow, as specified in section 2.2.2.9.

**Icon:** The URL of the icon associated with the Action. The value MUST be an Icon, as specified in section 2.2.2.10.

**URL:** The URL associated with the Action. The value MUST be a URL, as specified in section 2.2.2.11.

**Name:** The name of the Action. The value MUST be a Name, as specified in section 2.2.2.2.

**IsCached:** A bit that specifies whether this Action is frequently used. The value MUST be an IsCached, as specified in section 2.2.2.3.

**Version:** The object version of this Action.

### 2.2.5.2 ActionParameter Result Set

The ActionParameter result set contains information about action parameters. Each row in the result set contains all the attributes of a single ActionParameter.

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

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>int</td>
</tr>
<tr>
<td>ActionId</td>
<td>int</td>
</tr>
<tr>
<td>Index</td>
<td>tinyint</td>
</tr>
<tr>
<td>Name</td>
<td>nvarchar(255)</td>
</tr>
<tr>
<td>IsCached</td>
<td>bit</td>
</tr>
<tr>
<td>Version</td>
<td>int</td>
</tr>
</tbody>
</table>

**Id:** The MetadataObjectId of the ActionParameter that is returned. The value MUST be an Id, as specified in section 2.2.2.1.

**ActionId:** The MetadataObjectId of the Action that contains this ActionParameter. The value MUST be an Id.
Index: A value indicating the position of this ActionParameter among the other ActionParameters in the Action that contains this ActionParameter. The value MUST be an Index, as specified in section 2.2.2.12.

Name: The name of the ActionParameter. The value MUST be a Name, as specified in section 2.2.2.2.

IsCached: A bit that specifies whether this ActionParameter is frequently used. The value MUST be an IsCached, as specified in section 2.2.2.3.

Version: The object version of this ActionParameter.

2.2.5.3 Association Result Set

The Association result set contains information about associations. Each row in the result set contains all the attributes of a single Association.

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

```
Id                      int,
MethodId                int,
ReturnTypeDescriptorId  int,
Type                    tinyint,
Name                    nvarchar(255),
IsCached                bit,
Version                 int;
```

Id: The MetadataObjectId of the Association that is returned. The value MUST be an Id, as specified in section 2.2.2.1.

MethodId: The MetadataObjectId of the Method that contains the Association. The value MUST be an Id.

ReturnTypeDescriptorId: The MetadataObjectId of the ReturnTypeDescriptor. The value MUST be an Id.

Type: The type of the MethodInstance. The value MUST be a MethodInstanceType, as specified in section 2.2.2.15.

Name: The name of the Association. The value MUST be a Name, as specified in section 2.2.2.2.

IsCached: A bit that specifies whether this Association is frequently used. The value MUST be an IsCached, as specified in section 2.2.2.3.

Version: The object version of this Association.

2.2.5.4 Count Result Set

The Count result set contains the number of rows expected in the subsequent result set. The stored procedures that return the Count result set MUST also return a subsequent result set with rows that represent MetadataObjects, version stamp for the cache, or localized names. The protocol client can use the count result set to initialize collections with enough size to store items that are returned in the immediately following result set.

The T-SQL syntax for the result set is as follows:
2.2.5.5 DataClass Result Set

The dataClass result set contains information about DataClasses. Each row in the result set contains all the attributes of a single DataClass.

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

```sql
Id                    int,
SystemId              int,
Name                  nvarchar(255),
IsCached              bit,
Version               int;
```

Id: The MetadataObjectId of the DataClass that is returned. The value MUST be an Id, as specified in section 2.2.2.1.

SystemId: The MetadataObjectId of the LobSystem that contains this DataClass. The value MUST be an Id.

Name: The name of this DataClass. The value MUST be a Name, as specified in section 2.2.2.2.

IsCached: A bit that specifies whether this DataClass is frequently used. The value MUST be an IsCached, as specified in section 2.2.2.3.

Version: The object version of this DataClass.

2.2.5.6 Entity Result Set

The Entity result set contains information about entities. Each row in the result set contains all the attributes of a single Entity.

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

```sql
Id                      int,
EstimatedInstanceCount  int,
SystemId                int,
Name                    nvarchar(255),
IsCached                bit,
Version                 int;
```

Id: The MetadataObjectId of the Entity that is returned. The value MUST be an Id, as specified in section 2.2.2.1.

EstimatedInstanceCount: The maximum estimated number of instances of this Entity returned from the LobSystemInstance. The value MUST be an EstimatedInstanceCount, as specified in section 2.2.2.4.

SystemId: The MetadataObjectId of the LobSystem that contains this Entity. The value MUST be an Id.
**Name:** The name of this Entity. The value MUST be a Name, as specified in section 2.2.2.

**IsCached:** A bit that specifies whether this Entity is frequently used. The value MUST be an IsCached, as specified in section 2.2.3.

**Version:** The object version of this Entity.

### 2.2.5.7 FilterDescriptor Result Set

The FilterDescriptor result set contains information about FilterDescriptors. Each row in the result set contains all the attributes of a single FilterDescriptor.

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

```sql
Id                   int,
TypeName             nvarchar(255),
MethodId             int,
Name                 nvarchar(255),
IsCached             bit,
Version              int;
```

**Id:** The MetadataObjectId of the FilterDescriptor that is returned. The value MUST be an Id, as specified in section 2.2.2.1.

**TypeName:** Identifies the type of a FilterDescriptor. The value MUST be a FilterDescriptorTypeName, as specified in section 2.2.2.13.

**MethodId:** The MetadataObjectId of the Method that contains this FilterDescriptor. The value MUST be an Id.

**Name:** The name of this FilterDescriptor. The value MUST be a Name, as specified in section 2.2.2.2.

**IsCached:** A bit that specifies whether this FilterDescriptor is frequently used. The value MUST be an IsCached, as specified in section 2.2.3.

**Version:** The object version of this FilterDescriptor.

### 2.2.5.8 Identifier Result Set

The Identifier result set contains information about Identifiers. Each row in the result set contains all the attributes of a single Identifier.

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

```sql
Id                     int,
TypeName               nvarchar(255),
EntityId               int,
OrdinalNumber          tinyint,
Name                   nvarchar(255),
IsCached               bit,
Version                int;
```

**Id:** The MetadataObjectId of the Identifier that is returned. The value MUST be an Id, as specified in section 2.2.2.1.
**TypeName:** The type name of the identifiers of instances of an **Entity** returned from an LOB system. The value MUST be an **IdentifierTypeName**, as specified in section 2.2.2.14.

**EntityId:** The **MetadataObjectId** of the **Entity** that contains this **Identifier**. The value MUST be an **Id**.

**OrdinalNumber:** The sequence number for the **Identifier** that imposes a deterministic ordering of all **Identifiers** of an **Entity**. The rows in the result set MUST be ordered by increasing **OrdinalNumber**.

**Name:** The name of this **Identifier**. The value MUST be a **Name**, as specified in section 2.2.2.2.

**IsCached:** A bit that specifies whether this **Identifier** is frequently used. The value MUST be an **IsCached**, as specified in section 2.2.2.3.

**Version:** The object version of this **Identifier**.

### 2.2.5.9 Method Result Set

The **Method** result set contains information about **Methods**. Each row in the result set contains all the attributes of a single **Method**.

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

```sql
Id int,
ClassId int,
IsStatic bit,
Name nvarchar(255),
IsCached bit,
Version int;
```

**Id:** The **MetadataObjectId** of the **Method** that is returned. The value MUST be an **Id**, as specified in section 2.2.2.1.

**ClassId:** The **MetadataObjectId** of the **DataClass** of this **Method**.

**IsStatic:** A bit that specifies whether the **Method** is associated with an **EntityInstance**. The value MUST be an **IsStatic**, as specified in section 2.2.2.3.

**Name:** The name of this **Method**. The value MUST be a **Name**, as specified in section 2.2.2.2.

**IsCached:** A bit that specifies whether this **Method** is frequently used. The value MUST be an **IsCached**, as specified in section 2.2.2.3.

**Version:** The object version of this **Method**.

### 2.2.5.10 MethodInstance Result Set

The **MethodInstance** result set contains information about **MethodInstances**. Each row in the result set contains all the attributes of a single **MethodInstance**.

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

```sql
Id int,
MethodId int,
ReturnTypeDescriptorId int,
```

---

[MS-BDCSP] — v20120630
*Business Data Catalog Database Protocol Specification*

Copyright © 2012 Microsoft Corporation.

Release: July 16, 2012
Type                    tinyint,
Name                    nvarchar(255),
IsCached                bit,
Version                 int;

Id: The MetadataObjectId of the MethodInstance that is returned. The value MUST be an Id, as specified in section 2.2.2.1.

MethodId: The MetadataObjectId of the Method that contains this MethodInstance. The value MUST be an Id.

ReturnTypeDescriptorId: The MetadataObjectId of the ReturnTypeDescriptor. The value MUST be an Id.

Type: The type of this MethodInstance. The value MUST be a MethodInstanceType, as specified in section 2.2.2.15.

Name: The name of this MethodInstance. The value MUST be a Name, as specified in section 2.2.2.2.

IsCached: A bit that specifies whether this MethodInstance is frequently used. The value MUST be an IsCached, as specified in section 2.2.2.3.

Version: The object version of this MethodInstance.

2.2.5.11 Parameter Result Set

The Parameter result set contains information about Parameters. Each row in the result set contains all the attributes of a single Parameter.

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

Id                      int,
MethodId                int,
Direction               tinyint,
OrdinalNumber           tinyint,
TypeReflectorTypeName   nvarchar(255),
Name                    nvarchar(255),
IsCached                bit,
Version                 int;

Id: The MetadataObjectId of the Parameter that is returned. The value MUST be an Id, as specified in section 2.2.2.1.

MethodId: The MetadataObjectId of the Method that contains this Parameter. The value MUST be an Id.

Direction: The direction of the Parameter while calling the Method this Parameter is contained by. The value MUST be a Direction, as specified in section 2.2.16.

OrdinalNumber: The sequence number that imposes a deterministic ordering of the Parameters while using them to call the Method this Parameter is contained by. The rows in the result set MUST be returned with OrdinalNumbers in ascending order.
TypeReflectorTypeName: The name of the TypeReflector associated with a Parameter. The value MUST be a TypeReflectorTypeName, as specified in section 2.2.2.17.

Name: The name of this Parameter. The value MUST be a Name, as specified in section 2.2.2.

IsCached: A bit that specifies whether this Parameter is frequently used. The value MUST be an IsCached, as specified in section 2.2.3.

Version: The object version of this Parameter.

2.2.5.12 System Instance Result Set

The System Instance result set contains information about LobSystemInstances. Each row in the result set contains all the attributes of a single LobSystemInstance.

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

```sql
Id                     int,
SystemId               int,
Name                   nvarchar(255),
IsCached               bit,
Version                int;
```

Id: The MetadataObjectId of the LobSystemInstance that is returned. The value MUST be an Id, as specified in section 2.2.2.1.

SystemId: The MetadataObjectId of the LobSystem that contains this LobSystemInstance. The value MUST be an Id.

Name: The name of this LobSystemInstance. The value MUST be a Name, as specified in section 2.2.2.

IsCached: A bit that specifies whether this LobSystemInstance is frequently used. The value MUST be an IsCached, as specified in section 2.2.3.

Version: The object version of this LobSystemInstance.

2.2.5.13 System Result Set

The System result set contains information about LobSystems. Each row in the result set contains all the attributes of a single LobSystem.

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

```sql
Id                         int,
ConnectionFactoryTypeName  nvarchar(255),
SystemUtilityTypeName      nvarchar(255),
SystemEntityTypeName       nvarchar(255),
Name                       nvarchar(255),
IsCached                   bit,
Version                    int;
```

Id: The MetadataObjectId of the LobSystem that is returned. The value MUST be an Id, as specified in section 2.2.2.1.
**ConnectionFactoryTypeName**: The name of the connection manager that will be used while connecting to this LOB system. The value MUST be a **ConnectionManagerTypeName**, as specified in section 2.2.2.19.

**SystemUtilityTypeName**: The name of the system utility that is used to execute **Methods** in this **LobSystem**. The value MUST be a **SystemUtilityTypeName**, as specified in section 2.2.2.20.

**SystemEntityTypeName**: The name of the unit of implementation-specific <3> business logic (2) that is used to create the objects that carry **EntityInstance** data to client applications. The value MUST be a **EntityInstanceTypeName**, as specified in section 2.2.2.21.

**Name**: The name of this **LobSystem**. The value MUST be a **Name**, as specified in section 2.2.2.2.

**IsCached**: A bit that specifies whether this **LobSystem** is frequently used. The value MUST be an **IsCached**, as specified in section 2.2.2.3.

**Version**: The object version of this **LobSystem**.

### 2.2.5.14 TypeDescriptor Result Set

The **TypeDescriptor** result set contains information about **TypeDescriptors**. Each row in the result set contains all the attributes of a single **TypeDescriptor**.

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

```
Id                int,
ParameterId       int,
ParentTypeDescriptorId int,
TypeName          nvarchar(255),
InterpretedTypeName nvarchar(255),
ContainsIdentifier bit,
IdentifierId      int,
ContainsFilterDescriptor bit,
FilterDescriptorId int,
IsCollection      bit,
Name              nvarchar(255),
IsCached          bit,
Version           int;
```

**Id**: The **MetadataObjectId** of the **TypeDescriptor** that is returned. The value MUST be an **Id**, as specified in section 2.2.2.1.

**ParameterId**: The **MetadataObjectId** of the **Parameter** that contains this **TypeDescriptor**. The value MUST be an **Id**.

**ParentTypeDescriptorId**: The **MetadataObjectId** of the parent **TypeDescriptor** that contains this **TypeDescriptor**. The value MUST be an **Id**.

**TypeName**: The name of a unit of implementation-specific <4> business logic (2) that exists in an implementation-specific <5> business logic module. The value MUST be a **TypeDescriptorTypeName**, as specified in section 2.2.2.18.

**InterpretedTypeName**: Same as the **TypeName**. The value MUST be a **TypeDescriptorTypeName**.

**ContainsIdentifier**: A bit that specifies if any **TypeDescriptor** in the TypeDescriptor tree of this **TypeDescriptor** references an **Identifier**.
IdentifierId: The MetadataObjectId of the Identifier associated with this TypeDescriptor. If this is NULL, this TypeDescriptor does not have an associated Identifier. If this is not NULL, ContainsIdentifier MUST be "1".

ContainsFilterDescriptor: A bit that specifies if any TypeDescriptor in the TypeDescriptor tree of this TypeDescriptor has an associated FilterDescriptor.

FilterDescriptorId: The MetadataObjectId of the FilterDescriptor associated with this TypeDescriptor. If this is NULL, this TypeDescriptor does not have an associated FilterDescriptor. If this is not NULL, ContainsFilterDescriptor MUST be "1".

IsCollection: A bit that specifies whether this TypeDescriptor is to be interpreted by protocol clients as a collection of native LOB System data structures. The value MUST be an IsCollection, as specified in section 2.2.2.8.

Name: The name of this TypeDescriptor. The value MUST be a Name, as specified in section 2.2.2.

IsCached: A bit that specifies whether this TypeDescriptor is frequently used. The value MUST be an IsCached, as specified in section 2.2.3.

Version: The object version of this TypeDescriptor.

2.2.6 Tables and Views
None.

2.2.7 XML Structures
No common XML Structures are defined in this protocol.
3 Protocol Details

3.1 Back End Database Server Details

The back-end database protocol server responds only to stored procedure calls from the protocol client. It returns result sets and return codes and never initiates communication with other endpoints of the protocol.

3.1.1 Abstract Data Model

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

For this protocol, the back-end database server maintains lists to store the attributes of each of the following MetadataObjects:

- Action
- ActionParameter
- Association
- DataClass
- Entity
- FilterDescriptor
- Identifier
- LobSystem
- LobSystemInstance
- Method
- MethodInstance
- Parameter
- TypeDescriptor

The server MUST maintain a set of relations between these MetadataObject types. These relations are the following:

- Each LobSystemInstance is contained by a LobSystem.
- Each Entity is contained by a LobSystem.
- Each Method is contained by an Entity.
- Each MethodInstance is contained by a Method.
- Each Parameter is contained by a Method.
Each **TypeDescriptor** that is not a root **TypeDescriptor** is contained by a **TypeDescriptor**.

Each root **TypeDescriptor** is contained by a **Parameter**.

Each **FilterDescriptor** is contained by a **Method**.

Each **Identifier** is contained by an **Entity**.

Each **Association**<sup>6</sup> is contained by a **Method**.

Each **DataClass** is contained by a **LobSystem**.

Each **Action** is contained by an **Entity**.

Each **ActionParameter** is contained by an **Action**.

The implementations of the basic create, read, update, and delete stored procedures simply insert, read, update, and delete items in each of these lists where the **MetadataObjectId** serves as the primary identifier.

The ACEs, localized names, and **MetadataObject** properties are also stored in their own lists along with the **MetadataObjectId** of the associated **MetadataObject**.

### 3.1.2 Timers

None.

### 3.1.3 Initialization

None.

### 3.1.4 Higher-Layer Triggered Events

None

### 3.1.5 Message Processing Events and Sequencing Rules

The T-SQL syntax for each stored procedure and result set, and the variables they are composed of, is defined in the [MSDN-TSQL-Ref](#) protocol. In the T-SQL syntax, the variable name is followed by the type of the variable that can optionally have a length value in brackets and can optionally have a default value indicated by an equals sign followed by the default value. Unless otherwise specified, all stored procedures defined in this section are located in the metadata store.

For definitional clarity, a name has been assigned to any columns in the result sets that do not have a defined name in their current implementation. This does not affect the operation of the result set, as the ordinal position of any column with no defined name is expected by the **front-end Web server**. Such names are designated in the text using curly braces in the form `{name}`. The stored procedures that return multiple result sets should return them in the order they are specified. For example, **proc_ar_GetActionParametersForActionWithCount** should return the **Count** result set followed by the **Action** result set.

#### 3.1.5.1 proc_ar_AddOrInsertLocalizedNameForMetadataObjectId

The **proc_ar_AddOrInsertLocalizedNameForMetadataObjectId** stored procedure is called to add a localized name for a **MetadataObject** in the specified locale. If a localized name already exists for the specified locale, it MUST be replaced by the specified localized name.
The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_ar_AddOrInsertLocalizedNameForMetadataObjectId(
    @MetadataObjectId       int,
    @LocalizedName          nvarchar(255),
    @LCID                   int,
    @ErrorCode              int OUTPUT
);
```

**@MetadataObjectId:** The MetadataObjectId of the MetadataObject to which this localized name is added or replaced. The value MUST be an Id, as specified in section 2.2.2.1.

**@LocalizedName:** The localized name of this MetadataObject in the specified locale. If a localized name already exists for the specified locale and the specified MetadataObject, it MUST be replaced by the specified localized name.

**@LCID:** The LCID representing the locale of the specified localized name.

**@ErrorCode:** The error code. Upon return from this stored procedure, this parameter MUST be set to an integer that is listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-1</td>
<td>An error occurred while adding or replacing the specified localized name.</td>
</tr>
<tr>
<td>-3</td>
<td>The MetadataObject already contains the implementation-specific maximum allowed number of localized names.</td>
</tr>
<tr>
<td></td>
<td>A positive integer A T-SQL error code.</td>
</tr>
</tbody>
</table>

**Return Code Values:** An integer that the protocol client MUST ignore.

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

### 3.1.5.2 proc_ar_AddOrInsertPropertyForMetadataObjectId

The proc_ar_AddOrInsertPropertyForMetadataObjectId stored procedure is called to add a Property for a MetadataObject. If a Property with the specified name already exists for the specified MetadataObject, its value MUST be replaced by the specified value.

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

```sql
PROCEDURE proc_ar_AddOrInsertPropertyForMetadataObjectId(
    @MetadataObjectId            int,
    @Name                        nvarchar(255),
    @Value                       sql_variant,
    @ErrorCode                   int OUTPUT
);
```

**@MetadataObjectId:** The MetadataObjectId of the MetadataObject to which this Property is added or replaced. The value MUST be an Id, as specified in section 2.2.2.1.
@Name: The programmatic name of the Property. If a Property with this name is already associated with the specified MetadataObject, its value MUST be replaced with the specified value.

@Value: The value of the Property.

@ErrorCode: An error code. Upon return from this stored procedure, this parameter MUST be set to an integer that is listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-1</td>
<td>An error occurred while adding or replacing the specified Property.</td>
</tr>
<tr>
<td>-2</td>
<td>A MetadataObject with the specified @MetadataObjectId does not exist.</td>
</tr>
<tr>
<td>-3</td>
<td>The MetadataObject already contains the implementation-specific maximum allowed number of Properties.</td>
</tr>
<tr>
<td></td>
<td>A positive integer A T-SQL SQL error code.</td>
</tr>
</tbody>
</table>

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MUST NOT return any result sets.

3.1.5.3 proc_ar_BumpCacheInvalidationCounter

The proc_ar_BumpCacheInvalidationCounter stored procedure is called to increment the cache version stamp for the given MetadataObject MetadataObjectType, as specified in section 2.2.2.5, if the cache version stamp is not at an implementation-specific maximum value. If the cache version stamp is at the implementation-specific maximum value, the protocol server MUST set it to zero.

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

```
PROCEDURE proc_ar_BumpCacheInvalidationCounter(
    @MetadataObjectType    nvarchar(250),
    @ObjectCache           bit
);
```

@MetadataObjectType: The type name of the MetadataObject whose cache version stamp is incremented. The value MUST be a MetadataObjectType, as specified in section 2.2.2.5.

@ObjectCache: A bit that specifies the type of the cache version stamp to be affected. The value of this parameter MUST be listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>The stored procedure MUST increment the version stamp of the relationship cache for the specified MetadataObject MetadataObjectType.</td>
</tr>
<tr>
<td>1</td>
<td>The stored procedure MUST increment the version stamp of the object cache for the specified MetadataObject MetadataObjectType.</td>
</tr>
</tbody>
</table>

Return Code Values: An integer that the protocol client MUST ignore.
Result Sets: MUST NOT return any result sets.

### 3.1.5.4 proc_ar_ClearAccessControlEntriesForMetadataObject

The **proc_ar_ClearAccessControlEntriesForMetadataObject** stored procedure is called to delete all ACEs associated with the specified MetadataObject.

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

```sql
PROCEDURE proc_ar_ClearAccessControlEntriesForMetadataObject(
    @MetadataObjectId int
);
```

@MetadataObjectId: The MetadataObjectId of the MetadataObject whose ACEs are deleted.

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MUST NOT return any result sets.

### 3.1.5.5 proc_ar_CopyAccessControlEntriesForMetadataObjectId

The **proc_ar_CopyAccessControlEntriesForMetadataObjectId** stored procedure is called to copy ACEs for a MetadataObject from another MetadataObject. If @SourceMetadataObjectId and @DestinationMetadataObjectId are equal, this stored procedure MUST make no changes. If @SourceMetadataObjectId and @DestinationMetadataObjectId are not equal, this stored procedure MUST delete all ACEs associated with the specified target MetadataObject and then copy the ACEs associated with the specified source MetadataObject.

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

```sql
PROCEDURE proc_ar_CopyAccessControlEntriesForMetadataObjectId(
    @SourceMetadataObjectId int,
    @DestinationMetadataObjectId int
);
```

@SourceMetadataObjectId: The MetadataObjectId of the source MetadataObject from which the ACEs are copied. The value MUST be an Id, as specified in section 2.2.2.1.

@DestinationMetadataObjectId: The MetadataObjectId of the target MetadataObject to which the ACEs are copied. The value MUST be an Id.

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MUST NOT return any result sets.

### 3.1.5.6 proc_ar_CreateAction

The **proc_ar_CreateAction** stored procedure is called to create an Action in the specified Entity.

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

```sql
PROCEDURE proc_ar_CreateAction(
    @Name nvarchar(50),
    @IsCached bit,
    @EntityId int,
);
```
@Position: The name of the Action. The value MUST be a Name, as specified in section 2.2.2.2.

@IsCached: A bit that specifies whether this Action is frequently used. The value MUST be an IsCached, as specified in section 2.2.2.3.

@EntityId: The MetadataObjectId of the Entity that contains this Action. The value MUST be an Id.

@Position: The order of this Action among the other Actions displayed in a user interface for this Entity. The value MUST be a Position, as specified in section 2.2.2.6.

@IsDisplayed: A bit that provides a hint on whether the Action is displayed in the user interface presented to the user. The value MUST be an IsDisplayed, as specified in section 2.2.2.7.

@IsOpenedInNewWindow: A bit that provides a hint on whether the results of executing the Action are displayed in a new window in the user interface presented to the user. The value MUST be an IsOpenedInNewWindow, as specified in section 2.2.2.9.

@Icon: The URL of the icon associated with this Action. The value MUST be an Icon, as specified in section 2.2.2.10.

@URL: The URL associated with this Action. The value MUST be an URL, as specified in section 2.2.2.11.

@CreatedId: The identifier for the newly created Action. Upon return from this stored procedure with an @ErrorCode set to zero, this parameter MUST be set to the MetadataObjectId of the newly created Action. Upon return from this stored procedure with an @ErrorCode set to a value other than zero, this parameter is set to a value that MUST be ignored.

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-1</td>
<td>The Entity already contains another Action with the specified @Name.</td>
</tr>
<tr>
<td>-3</td>
<td>The Entity already contains the implementation-specific maximum allowed number of Actions.</td>
</tr>
<tr>
<td>A positive integer</td>
<td>A T-SQL error code.</td>
</tr>
</tbody>
</table>

Return Code Values: An integer that the protocol client MUST ignore.

Results Sets: MAY <7>return zero or more result sets that the protocol client MUST ignore.
3.1.5.7 proc_ar_CreateActionParameter

The proc_ar_CreateActionParameter stored procedure is called to create an ActionParameter in the specified Action.

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

```sql
PROCEDURE proc_ar_CreateActionParameter(
    @Name                      nvarchar(50),
    @IsCached                  bit,
    @ActionId                  int,
    @Index                     tinyint,
    @CreatedId                 int OUTPUT,
    @ErrorCode                 int OUTPUT
);
```

@Name: The name of the ActionParameter. The value MUST be a Name, as specified in section 2.2.2.2.

@IsCached: A bit that specifies whether this ActionParameter is frequently used. The value MUST be an IsCached, as specified in section 2.2.2.3.

@ActionId: The MetadataObjectId of the Action that contains this ActionParameter. It MUST be an Id, as specified in section 2.2.2.1.

@Index: A value indicating the position of this ActionParameter among the ActionParameters of the Action that contains this ActionParameter. It MUST be an Index, as specified in section 2.2.2.12.

@CreatedId: The MetadataObjectId of the newly created ActionParameter. Upon return from this stored procedure with an @ErrorCode set to zero, this parameter MUST be set to the MetadataObjectId of the newly created ActionParameter. Upon return from this stored procedure with an @ErrorCode set to a value other than zero, this parameter is set to a value that MUST be ignored.

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors occurred.</td>
</tr>
<tr>
<td>-1</td>
<td>The Action already contains another ActionParameter with the specified @Name.</td>
</tr>
<tr>
<td>-3</td>
<td>The Action already contains the implementation-specific maximum allowed number of ActionParameters.</td>
</tr>
<tr>
<td>Positive integer</td>
<td>A T-SQL error code.</td>
</tr>
</tbody>
</table>

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MAY return zero or more result sets that the protocol client MUST ignore.
3.1.5.8 proc_ar_CreateAssociation

The **proc_ar_CreateAssociation** stored procedure is called to create an **Association**. The stored procedure MUST copy the list of ACEs associated with the **Entity** that contains the **Method** that contains the **Parameter** that contains the **TypeDescriptor** whose **MetadataObjectId** is **@ReturnTypeDescriptorId** and associate them with the newly created **Association**.

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

```sql
PROCEDURE proc_ar_CreateAssociation (  
    @Name                      nvarchar(255),  
    @IsCached                  bit,  
    @ReturnTypeDescriptorId    int,  
    @SourceEntityIds           nvarchar(255),  
    @DestinationEntityId       int,  
    @CreatedId                 int OUTPUT,  
    @ErrorCode                 int OUTPUT  
);  

@Name: The name of the **Association**. The value MUST be a **Name**, as specified in section 2.2.2.2.

@IsCached: A bit that specifies where this **Association** is frequently used. The value MUST be an **IsCached**, as specified in section 2.2.2.3.

@ReturnTypeDescriptorId: The **MetadataObjectId** of the **ReturnTypeDescriptor**. The value MUST be an **Id**, as specified in section 2.2.2.1.

@SourceEntityIds: A comma-delimited list of **MetadataObjectIds** encoded as **Unicode strings** that represent the sources of the **Association**. For each value, there MUST exist an **Entity** with the specified **MetadataObjectId**. The value MUST NOT be NULL.

@DestinationEntityId: The **MetadataObjectId** of the **Entity** that represents the destination of the **Association**. The value MUST NOT be NULL.

@CreatedId: The identifier for the newly created **Association**. Upon return from this stored procedure with an **@ErrorCode** set to zero, this parameter MUST be set to the **MetadataObjectId** of the newly created **Association**. Upon return from this stored procedure with an **@ErrorCode** set to a value other than zero, this parameter is set to a value that MUST be ignored.

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-1</td>
<td>The <strong>LobSystem</strong> containing the <strong>Entity</strong> containing the <strong>Method</strong> containing the <strong>Parameter</strong> containing the <strong>TypeDescriptor</strong> with <strong>MetadataObjectId</strong> equal to <strong>@ReturnTypeDescriptorId</strong> already contains another <strong>Association</strong> with the specified <strong>@Name</strong>.</td>
</tr>
</tbody>
</table>
| -3    | Condition 1 MUST be true and condition 2 SHOULD <9> be true:  
|       | Condition 1:  
|       | The **LobSystem** containing the **Entity** containing the **Method** containing the **Parameter** containing the **TypeDescriptor** with **MetadataObjectId** equal to **@ReturnTypeDescriptorId** already contains the implementation-specific maximum allowed number of **Associations**. |
3.1.5.9 proc_ar_CreateEntity

The **proc_ar_CreateEntity** stored procedure is called to create an **Entity** in the specified **LobSystem**. The stored procedure MUST copy the list of ACEs associated with the specified **LobSystem** and associate them with the newly created **Entity**.

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

```sql
PROCEDURE proc_ar_CreateEntity] (  
  @Name                      nvarchar(255),  
  @IsCached                  bit,  
  @SystemId                  int,  
  @EstimatedInstanceCount    int,  
  @CreatedId                 int OUTPUT,  
  @ErrorCode                 int OUTPUT  
);  
```

**@Name**: The name of the **Entity**. The value MUST be a **Name**, as specified in section 2.2.2.2.

**@IsCached**: A bit that specifies whether this **Entity** is frequently used. The value MUST be an **IsCached**, as specified in section 2.2.2.3.

**@SystemId**: The **MetadataObjectId** of the **LobSystem** that contains this **Entity**. The value MUST be an **Id**, as specified in section 2.2.2.1.

**@EstimatedInstanceCount**: The estimated number of instances of this **Entity** present within the **LobSystemInstance**. The value MUST be an **EstimatedInstanceCount**, as specified in section 2.2.2.4. If this is NULL, the **Entity** MUST be created in the metadata store with a default value of "10000".

**@CreatedId**: The identifier for the newly created **Entity**. Upon return from this stored procedure with an **@ErrorCode** set to zero, this parameter MUST be set to the **MetadataObjectId** of the newly created **Entity**. Upon return from this stored procedure with an **@ErrorCode** set to a value other than zero, this parameter is set to a value that MUST be ignored.

**@ErrorCode**: The error code. Upon return from this stored procedure, this parameter MUST be set to an **integer** listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-1</td>
<td>The <strong>LobSystem</strong> already contains another <strong>Entity</strong> with the specified <strong>@Name</strong>.</td>
</tr>
<tr>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>-3</td>
<td>The LobSystem already contains the implementation-specific maximum allowed number of Entities.</td>
</tr>
<tr>
<td>A positive integer</td>
<td>A T-SQL error code.</td>
</tr>
</tbody>
</table>

**Return Code Values:** An integer that the protocol client MUST ignore.

**Result Sets:** MAY `<11>` return zero or more result sets that the protocol client MUST ignore.

### 3.1.5.10 proc_ar_CreateFilterDescriptor

The `proc_ar_CreateFilterDescriptor` stored procedure is called to create a FilterDescriptor in the specified Method.

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

```sql
PROCEDURE proc_ar_CreateFilterDescriptor (  @Name                      nvarchar(255),  @IsCached                  bit,  @MethodId                  int,  @TypeName                  nvarchar(255),  @CreatedId                 int OUTPUT,  @ErrorCode                 int OUTPUT);  
```

**@Name:** The name of the FilterDescriptor. The value MUST be a Name, as specified in section 2.2.2.2.

**@IsCached:** A bit that specifies whether this FilterDescriptor is frequently used. The value MUST be an IsCached, as specified in section 2.2.2.3.

**@MethodId:** The MetadataObjectId of the Method that contains this FilterDescriptor. The value MUST be an Id, as specified in section 2.2.2.1.

**@TypeName:** The type name of the FilterDescriptor. The value MUST be a FilterDescriptorTypeName, as specified in section 2.2.2.13.

**@CreatedId:** The identifier for the newly created FilterDescriptor. Upon return from this stored procedure with an @ErrorCode set to zero, this parameter MUST be set to the MetadataObjectId of the newly created FilterDescriptor. Upon return from this stored procedure with an @ErrorCode set to a value other than zero, this parameter is set to a value that MUST be ignored.

**@ErrorCode:** The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-1</td>
<td>The Method already contains another FilterDescriptor with the specified @Name.</td>
</tr>
<tr>
<td>-3</td>
<td>The Method already contains the implementation-specific maximum allowed number of FilterDescriptors.</td>
</tr>
<tr>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>A positive</td>
<td>A T-SQL error code.</td>
</tr>
<tr>
<td>integer</td>
<td></td>
</tr>
</tbody>
</table>

**Return Code Values:** An integer that the protocol client MUST ignore.

**Result Sets:** MAY `<12>` return zero or more result sets that the protocol client MUST ignore.

### 3.1.5.11 proc_ar_CreateIdentifier

The `proc_ar_CreateIdentifier` stored procedure is called to create an Identifier in the specified Entity.

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

```sql
PROCEDURE proc_ar_CreateIdentifier (  
    @Name                nvarchar(255),  
    @IsCached            bit,  
    @EntityId            int,  
    @TypeName            nvarchar(255),  
    @CreatedId           int OUTPUT,  
    @ErrorCode           int OUTPUT  
);  
```

@**Name**: The name of the Identifier. The value MUST be a Name, as specified in section 2.2.2.2.

@**IsCached**: A bit that specifies whether this Identifier is frequently used. The value MUST be an IsCached, as specified in section 2.2.2.3.

@**EntityId**: The MetadataObjectId of the Entity that this contains this Identifier. The value MUST be an Id, as specified in section 2.2.2.1.

@**TypeName**: The type name of the Identifier. The value MUST be an IdentifierTypeName, as specified in section 2.2.2.14.

@**CreatedId**: The identifier for the newly created Identifier. Upon return from this stored procedure with an @ErrorCode set to zero, this parameter MUST be set to the MetadataObjectId of the newly created Identifier. Upon return from this stored procedure with an @ErrorCode set to a value other than zero, this parameter is set to a value that MUST be ignored.

@**ErrorCode**: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-1</td>
<td>The Entity already contains another Identifier with the specified @Name.</td>
</tr>
<tr>
<td>-3</td>
<td>The Entity already contains the implementation-specific maximum number of</td>
</tr>
<tr>
<td></td>
<td>Identifiers.</td>
</tr>
<tr>
<td>A positive integer</td>
<td>A T-SQL error code.</td>
</tr>
</tbody>
</table>
Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MAY <13> return zero or more result sets that the protocol client MUST ignore.

3.1.5.12 proc_ar_CreateMethod

The proc_ar_CreateMethod stored procedure is called to create a Method in the specified DataClass. The stored procedure MUST copy the list of ACEs associated with the specified DataClass and associate them with the newly created Method.

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

```sql
PROCEDURE proc_ar_CreateMethod (
    @Name                nvarchar(255),
    @IsCached            bit,
    @ClassId             int,
    @IsStatic            bit,
    @CreatedId           int OUTPUT,
    @ErrorCode           int OUTPUT
);
```

@Name: The name of the Method. The value MUST be a Name, as specified in section 2.2.2.2.

@IsCached: A bit that specifies whether this Method is frequently used. The value MUST be an IsCached, as specified in section 2.2.2.3.

@ClassId: The MetadataObjectId of the DataClass that contains this Method. This MUST be Id, as specified in section 2.2.2.1.

@IsStatic: A bit specifying whether the Method is associated with an EntityInstance. The value MUST be an IsStatic, as specified in section 2.2.2.23.

@CreatedId: The identifier for the newly created Method. Upon return from this stored procedure with an @ErrorCode set to zero, this parameter MUST be set to the MetadataObjectId of the newly created Method. Upon return from this stored procedure with an @ErrorCode set to a value other than zero, this parameter is set to a value that MUST be ignored.

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-1</td>
<td>The DataClass already contains another Method with the specified @Name.</td>
</tr>
<tr>
<td>-3</td>
<td>The DataClass already contains the implementation-specific maximum allowed number of Methods.</td>
</tr>
<tr>
<td>A positive integer</td>
<td>A T-SQL error code.</td>
</tr>
</tbody>
</table>

Return Code Values: An integer that the protocol client MUST ignore.

Results Sets: MAY <14> return zero or more result sets that the protocol client MUST ignore.
3.1.5.13  proc_ar_CreateMethodInstance

The proc_ar_CreateMethodInstance stored procedure is called to create a MethodInvocation in the specified Method. The stored procedure MUST copy the list of ACEs associated with the parent DataClass of the specified Method and associate them with the newly created MethodInvocation.

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

```sql
PROCEDURE proc_ar_CreateMethodInstance (  
    @Name                   nvarchar(255),
    @IsCached               bit,
    @MethodId               int,
    @ReturnTypeDescriptorId int,
    @Type                   tinyint,
    @CreatedId              int OUTPUT,
    @ErrorCode              int OUTPUT
);  
```

@Name: The name of the MethodInvocation. The value MUST be a Name, as specified in section 2.2.2.2.

@IsCached: A bit that specifies whether this MethodInvocation is frequently used. The value MUST be an IsCached, as specified in section 2.2.2.3.

@MethodId: The MetadataObjectId of the Method that contains this MethodInvocation. The value MUST be an Id, as specified in section 2.2.2.4.

@ReturnTypeDescriptorId: The MetadataObjectId of the ReturnTypeDescriptor. The value MUST be an Id.

@Type: The type of the MethodInvocation. The value MUST be a MethodInvocationType, as specified in section 2.2.2.15.

@CreatedId: The identifier for the newly created MethodInvocation. Upon return from this stored procedure with an @ErrorCode set to zero, this parameter MUST be set to the MetadataObjectId of the newly created MethodInvocation. Upon return from this stored procedure with an @ErrorCode set to a value other than zero, this parameter is set to a value that MUST be ignored.

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors occurred.</td>
</tr>
<tr>
<td>-1</td>
<td>The DataClass of the specified Method already contains another MethodInvocation with the specified @Name.</td>
</tr>
<tr>
<td>-3</td>
<td>The specified Method already contains the implementation-specific maximum allowed number of MethodInstances.</td>
</tr>
<tr>
<td>-200</td>
<td>@Type equals &quot;1&quot; and Entity or DataClass of the Method already contains another MethodInvocation of type Finder.</td>
</tr>
<tr>
<td>-201</td>
<td>@Type equals &quot;2&quot; and Entity or DataClass of the Method already contains another MethodInvocation of type SpecificFinder.</td>
</tr>
<tr>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>-202</td>
<td>@Type equals &quot;5&quot; and Entity or DataClass of the Method already contains another MethodInstance of type IdEnumerator.</td>
</tr>
<tr>
<td>-203</td>
<td>MetadataObjectId of the Method containing the Parameter of the TypeDescriptor with MetadataObjectId equal to @ReturnTypeDescriptorId is not equal to @MethodId.</td>
</tr>
<tr>
<td>-204</td>
<td>The Parameter that contains the specified ReturnTypeDescriptor cannot have a Direction of &quot;1&quot;.</td>
</tr>
</tbody>
</table>

A positive integer A T-SQL error code.

**Return Code Values:** An integer that the protocol client MUST ignore.

**Results Sets:** MAY <15> return zero or more result sets that the protocol client MUST ignore.

### 3.1.5.14 proc_ar_CreateParameter

The proc_ar_CreateParameter stored procedure is called to create a Parameter contained by the Method identified by the specified MetadataObjectId.

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

```sql
PROCEDURE proc_ar_CreateParameter (  
    @Name                     nvarchar(255),  
    @IsCached                 bit,  
    @MethodId                 int,  
    @Direction                tinyint,  
    @TypeReflectorTypeName    nvarchar(255),  
    @CreatedId                int OUTPUT,  
    @ErrorCode                int OUTPUT
 );
```

**@Name:** The programmatic name of the Parameter. The value MUST be a Name, as specified in section 2.2.2.2.

**@IsCached:** A bit that specifies whether this Parameter is frequently used. The value MUST be an IsCached, as specified in section 2.2.2.3.

**@MethodId:** The MetadataObjectId of the Method that contains this Parameter. The value MUST be an Id, as specified in section 2.2.2.1.

**@Direction:** The direction in which this Parameter is passed to the LOBSystem while calling the Method. The value MUST be a Direction, as specified in section 2.2.2.16.

**@TypeReflectorTypeName:** The type name of the TypeReflector that is used to resolve the native type of this parameter. The value MUST be a TypeReflectorTypeName, as specified in section 2.2.2.17.

**@CreatedId:** The identifier for the newly created Parameter. Upon return from this stored procedure with an @ErrorCode set to zero, this parameter MUST be set to the MetadataObjectId of the newly created Parameter. Upon return from this stored procedure with an @ErrorCode set to a value other than zero, this parameter is set to a value that MUST be ignored.
**@ErrorCode:** The error code. Upon return from this stored procedure, this parameter MUST be set to an integer that is listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors occurred.</td>
</tr>
<tr>
<td>-1</td>
<td>The Method already has a Parameter with the specified @Name.</td>
</tr>
<tr>
<td>-3</td>
<td>The Method already contains the implementation-specific maximum allowed number of Parameters.</td>
</tr>
<tr>
<td>-100</td>
<td>@Direction is set to “4” and the Method already has a return Parameter.</td>
</tr>
</tbody>
</table>

A positive integer A T-SQL error code.

**Return Code Values:** An integer that the protocol client MUST ignore.

**Result Sets:** MAY return zero or more result sets that the protocol client MUST ignore.

### 3.1.5.15 proc_ar_CreateSystem

The **proc_ar_CreateSystem** stored procedure is called to create a **LobSystem** in the **ApplicationRegistry**. It MUST copy the list of ACEs associated with the **ApplicationRegistry** and associate them with the newly created **LobSystem**.

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

```sql
PROCEDURE proc_ar_CreateSystem(  
    @Name nvarchar(255),  
    @IsCached bit,  
    @SystemUtilityTypeName nvarchar(255),  
    @ConnectionManagerTypeName nvarchar(255),  
    @EntityInstanceTypeName nvarchar(255),  
    @CreatedId int OUTPUT,  
    @ErrorCode int OUTPUT  
);  
```

**@Name:** The name of the **LobSystem**. The value MUST be a Name, as specified in section 2.2.2.2.

**@IsCached:** A bit that specifies whether this **LobSystem** is frequently used. The value MUST be an IsCached, as specified in section 2.2.2.3.

**@SystemUtilityTypeName:** The name of the system utility that is used to execute the **Methods** in this **LobSystem**. The value MUST be a SystemUtilityTypeName, as specified in section 2.2.2.20.

**@ConnectionManagerTypeName:** The name of the connection manager that is used while connecting to this **LobSystem**. The value MUST be a ConnectionManagerTypeName, as specified in section 2.2.2.19.

**@EntityInstanceTypeName:** The name of the unit of implementation-specific business logic (2) that is used to create the objects that carry EntityInstance data to client applications. The value MUST be an EntityInstanceTypeName, as specified in section 2.2.2.21.

**@CreatedId:** The identifier for the newly created **LobSystem**. Upon return from this stored procedure with an @ErrorCode set to zero, this parameter MUST be set to the MetadataObjectId.
of the newly created **LobSystem**. Upon return from this stored procedure with an @**ErrorCode** set to a value other than zero, this parameter is set to a value that MUST be ignored.

**@ErrorCode:** The error code. Upon return from this stored procedure, this parameter MUST be set to an **integer** listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors occurred.</td>
</tr>
<tr>
<td>-1</td>
<td>The <strong>ApplicationRegistry</strong> already contains another <strong>LobSystem</strong> with the specified <strong>@Name</strong>.</td>
</tr>
<tr>
<td>A positive integer</td>
<td>A T-SQL error code.</td>
</tr>
</tbody>
</table>

**Return Code Values:** An **integer** that the protocol client MUST ignore.

**Result Sets:** MAY <19> return zero or more result sets that the protocol client MUST ignore.

### 3.1.5.16 proc_ar_CreateSystemInstance

The **proc_ar_CreateSystemInstance** stored procedure is called to create a **LobSystemInstance** in the specified **LobSystem**. It MUST copy the list of ACEs associated with the **LobSystem** and associate them with the newly created **LobSystemInstance**.

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

```sql
PROCEDURE proc_ar_CreateSystemInstance ( 
    @Name                      nvarchar(255),
    @IsCached                  bit,
    @SystemId                  int,
    @CreatedId                 int OUTPUT,
    @ErrorCode                 int OUTPUT
); 
```

**@Name:** The name of the **LobSystemInstance**. The value MUST be a **Name**, as specified in section 2.2.2.2.

**@IsCached:** A bit that specifies whether this **LobSystemInstance** is frequently used. The value MUST be an **IsCached**, as specified in section 2.2.2.3.

**@SystemId:** The **MetadataObjectId** of the **LobSystem** that contains this **LobSystemInstance**. The value MUST be an **Id**, as specified in section 2.2.2.1.

**@CreatedId:** The identifier for the newly created **LobSystemInstance**. Upon return from this stored procedure with an @**ErrorCode** set to zero, this parameter MUST be set to the **MetadataObjectId** of the newly created **LobSystemInstance**. Upon return from this stored procedure with an @**ErrorCode** set to a value other than zero, this parameter is set to a value that MUST be ignored.

**@ErrorCode:** The error code. Upon return from this stored procedure, this parameter MUST be set to an **integer** listed in the following table.
### Return Code Values:
An integer that the protocol client MUST ignore.

### Results Sets:
MAY `<20>` return zero or more result sets that the protocol client MUST ignore.

#### 3.1.5.17 proc_ar_CreateTypeDescriptor

The `proc_ar_CreateTypeDescriptor` stored procedure is called to create a `TypeDescriptor`.

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

```sql
PROCEDURE proc_ar_CreateTypeDescriptor (  
    @Name                      nvarchar(255),
    @IsCached                  bit,
    @ParameterId               int,
    @ParentTypeDescriptorId    int,
    @TypeName                  nvarchar(255),
    @IdentifierId              int,
    @FilterDescriptorId        int,
    @IsCollection              bit,
    @CreatedId                 int OUTPUT,
    @ErrorCode                 int OUTPUT
);
```

- **@Name**: The name of the `TypeDescriptor`. The value MUST be a `Name`, as specified in section 2.2.2.2.

- **@IsCached**: A bit that specifies whether this `TypeDescriptor` is frequently used. The value MUST be an `IsCached`, as specified in section 2.2.2.3.

- **@ParameterId**: The `MetadataObjectId` of the `Parameter` that contains this `TypeDescriptor`. The value MUST be an `Id`, as specified in section 2.2.2.1.

- **@ParentTypeDescriptorId**: The `MetadataObjectId` of the parent `TypeDescriptor` that contains this `TypeDescriptor`. The value MUST be an `Id`, as specified in section 2.2.2.1.

- **@TypeName**: The programmatic name of the data type that is represented by this `TypeDescriptor`. The value MUST be a `TypeDescriptorTypeName`, as specified in section 2.2.2.18.

- **@IdentifierId**: The `MetadataObjectId` of the `Identifier` referenced by this `TypeDescriptor`. The value MUST be an `Id`.

- **@FilterDescriptorId**: The `MetadataObjectId` of the `FilterDescriptor` associated with this `TypeDescriptor`. The value MUST be an `Id`.

---

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors occurred.</td>
</tr>
<tr>
<td>-1</td>
<td>A <code>LobSystemInstance</code> with <code>MetadataObject</code> name equal to <code>@Name</code> already exists.</td>
</tr>
<tr>
<td>-3</td>
<td>The <code>LobSystem</code> already contains the implementation-specific maximum allowed number of <code>LobSystemInstances</code>.</td>
</tr>
</tbody>
</table>

A positive integer A T-SQL error code.
@IsCollection: A bit that specifies whether this TypeDescriptor is to be interpreted by protocol clients as a collection of native LOB System data structures. The value MUST be an IsCollection, as specified in section 2.2.2.8.

@CreatedId: The identifier for the newly created TypeDescriptor. Upon return from this stored procedure with an ErrorCode set to zero, this parameter MUST be set to the MetadataObjectId of the newly created TypeDescriptor. Upon return from this stored procedure with an ErrorCode set to a value other than zero, this parameter is set to a value that MUST be ignored.

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors occurred.</td>
</tr>
<tr>
<td>-1</td>
<td>The TypeDescriptor identified by @ParentTypeDescriptor already contains a child TypeDescriptor with the specified @Name.</td>
</tr>
</tbody>
</table>
| -3    | At least one of the following two statements is true:  
  - @ParentTypeDescriptorId is not NULL and a TypeDescriptor with MetadataObjectId equal to @ParentTypeDescriptorId already contains the implementation-specific maximum allowed number of TypeDescriptors.  
  - @FilterDescriptorId is not NULL and a FilterDescriptor with MetadataObjectId equal to @FilterDescriptorId already contains the implementation-specific maximum allowed number of associated TypeDescriptors. |
| -302  | The @ParentTypeDescriptorId is equal to NULL and Parameter already has a root TypeDescriptor. |
| -303  | The Parameter and FilterDescriptor with MetadataObjectId equal to @FilterDescriptorId don't belong to the same Method. |
| -305  | @IsCollection equals "1" and the data type represented by the TypeDescriptor with MetadataObjectId equal to @ParentTypeDescriptorId is interpreted as a collection. A child TypeDescriptor of a parent TypeDescriptor that represents a collection MUST NOT be a collection. |
| -306  | The data type specified by @ParentTypeDescriptorId is interpreted as a collection and already contains a child TypeDescriptor. A TypeDescriptor that represents a collection MUST NOT have more than one child. |

A positive integer A T-SQL error code.

Return Code Values: An integer that the protocol client MUST ignore.

Results Sets: MAY <21> return zero or more result sets that the protocol client MUST ignore.

3.1.5.18 proc_ar_DeleteActionById

The proc_ar_DeleteActionById stored procedure is called to delete the specified Action along with its properties, localized names, ACEs and ActionParameters.

The T-SQL syntax for the stored procedure is as follows:
PROCEDURE proc_ar_DeleteActionById(
    @Id                   int,
    @Version              int,
    @ErrorCode            int OUTPUT
);

@Id: The MetadataObjectId of the Action that is deleted. The value MUST be an Id, as specified in section 2.2.2.1.

@Version: The object version of this Action at the time it was last read.

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer that is listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-2</td>
<td>An Action with the specified MetadataObjectId does not exist.</td>
</tr>
<tr>
<td>-6</td>
<td>An Action with the specified MetadataObjectId has been updated by a context other than the one that it has been currently read by. This happens when the version specified does not match with the current version of the Action.</td>
</tr>
<tr>
<td>A positive integer</td>
<td>A T-SQL error code.</td>
</tr>
</tbody>
</table>

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MAY <22> return zero or more result sets that the protocol client MUST ignore.

3.1.5.19 proc_ar_DeleteActionParameterById

The proc_ar_DeleteActionParameterById stored procedure is called to delete the ActionParameter identified by its given MetadataObjectId along with its Properties, localized names and access ACEs.

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

PROCEDURE proc_ar_DeleteActionParameterById (
    @Id                   int,
    @Version              int,
    @ErrorCode            int OUTPUT
);

@Id: The identifier for the ActionParameter that is to be deleted. The value MUST be an Id, as specified in section 2.2.2.1.

@Version: The object version of this ActionParameter at the time it was last read.

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.
<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-2</td>
<td>An ActionParameter with MetadataObjectId equal to @Id doesn't exist.</td>
</tr>
<tr>
<td>-6</td>
<td>An ActionParameter with MetadataObjectId equal to @Id has been updated by a context other than the one that it has been currently read by. This happens when the value of @Version does not match with the version for the ActionParameter.</td>
</tr>
<tr>
<td></td>
<td>A positive integer</td>
</tr>
</tbody>
</table>

**Return Code Values:** An integer that the protocol client MUST ignore.

**Results Sets:** MAY <23> return zero or more result sets that the protocol client MUST ignore.

### 3.1.5.20 proc_ar_DeleteAssociationById

The proc_ar_DeleteAssociationById stored procedure is called to delete the Association identified by its given MetadataObjectId along with its properties, localized names and ACEs.

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

```sql
PROCEDURE proc_ar_DeleteAssociationById (  
    @Id                   int,  
    @Version              int,  
    @ErrorCode            int OUTPUT  
);
```

**@Id:** The identifier for the Association that is to be deleted. The value MUST be an Id, as specified in section 2.2.2.1.

**@Version:** The object version of this Association at the time it was last read.

**@ErrorCode:** The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-2</td>
<td>An Association with MetadataObjectId equal to @Id doesn't exist.</td>
</tr>
<tr>
<td>-6</td>
<td>An Association with MetadataObjectId equal to @Id has been updated by a context other than the one that it has been currently read by. This happens when the value of @Version does not match with the version of the Association.</td>
</tr>
<tr>
<td></td>
<td>A positive integer</td>
</tr>
</tbody>
</table>

**Return Code Values:** An integer that the protocol client MUST ignore.

**Results Sets:** MAY <24> return zero or more result sets that the protocol client MUST ignore.
3.1.5.21  proc_ar_DeleteDefaultValue

The proc_ar_DeleteDefaultValue stored procedure is called to delete the DefaultValue identified by the specified TypeDescriptor and MethodInstance.

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

```
PROCEDURE proc_ar_DeleteDefaultValue (  
    @TypeDescriptorId    int,  
    @MethodInstanceId   int,  
    @ErrorCode           int OUTPUT
);  
```

@TypeDescriptorId: The MetadataObjectId of the TypeDescriptor associated with the DefaultValue. The value MUST be an Id, as specified in section 2.2.2.1.

@MethodInstanceId: The MetadataObjectId of the MethodInstance associated with the DefaultValue. The value MUST be an Id.

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-2</td>
<td>At least one of the following two statements is true:</td>
</tr>
<tr>
<td></td>
<td>- A TypeDescriptor with MetadataObjectId equal to @TypeDescriptorId doesn't exist.</td>
</tr>
<tr>
<td></td>
<td>- A MethodInstance with MetadataObjectId equal to @MethodInstanceId doesn't exist.</td>
</tr>
<tr>
<td>A positive integer</td>
<td>A T-SQL error code.</td>
</tr>
</tbody>
</table>

Return Code Values: An integer that the protocol client MUST ignore.

Results Sets: MAY <25> return zero or more result sets that the protocol client MUST ignore.

3.1.5.22  proc_ar_DeleteEntityById

The proc_ar_DeleteEntityById stored procedure is called to delete the Entity identified by the specified MetadataObjectId along with its properties, localized names, and ACEs.

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

```
PROCEDURE proc_ar_DeleteEntityById (  
    @Id       int,  
    @Version  int,  
    @ErrorCode int OUTPUT
);  
```
@Id: The MetadataObjectId of the Entity that is updated. The value MUST be an Id, as specified in section 2.2.2.1.

@Version: The object version of this Entity at the time it was last read.

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer that is listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-2</td>
<td>An Entity with the specified @Id does not exist.</td>
</tr>
</tbody>
</table>
| -5    | An Entity with the specified @Id is referenced as a source or destination in any Association, or contains at least one of the following child objects:  
  ▪ Action  
  ▪ Method  
  ▪ Identifier |
| -6    | An Entity with the specified @Id has been updated by a context other than the one that it has been currently read by. This happens when the version specified does not match with the current version of the Entity. |

A positive integer A T-SQL error code.

Return Code Values: An integer that the protocol client MUST ignore.

Results Sets: MAY <26> return zero or more result sets that the protocol client MUST ignore.

3.1.5.23 proc_ar_DeleteFilterDescriptorById

The proc_ar_DeleteFilterDescriptorById stored procedure is called to delete the FilterDescriptor identified by the specified MetadataObjectId along with its properties, localized names, and ACEs.

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

```sql
PROCEDURE proc_ar_DeleteFilterDescriptorById (  
  @Id int,  
  @Version int,  
  @ErrorCode int OUTPUT
);  
```

@Id: The MetadataObjectId of the FilterDescriptor that is deleted. The value MUST be an Id, as specified in section 2.2.2.1.

@Version: The object version of this FilterDescriptor at the time it was last read.

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.
<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-2</td>
<td>An FilterDescriptor with the specified MetadataObjectId does not exist.</td>
</tr>
<tr>
<td>-6</td>
<td>An FilterDescriptor with the specified MetadataObjectId has been updated by a context other than the one that it has been currently read by. This happens when the version specified does not match with the current version of the FilterDescriptor.</td>
</tr>
</tbody>
</table>

**Return Code Values:** An integer that the protocol client MUST ignore.

**Result Sets:** MAY <27> return zero or more result sets that the protocol client MUST ignore.

### 3.1.5.24 proc_ar_DeleteIdentifierById

The proc_ar_DeleteIdentifierById stored procedure is called to delete the Identifier identified by the specified MetadataObjectId along with its properties, localized names, and ACEs. After a successful deletion, the ordinal number attribute of all Identifiers MUST be normalized for Identifiers that are contained by the same Entity that contained the deleted Identifier. After normalization, the ordinal number of all these Identifiers MUST be renumbered starting from zero, incrementing by 1, and preserving the original ordering. During this renumbering, the version attribute of all these Identifiers SHOULD <28> be updated.

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

```sql
PROCEDURE proc_ar_DeleteIdentifierById (  
    @Id                  int,  
    @Version             int,  
    @ErrorCode           int OUTPUT
);
```

@Id: The MetadataObjectId of the Identifier to be deleted. The value MUST be an Id, as specified in section 2.2.2.1.

@Version: The object version of this Identifier at the time it was last read.

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer that is listed in the following table.
**Return Code Values:** An integer that the protocol client MUST ignore.

**Result Sets:** MAY<29> return zero or more result sets that the protocol client MUST ignore.

### 3.1.5.25 proc_ar_DeleteLocalizedNameForMetadataObjectByLCID

The proc_ar_DeleteLocalizedNameForMetadataObjectByLCID stored procedure is called to delete a localized name contained by the MetadataObject for a given LCID.

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

```sql
PROCEDURE proc_ar_DeleteLocalizedNameForMetadataObjectByLCID (
    @MetadataObjectId        int,
    @LCID                    int,
    @ErrorCode               int OUTPUT
);
```

**@MetadataObjectId:** The MetadataObjectId of the MetadataObject that contains the localized name to be deleted. The value MUST be an Id, as specified in section 2.2.2.1.

**@LCID:** The LCID of the localized name to be deleted.

**@ErrorCode:** The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-2</td>
<td>A localized name for the given @LCID doesn’t exist or exists more than once for the MetadataObject with MetadataObjectId equal to @MetadataObjectId.</td>
</tr>
</tbody>
</table>

**Return Code Values:** An integer that the protocol client MUST ignore.

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

### 3.1.5.26 proc_ar_DeleteLocalizedNamesByMetadataObjectId

The proc_ar_DeleteLocalizedNamesByMetadataObjectId stored procedure is called to delete all localized names of the MetadataObject identified by its specified MetadataObjectId.

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

```sql
PROCEDURE proc_ar_DeleteLocalizedNamesByMetadataObjectId (
    @MetadataObjectId      int,
    @ErrorCode             int OUTPUT
);
```

**@MetadataObjectId:** The MetadataObjectId of the MetadataObject that owns the localized names to be deleted.

**@ErrorCode:** The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.
### 3.1.5.27 proc_ar_DeleteMethodById

The `proc_ar_DeleteMethodById` stored procedure is called to delete the Method identified by the specified `MetadataObjectId` along with its properties, localized names, and ACEs.

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

```sql
PROCEDURE proc_ar_DeleteMethodById ( 
    @Id                   int, 
    @Version              int, 
    @ErrorCode            int OUTPUT );
```

@Id: The `MetadataObjectId` of the Method to be deleted. The value MUST be an Id, as specified in section 2.2.2.1.

@Version: The object version of this Method at the time it was last read.

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer that is listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-2</td>
<td>A Method with the specified <code>MetadataObjectId</code> does not exist.</td>
</tr>
</tbody>
</table>
| -5    | A Method with the specified `MetadataObjectId` contains at least one of the following child objects:  
  - FilterDescriptor  
  - MethodInstance  
  - Parameter |
| -6    | A Method with the specified `MetadataObjectId` has been updated by a context other than the one that it has been currently read by. This happens when the version specified does not match with the current version of the Method. |

A positive integer A T-SQL error code.

**Return Code Values:** An integer that the protocol client MUST ignore.

**Results Sets:** MUST NOT return any result sets.
3.1.5.28 proc_ar_DeleteMethodInstanceById

The proc_ar_DeleteMethodInstanceById stored procedure is called to delete the MethodInstance identified by the specified MetadataObjectId along with its properties, localized names, and ACEs. It MUST also delete any DefaultValues associated with the MethodInstance identified by the specified MetadataObjectId.

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

```sql
PROCEDURE proc_ar_DeleteMethodInstanceById (
    @Id                  int,
    @Version             int,
    @ErrorCode           int OUTPUT
);```

@Id: The MetadataObjectId of the MethodInstance to be deleted. The value MUST be an Id, as specified in section 2.2.2.1.

@Version: The object version of this MethodInstance at the time it was last read.

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer that is listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-2</td>
<td>A MethodInstance with the specified MetadataObjectId does not exist.</td>
</tr>
<tr>
<td>-6</td>
<td>A MethodInstance with the specified MetadataObjectId has been updated by a context other than the one that it has been currently read by. This happens when the version specified does not match with the current version of the MethodInstance.</td>
</tr>
</tbody>
</table>

A positive integer A T-SQL error code.

Return Code Values: An integer that the protocol client MUST ignore.

Results Sets: MAY return zero or more result sets that the protocol client MUST ignore.

3.1.5.29 proc_ar_DeleteParameterById

The proc_ar_DeleteParameterById stored procedure is called to delete the Parameter identified by its given MetadataObjectId along with its properties, localized names, and ACEs. After a successful deletion, the ordinal number attribute of all Parameters MUST be normalized for Parameters that are contained by the same Method that contained the deleted Parameter. After normalization, the ordinal number of all these Parameters MUST be renumbered starting from zero, incrementing by 1, and preserving the original ordering. During this renumbering, the version attribute of all these Parameters SHOULD be updated.

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

```sql
PROCEDURE proc_ar_DeleteParameterById (
    @Id                     int,
    @Version                int,
    @ErrorCode              int OUTPUT
);```
@Id: The MetadataObjectId of the Parameter to be deleted. The value MUST be an Id, as specified in section 2.2.2.1.

@Version: The object version of this Parameter at the time it was last read.

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-2</td>
<td>A Parameter with MetadataObjectId equal to @Id doesn't exist.</td>
</tr>
<tr>
<td>-5</td>
<td>One or more TypeDescriptors with ParameterId equal to @Id already exist.</td>
</tr>
<tr>
<td>-6</td>
<td>A Parameter with MetadataObjectId equal to @Id has been updated by a context other than the one that it has been currently read by. This happens when the value of @Version does not match the version for the Parameter.</td>
</tr>
<tr>
<td>A positive integer</td>
<td>A T-SQL error.</td>
</tr>
</tbody>
</table>

Return Code Values: An integer that the protocol client MUST ignore.

Results Sets: MAY return zero or more result sets that the protocol client MUST ignore.

3.1.5.30 proc_ar_DeletePropertiesById

The proc_ar_DeletePropertiesById stored procedure is called to delete all properties contained by the MetadataObject identified by its given MetadataObjectId.

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

```sql
PROCEDURE proc_ar_DeletePropertiesById (  @MetadataObjectId int,  @ErrorCode int OUTPUT
);  
```

@MetadataObjectId: The MetadataObjectId of the MetadataObject that contains the properties to be deleted. The value MUST be an Id, as specified in section 2.2.2.1.

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>A positive integer</td>
<td>A T-SQL error.</td>
</tr>
</tbody>
</table>

Return Code Values: An integer that the protocol client MUST ignore.
**Results Sets:** MUST NOT return any result sets.

### 3.1.5.31 proc_ar_DeletePropertyForMetadataObjectId

The `proc_ar_DeletePropertyForMetadataObjectId` stored procedure is called to delete the Property with a given programmatic name and contained by a MetadataObject identified by its given MetadataObjectId.

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

```sql
PROCEDURE proc_ar_DeletePropertyForMetadataObjectId (  
    @MetadataObjectId        int,  
    @Name                    nvarchar(255),  
    @ErrorCode               int OUTPUT
);
```

**@MetadataObjectId:** The MetadataObjectId of the MetadataObject that contains the Property to be deleted. The value MUST be an Id, as specified in section 2.2.2.1.

**@Name:** The name of the Property. The value MUST be a Name, as specified in section 2.2.2.

**@ErrorCode:** The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-2</td>
<td>A Property with name equal to @Name doesn’t exist, or exists more than once for the MetadataObject with a MetadataObjectId equal to @MetadataObjectId.</td>
</tr>
</tbody>
</table>

**Return Code Values:** An integer that the protocol client MUST ignore.

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

### 3.1.5.32 proc_ar_DeleteSystemById

The `proc_ar_DeleteSystemById` stored procedure is called to delete the LobSystem identified by the specified MetadataObjectId along with its properties, localized names, and ACEs.

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

```sql
PROCEDURE proc_ar_DeleteSystemById (  
    @Id                     int,  
    @Version                int,  
    @ErrorCode              int OUTPUT
);
```

**@Id:** The MetadataObjectId of the LobSystem to be deleted. The value MUST be an Id, as specified in section 2.2.1.

**@Version:** The object version of this LobSystem at the time it was last read.

**@ErrorCode:** The object version of this LobSystem at the time it was last read. The error code. Upon return from this stored procedure, this parameter MUST be set to an integer that is listed in the following table.

**Results Sets:** MUST NOT return any result sets.
<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-2</td>
<td>An LobSystem with the specified MetadataObjectId does not exist.</td>
</tr>
</tbody>
</table>
| -5    | An LobSystem with the specified MetadataObjectId contains at least one of the following child objects:  
  - DataClasses  
  - LobSystemInstances |
| -6    | An LobSystem with the specified MetadataObjectId has been updated by a context other than the one that it has been currently read by. This happens when the version specified does not match with the current version of the LobSystem. |

A positive integer: A T-SQL error code.

**Return Code Values:** An integer that the protocol client MUST ignore.

**Results Sets:** MAY return zero or more result sets that the protocol client MUST ignore.

### 3.1.5.33 proc_ar_DeleteSystemInstanceById

The proc_ar_DeleteSystemInstanceById stored procedure is called to delete the LobSystemInstance identified by the specific MetadataObjectId along with its ACEs.

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

```sql
PROCEDURE proc_ar_DeleteSystemInstanceById (  
    @Id                        int,  
    @Version                   int,  
    @ErrorCode                 int OUTPUT
 );
```

@Id: The MetadataObjectId of the LobSystemInstance to be deleted. The value MUST be an Id, as specified in section 2.2.2.1.

@Version: The object version of this LobSystemInstance at the time it was last read.

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer that is listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-2</td>
<td>An LobSystemInstance with the specified MetadataObjectId does not exist.</td>
</tr>
<tr>
<td>-6</td>
<td>An LobSystemInstance with the specified MetadataObjectId has been updated by a context other than the one that it has been currently read by. This happens when the version specified does not match with the current version of the LobSystemInstance.</td>
</tr>
</tbody>
</table>

A positive integer: A T-SQL error code.
Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MAY<35> return zero or more result sets that the protocol client MUST ignore.

3.1.5.34 proc_ar_DeleteTypeDescriptorById

The proc_ar_DeleteTypeDescriptorById stored procedure is called to delete the TypeDescriptor identified by the specified MetadataObjectId along with its properties, localized names, ACEs and all its child TypeDescriptors, recursively.

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

    PROCEDURE proc_ar_DeleteTypeDescriptorById (  
      @Id                         int,  
      @Version                    int,  
      @ErrorCode                  int OUTPUT  
    );

@Id: The MetadataObjectId of the TypeDescriptor to be deleted. The value MUST be an Id, as specified in section 2.2.2.1.

@Version: The object version of this TypeDescriptor at the time it was last read.

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer that is listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-2</td>
<td>A TypeDescriptor with the specified MetadataObjectId does not exist.</td>
</tr>
<tr>
<td>-5</td>
<td>A MethodInstance refers to the TypeDescriptor identified by the specified MetadataObjectId as its ReturnTypeDescriptor.</td>
</tr>
<tr>
<td>-6</td>
<td>A TypeDescriptor with the specified MetadataObjectId has been updated by a context other than the one that it has been currently read by. This happens when the version specified does not match with the current version of the TypeDescriptor.</td>
</tr>
</tbody>
</table>

A positive integer A T-SQL error code.

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MAY<36> return zero or more result sets that the protocol client MUST ignore.

3.1.5.35 proc_ar_EnsureApplicationRegistryExists

The proc_ar_EnsureApplicationRegistryExists stored procedure is called to verify that an ApplicationRegistry exists. When not found, the stored procedure MUST create a new ApplicationRegistry.
The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_ar_EnsureApplicationRegistryExists();
```

**Return Code Values:** An integer that the protocol client MUST ignore.

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

### 3.1.5.36 proc_ar_GetAccessControlEntriesForMetadataObject

The `proc_ar_GetAccessControlEntriesForMetadataObject` stored procedure is called to retrieve ACEs for a MetadataObject with the specified MetadataObjectId.

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

```sql
PROCEDURE proc_ar_GetAccessControlEntriesForMetadataObject (  
    @MetadataObjectId          int,  
    @ErrorCode                 int OUTPUT  
);  
```

@MetadataObjectId: The MetadataObjectId of the MetadataObject.

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-2</td>
<td>Object not found.</td>
</tr>
</tbody>
</table>

**Return Code Values:** An integer that the protocol client MUST ignore.

**Result Sets:** If @ErrorCode is set to "-2", this stored procedure MUST NOT return any result sets. Otherwise, this stored procedure MUST return the Access Control Entry result set, as specified in section 3.1.5.36.1.

#### 3.1.5.36.1 Access Control Entry Result Set

The Access Control Entry result det returns information about ACEs that authorize what may be done with the MetadataObject it is associated with. Each row in the result set contains all the attributes of a single ACE. The result set MUST have zero or more rows.

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

```sql
MetadataObjectId       int,  
IdentityName           nvarchar(255),  
DisplayName            nvarchar(255),  
RawSid                 varbinary(512),  
Rights                 bigint;
```

MetadataObjectId: The MetadataObjectId of the MetadataObject that the ACE is associated with.
**IdentityName**: The programmatic name of the security principal (2).

**DisplayName**: The name of the security principal (2) used for display purposes.

**RawSid**: The SID, if the security principal (2) is a Windows security principal. If the security principal (2) is not a windows security principal, the value MUST be NULL.

**Rights**: The permissions available to the security principal (2) for the MetadataObject identified by the MetadataObjectId. It MUST be a MetadataRights, as specified in section 2.2.2.22.

### 3.1.5.37 proc_ar_GetActionById

The proc_ar_GetActionById stored procedure is called to retrieve Action information for the Action with a specified MetadataObjectId.

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

```
PROCEDURE proc_ar_GetActionById (  
   @MetadataObjectId          int
);

@MetadataObjectId: The MetadataObjectId of the Action. The value MUST be an Id, as specified in section 2.2.2.1.

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MUST return the Action result set, as specified in section 3.1.5.37.1.

#### 3.1.5.37.1 Action Result Set

See section 2.2.5.1. The result set MUST contain zero or one row.

### 3.1.5.38 proc_ar_GetActionParameterById

The proc_ar_GetActionParameterById stored procedure is called to retrieve ActionParameter data for the ActionParameter with a specified MetadataObjectId.

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

```
PROCEDURE proc_ar_GetActionParameterById (  
   @MetadataObjectId               int
);

@MetadataObjectId: The MetadataObjectId of the ActionParameter.

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MUST return the ActionParameter result set, as specified in section 3.1.5.38.1.

#### 3.1.5.38.1 ActionParameter Result Set

See section 2.2.5.2. The result set MUST contain zero or one row.
3.1.5.39  proc_ar_GetActionParametersForActionWithCount

The `proc_ar_GetActionParametersForActionWithCount` stored procedure is called to retrieve the `ActionParameters` contained by the `Action` with the specified `MetadataObjectId`.

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

```sql
PROCEDURE proc_ar_GetActionParametersForActionWithCount (  
    @ActionId                    int  
) ;
```

@ActionId: The `MetadataObjectId` of the `Action`. The value MUST be an `Id`, as specified in section 2.2.2.1.

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MUST return the `Count` result set, as specified in section 3.1.5.39.1, and the `ActionParameter` result set, as specified in section 3.1.5.39.2.

3.1.5.39.1  Count Result Set

See section 2.2.5.4. The result set MUST contain one row.

3.1.5.39.2  ActionParameter Result Set

See section 2.2.5.2. The result set MUST contain zero or more rows.

3.1.5.40  proc_ar_GetActionsForEntityWithCount

The `proc_ar_GetActionsForEntityWithCount` stored procedure is called to retrieve the `Actions` contained by the `Entity` with the specified `MetadataObjectId`.

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

```sql
PROCEDURE proc_ar_GetActionsForEntityWithCount (  
    @EntityId                    int  
) ;
```

@EntityId: The `MetadataObjectId` of the `Entity`. The value MUST be an `Id`, as specified in section 2.2.2.1.

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MUST return a `Count` result set, as specified in section 3.1.5.40.1, and an `Action` result set, as specified in section 3.1.5.40.2.

3.1.5.40.1  Count Result Set

See section 2.2.5.4. The result set MUST contain one row.

3.1.5.40.2  Action Result Set

See section 2.2.5.1. The result set MUST contain zero or more rows.
3.1.5.41 proc_ar_GetAllLocalizedNamesForMetadataObjectWithCount

The proc_ar_GetAllLocalizedNamesForMetadataObjectWithCount stored procedure is called to retrieve all localized names of a MetadataObject identified by the specified MetadataObjectId.

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

```sql
PROCEDURE [proc_ar_GetAllLocalizedNamesForMetadataObjectWithCount] (  
   @MetadataObjectId                 int
);
```

- **@MetadataObjectId**: The Id of the MetadataObject. The value MUST be an Id, as specified in section 2.2.2.1.

**Return Code Values:** An integer that the protocol client MUST ignore.

**Result Sets:** MUST return the Count result set, as specified in section 3.1.5.41.1, and the Localized Name result set, as specified in section 3.1.5.41.2.

3.1.5.41.1 Count Result Set

See section 2.2.5.4. The result set MUST contain one row.

3.1.5.41.2 Localized Name Result Set

The Localized Name result set contains localized names. Each row in the result set contains a single localized name of a MetadataObject in a specific locale.

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

```sql
Id    int,
LCID   int,
[Localized name]   nvarchar(255),
MetadataObjectId   int;
```

- **Id**: The MetadataObjectId of the localized name that is returned. The value MUST be an Id, as specified in section 2.2.2.1.

- **LCID**: The LCID corresponding to the returned localized name.

- **Localized name**: The localized name of the specified MetadataObject corresponding to the returned LCID.

- **MetadataObjectId**: The MetadataObjectId of the MetadataObject. The value MUST be an Id.

The result set MUST contain zero or more rows.

3.1.5.42 proc_ar_GetAllSystemInstancesLikeNameWithCount

The proc_ar_GetAllSystemInstancesLikeNameWithCount stored procedure is called to retrieve the count and the LobSystemInstances that satisfy either one of the following constraints:

- LobSystemInstances whose name matches the pattern given by @SystemInstanceName.
• **LobSystemInstances** whose localized names matches the pattern given by @SystemInstanceName, and for which the **LCID** of the same localized name is either equal to the value given by @LCID or is equal to zero.

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

```sql
PROCEDURE [proc_ar_GetAllSystemInstancesLikeNameWithCount] (  
    @SystemInstanceName nvarchar(255),  
    @LCID int  
);  
```

@SystemInstanceName: The **string** that specifies the name pattern of the **LobSystemInstances** to be returned. The characters in this **string** MUST be in uppercase. It can include wildcard characters. For example, if the **MetadataObjectName** is set to "A%", this stored procedure returns only the **LobSystemInstances** with names beginning with either "A" or "a".

@LCID: The **LCID** used to restrict which localized names of the **LobSystemInstances** to consider.

**Return Code Values:** An **integer** that the protocol client MUST ignore.

**Result Sets:** MUST return the Count result set, as specified in section 3.1.5.42.1, and the System Instance result set, as specified in section 3.1.5.42.2.

3.1.5.42.1 Count Result Set

See section 2.2.5.4. The result set MUST return one row.

3.1.5.42.2 System Instance Result Set

See section 2.2.5.12. The result set MUST return zero or more rows.

3.1.5.43 proc_ar_GetAllSystemInstancesWithCount

The **proc_ar_GetAllSystemInstancesWithCount** stored procedure is called to retrieve all **LobSystemInstances**, along with the count of such **LobSystemInstances**.

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

```sql
PROCEDURE proc_ar_GetAllSystemInstancesWithCount();
```

**Return Code Values:** An **integer** that the protocol client MUST ignore.

**Result Sets:** MUST return two result sets in the following order:

1. **Count** result set, as specified in section 3.1.5.43.1.

2. **System Instance** result set, as specified in section 3.1.5.43.2.

3.1.5.43.1 Count Result Set

See section 2.2.5.4. The result set MUST contain one row.
3.1.5.43.2 System Instance Result Set
See section 2.2.5.12. The result set MUST contain zero or more rows.

3.1.5.44 proc_ar_GetAllSystemsWithCount

The proc_ar_GetAllSystemsWithCount stored procedure is called to retrieve the count and details of all LobSystems.
The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_ar_GetAllSystemsWithCount();
```

**Return Code Values:** An integer that the protocol client MUST ignore.

**Result Sets:** MUST return the Count result set, as specified in section 3.1.5.44.1, and the System result set, as specified in section 3.1.5.44.2.

3.1.5.44.1 Count Result Set
See section 2.2.5.4. The result set MUST contain one row.

3.1.5.44.2 System Result Set
See section 2.2.5.13. The result set MUST contain zero or more rows.

3.1.5.45 proc_ar_GetAssociationById

The proc_ar_GetAssociationById stored procedure is called to retrieve the Association identified by the specified MetadataObjectId.
The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_ar_GetAssociationById (@MetadataObjectId int);
```

**@MetadataObjectId:** The MetadataObjectId of the Association to be retrieved. The value MUST be an Id, as specified in section 2.2.2.1.

**Return Code Values:** An integer that the protocol client MUST ignore.

**Result Set:** MUST return an Association result set, as specified in section 3.1.5.45.1.

3.1.5.45.1 Association Result Set
See section 2.2.5.3. The result set MUST contain zero or one row.

3.1.5.46 proc_ar_GetAssociationByName

The proc_ar_GetAssociationByName stored procedure is called to retrieve the Association with the specified name contained by the specified LobSystem identified by the specified MetadataObjectId.
The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_ar_GetAssociationByName (
    @AssociationName nvarchar(255),
    @SystemId int
);
```

@AssociationName: The programmatic name of the Association to be retrieved. The value MUST be a Name, as specified in section 2.2.2.

@SystemId: The MetadataObjectId of the LobSystem that contains this Association. The value MUST be an Id, as specified in section 2.2.2.1.

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MUST return the Association result set, as specified in section 3.1.5.46.1.

3.1.5.46.1 Association Result Set

See section 2.2.5.3. The result set MUST contain zero or one row.

3.1.5.47 proc_ar_GetAssociationsForDataClassWithCount

The proc_ar_GetAssociationsForDataClassWithCount stored procedure is invoked to retrieve the count and details of all Associations contained by all Methods contained by the specified DataClass. The stored procedure MUST return all Associations for all Methods of a DataClass, but no MethodInstances that are not Associations.

The T-SQL syntax is as follows:

```sql
PROCEDURE proc_ar_GetAssociationsForDataClassWithCount (
    @ClassId int
);
```

@ClassId: the MetadataObjectId for the DataClass.

Return Code Values: An integer that the protocol client MUST be ignore.

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

1. A Count result set, as specified in section 3.1.5.47.1.
2. An Association result set, as specified in section 3.1.5.47.2

3.1.5.47.1 Count Result Set

See section 2.2.5.4. The result set MUST contain one row.

3.1.5.47.2 Association Result Set

See section 2.2.5.3. The result set MUST contain zero or more rows.
3.1.5.48 proc_ar_GetAssociationsForEntityAndRoleWithCount

The proc_ar_GetAssociationsForEntityAndRoleWithCount stored procedure is called to retrieve the Associations that reference the specified Entity as an Association source or destination.

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

```sql
PROCEDURE proc_ar_GetAssociationsForEntityAndRoleWithCount (
    @EntityId                 int,
    @EntityRole               bit
);
```

@EntityId: The MetadataObjectId of the Entity. The value MUST be an Id, as specified in section 2.2.2.1.

@EntityRole: A bit that specifies whether @EntityId represents a source or a destination Entity. The value of this parameter MUST be listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Source Entity.</td>
</tr>
<tr>
<td>1</td>
<td>Destination Entity.</td>
</tr>
</tbody>
</table>

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MUST return two result sets in the following order:
1. A Count result set, as specified in section 3.1.5.48.1.
2. An Association result set, as specified in section 3.1.5.48.2.

3.1.5.48.1 Count Result Set

See section 2.2.4. The result set MUST contain one row.

3.1.5.48.2 Association Result Set

See section 2.2.5. The result set MUST contain zero or more rows.

3.1.5.49 proc_ar_GetAssociationsForMethodWithCount

The proc_ar_GetAssociationsForMethodWithCount stored procedure is called to retrieve the count and details of all Associations contained by the Method. MethodInstances that are not Associations MUST NOT be returned.

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

```sql
PROCEDURE proc_ar_GetAssociationsForMethodWithCount (  
    @MethodId                int
);
```
@MethodId: The MetadataObjectId of the Method object. The value MUST be an Id, as specified in section 2.2.2.1.

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MUST return a Count result set, as specified in section 3.1.5.49.1, and an Association result set, as specified in section 3.1.5.49.2.

3.1.5.49.1 Count Result Set
See section 2.2.5.4. The result set MUST contain one row.

3.1.5.49.2 Association Result Set
See section 2.2.5.3. The result set MUST contain zero or more rows.

3.1.5.50 proc_ar_GetCacheInvalidationCountersWithCount

The proc_ar_GetCacheInvalidationCountersWithCount stored procedure is called to retrieve current cache version stamp information along with the count of version stamp for the cache.

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

```
PROCEDURE proc_ar_GetCacheInvalidationCountersWithCount();
```

Return Code Values: An integer that the protocol client MUST ignore.

Results Sets: MUST return the Count result set, as specified in section 3.1.5.50.1, and the Cache Version Stamps result set, as specified in section 3.1.5.50.2.

3.1.5.50.1 Count Result Set
See section 2.2.5.4. The result set MUST contain one row.

3.1.5.50.2 Cache Version Stamps Result Set

The Cache Version Stamps result set returns information about the version stamps for the cache. Each row in the result set MUST represent object cache version stamp and relationship cache version stamp for a given MetadataObjectType. The version stamps for the cache MUST be in ascending order of their MetadataObjectType attribute.

The Cache Version Stamps result set is defined, using T-SQL syntax, as follows:

```
MetadataObjectType nvarchar(255),
ObjectCacheCounter int,
RelationshipCacheCounter int;
```

MetadataObjectType: The type of the MetadataObject. It MUST be MetadataObjectType, as specified in section 2.2.2.5.

ObjectCacheCounter: The Object cache version stamp for the MetadataObjectType.

RelationshipCacheCounter: The relationship cache version stamp for the MetadataObjectType.
**Result Sets:** The result set MUST contain zero or more rows.

### 3.1.5.51 proc_ar_GetChildTypeDescriptorsForTypeDescriptorWithCount

The `proc_ar_GetChildTypeDescriptorsForTypeDescriptorWithCount` stored procedure is called to retrieve the count and details of child **TypeDescriptors** for the specified **TypeDescriptor**.

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

```
PROCEDURE proc_ar_GetChildTypeDescriptorsForTypeDescriptorWithCount (  
    @ParentTypeDescriptorId        int
);
```

@ParentTypeDescriptorId: The **MetadataObjectId** for the parent **TypeDescriptor**. The value MUST be an **Id**, as specified in section 2.2.2.1.

**Return Code Values:** An integer that the protocol client MUST ignore.

**Result Sets:** MUST return two result sets in the following order:

1. A **Count** result set, as specified in section 3.1.5.51.1.
2. A **TypeDescriptor** result set, as specified in section 3.1.5.51.2.

#### 3.1.5.51.1 Count Result Set

See section 2.2.5.4. The result set MUST contain one row.

#### 3.1.5.51.2 TypeDescriptor Result Set

See section 2.2.5.14. The result set MUST contain zero or more rows.

### 3.1.5.52 proc_ar_GetDataClassById

The `proc_ar_GetDataClassById` stored procedure is called to retrieve the **DataClass** identified by the specified **MetadataObjectId**.

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

```
PROCEDURE proc_ar_GetDataClassById (  
    @MetadataObjectId            int
);
```

@MetadataObjectId: The **MetadataObjectId** of the **DataClass** to be returned. The value MUST be an **Id**, a specified in section 2.2.2.1.

**Return Code Values:** An integer that the protocol client MUST ignore.

**Result Set:** MUST return a **DataClass** result set, as specified in section 3.1.5.52.1.

#### 3.1.5.52.1 DataClass Result Set

See section 2.2.5.5. The result set MUST contain zero or one row.
3.1.5.53  proc_ar_GetDataClassesForSystemWithCount

The proc_ar_GetDataClassesForSystemWithCount stored procedure is called to retrieve the count and details of DataClasses contained by the specified LobSystem.

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

```sql
PROCEDURE proc_ar_GetDataClassesForSystemWithCount (  
    @SystemId                   int
);  
```

@SystemId: The MetadataObjectId for the LobSystem object. The value MUST be an Id, as specified in section 2.2.2.1.

Return Code Values: An integer that the protocol client MUST ignore.

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

1. A Count result set, as specified in section 2.2.5.4.
2. A DataClass result set, as specified in section 2.2.5.5.

3.1.5.53.1  Count Result Set

See section 2.2.5.4. The result set MUST contain one row.

3.1.5.53.2  DataClass Result Set

See section 2.2.5.5. The result set MUST contain zero or more rows.

3.1.5.54  proc_ar_GetDefaultValuesForTypeDescriptor

The proc_ar_GetDefaultValuesForTypeDescriptor stored procedure is called to retrieve DefaultValues associated with the TypeDescriptor identified by the specified MetadataObjectId.

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

```sql
PROCEDURE proc_ar_GetDefaultValuesForTypeDescriptor (  
    @TypeDescriptorId          int,  
    @ErrorCode                 int OUTPUT
);  
```

@TypeDescriptorId: The MetadataObjectId for the TypeDescriptor object. The value MUST be an Id, as specified in section 2.2.2.1.

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer that is listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-2</td>
<td>Object not found. The @TypeDescriptorId parameter does not identify a valid TypeDescriptor.</td>
</tr>
</tbody>
</table>

Return Code Values: An integer that the protocol client MUST ignore.
**Result Sets**: MUST return a single result set.

### 3.1.5.54.1 DefaultValues Result Set

The **Default Values** result set returns the **DefaultValues** of the **TypeDescriptor**. It MUST contain zero or more rows. The result set MUST contain zero rows if the specified **TypeDescriptor** does not have any **DefaultValues** defined for any of the **MethodInstances** using it.

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

```sql
Id              int,
Value           sql_variant,
TypeDescriptorId int,
MethodInstanceId int,
MethodInstanceName nvarchar(255);
```

**Id**: An implementation-specific identifier for the **DefaultValue**.

**Value**: The value of the **DefaultValue**.

**TypeDescriptorId**: The **MetadataObjectId** of the **TypeDescriptor** with which the **DefaultValue** is associated. It MUST be an **Id**, as specified in section 2.2.2.1.

**MethodInstanceId**: The **MetadataObjectId** of the **MethodInstance** with which the **DefaultValue** is associated. It MUST be an **Id**.

**MethodInstanceName**: The programmatic name of the **MethodInstance** identified by the specified **MetadataObjectId** equal to @MethodInstanceId.

### 3.1.5.55 proc_ar_GetDependentEntitiesForEntity

The **proc_ar_GetDependentEntitiesForEntity** stored procedure is called to retrieve **MetadataObjectIds** for the **Entities** that are referenced by the specified **Entity**. Whenever a **Method** for the specified **Entity** has a parameter with a **TypeDescriptor** that refers to another **Entity**, that entity information MUST be included. The information about the original **Entity** MUST be excluded.

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

```sql
PROCEDURE proc_ar_GetDependentEntitiesForEntity (
    @EntityId                int
);
```

**@EntityId**: The **MetadataObjectId** for the entity object. The value MUST be an **Id**, as specified in section 2.2.2.1.

**Return Code Values**: An integer that the protocol client MUST ignore.

**Result Sets**: MUST return a single **EntityId** result set, as specified in section 3.1.5.55.1.
3.1.5.55.1 EntityId Result Set

The EntityId result set returns MetadataObjectIds for a set of Entities. The result set can have zero or more rows. There MUST be no duplicate MetadataObjectIds in the result set.

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

    EntityId    int;

EntityId: The MetadataObjectId of an Entity.

3.1.5.56 proc_ar_GetEntitiesForAssociationAndRoleWithCount

The proc_ar_GetEntitiesForAssociationAndRoleWithCount stored procedure is invoked to retrieve the count and details of Entities representing an Association source or destination for the specified Association.

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

    PROCEDURE proc_ar_GetEntitiesForAssociationAndRoleWithCount (    
        @AssociationId int,     
        @EntityRole bit
    );

@AssociationId: The MetadataObjectId of the Association. The value MUST be an Id, as specified in section 2.2.2.1.

@EntityRole: A bit that specifies whether to return Entities representing an Association source or destination. The value of this parameter MUST be listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Source Entity.</td>
</tr>
<tr>
<td>1</td>
<td>Destination Entity.</td>
</tr>
</tbody>
</table>

Return Code Values: An integer that the protocol client MUST ignore.

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

1. A Count result set, as specified in section 3.1.5.56.1.
2. An Entity result set, as specified in section 3.1.5.56.2.

3.1.5.56.1 Count Result Set

See section 2.2.5.4. The result set MUST contain one row.

3.1.5.56.2 Entity Result Set

See section 2.2.5.6. The result set MUST contain zero or more rows.
3.1.5.57 proc_ar_GetEntitiesForSystemLikeNameWithCount

The proc_ar_GetEntitiesForSystemLikeNameWithCount stored procedure is called to retrieve the count and the Entities that satisfy all of the following constraints:

- **Entities** contained by the LobSystem with the specified MetadataObjectId.
- **Entities** whose name matches the given pattern.

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

```sql
PROCEDURE proc_ar_GetEntitiesForSystemLikeNameWithCount (  
    @MetadataObjectName    nvarchar(255),  
    @SystemId              int
);
```

@MetadataObjectName: The string that defines the name pattern of the Entities to be returned. The characters in this string MUST be in upper case. It can include wildcard characters. For example, if the @MetadataObjectName parameter is "A%", this stored procedure returns only the Entities whose names begin with either "A" or "a".

@SystemId: The MetadataObjectId of the LobSystem that contains the Entities to be returned. The value MUST be an Id, as specified in section 2.2.2.1.

Return Code Values: MUST return an integer that the protocol client MUST ignore.

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

1. A Count result set, as specified in section 3.1.5.57.1.
2. An Entity result set, as specified in section 3.1.5.57.2.

3.1.5.57.1 Count Result Set

See section 2.2.5.4. The result set MUST contain one row.

3.1.5.57.2 Entity Result Set

See section 2.2.5.6. The result set MUST contain zero or more rows.

3.1.5.58 proc_ar_GetEntitiesForSystemWithCount

The proc_ar_GetEntitiesForSystemWithCount stored procedure is invoked to get the Entities contained by the LobSystem with the specified MetadataObjectId, along with the count of such entities.

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

```sql
PROCEDURE proc_ar_GetEntitiesForSystemWithCount (  
    @SystemId    int
);
```

@SystemId: The MetadataObjectId of the LobSystem that contains the entities to be returned. The value MUST be an Id, as specified in section 2.2.2.1.
Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MUST return two result sets in the following order:
1. A Count result set, as specified in section 3.1.58.1.
2. An Entity result set, as specified in section 3.1.58.2.

3.1.58.1 Count Result Set
See section 2.2.5.4. The result set MUST contain one row.

3.1.58.2 Entity Result Set
See section 2.2.5.6. The result set MUST contain zero or more rows.

3.1.59 proc_ar_GetEntityById

The proc_ar_GetEntityById stored procedure is invoked to retrieve the Entity with the specified MetadataObjectId.

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

```sql
PROCEDURE proc_ar_GetEntityById (  
    @MetadataObjectId          int
);  
```

@MetadataObjectId: The MetadataObjectId for the Entity that is to be retrieved. The value MUST be an Id, as specified in section 2.2.2.1.

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MUST return a single Entity result set, as specified in section 3.1.59.1.

3.1.59.1 Entity Result Set
See section 2.2.5.6. The result set MUST contain zero or more rows.

3.1.60 proc_ar_GetFilterDescriptorById

The proc_ar_GetFilterDescriptorById stored procedure is called to retrieve the FilterDescriptor identified by the specified MetadataObjectId.

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

```sql
PROCEDURE proc_ar_GetFilterDescriptorById (  
    @MetadataObjectId         int
);  
```

@MetadataObjectId: The MetadataObjectId of the FilterDescriptor that is to be retrieved. The value MUST be an Id, as specified in section 2.2.2.1.

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MUST return a FilterDescriptor result set, as specified in section 3.1.60.1.
3.1.5.60.1 FilterDescriptor Result Set

See section 2.2.7. The result set MUST contain zero or one row.

3.1.5.61 proc_ar_GetFilterDescriptorsForMethodWithCount

The proc_ar_GetFilterDescriptorsForMethodWithCount stored procedure is called to retrieve the FilterDescriptors contained by the Method with the specified MetadataObjectId, along with the count of such FilterDescriptors.

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

```sql
PROCEDURE proc_ar_GetFilterDescriptorsForMethodWithCount (
    @MethodId int
);
```

@MethodId: The MetadataObjectId of the Method that contains the FilterDescriptors to be returned. The value MUST be an Id, as specified in section 2.2.1.

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MUST return a Count result set, as specified in section 3.1.5.61.1, and a FilterDescriptor result set, as specified in section 3.1.5.61.2.

3.1.5.61.1 Count Result Set

See section 2.2.4. The result set MUST contain one row.

3.1.5.61.2 FilterDescriptor Result Set

See section 2.2.7. The result set MUST contain zero or more rows.

3.1.5.62 proc_ar_GetIdentifierById

The proc_ar_GetIdentifierById stored procedure is called to retrieve the Identifier with the specified MetadataObjectId.

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

```sql
PROCEDURE proc_ar_GetIdentifierById (
    @MetadataObjectId int
);
```

@MetadataObjectId: The MetadataObjectId of the Identifier that is to be retrieved. The value MUST be an Id, as specified in section 2.2.1.

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MUST return an Identifier result set, as specified in section 3.1.5.62.1.

3.1.5.62.1 Identifier Result Set

See section 2.2.8. The result set MUST contain zero or more rows.
3.1.5.63 proc_ar_GetIdentifiersForEntityWithCount

The proc_ar_GetIdentifiersForEntityWithCount stored procedure is called to retrieve the Identifiers contained by the Entity with the specified MetadataObjectId, along with the count of such Identifiers.

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

```
PROCEDURE proc_ar_GetIdentifiersForEntityWithCount (  
  @EntityId                   int  
);  
```

@EntityId: The MetadataObjectId of the Entity that contains the Identifiers to be returned. The value MUST be an Id, as specified in section 2.2.2.1.

Return Code Values: An integer that the protocol client MUST ignore.

Results Sets: MUST return the Count result set, as specified in section 3.1.5.63.1, and the Identifier result set, as specified in section 3.1.5.63.2.

3.1.5.63.1 Count Result Set

See section 2.2.5.4. The result set MUST contain one row.

3.1.5.63.2 Identifier Result Set

See section 2.2.5.8. The result set MUST contain zero or more rows.

3.1.5.64 proc_ar_GetMethodById

The proc_ar_GetMethodById stored procedure is called to retrieve the Method with the specified MetadataObjectId.

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

```
PROCEDURE proc_ar_GetMethodById (  
  @MetadataObjectId          int  
);  
```

@MetadataObjectId: The MetadataObjectId of the Method that is to be retrieved. The value MUST be an Id, as specified in section 2.2.2.1.

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MUST return the Method result set, as specified in section 3.1.5.64.1.

3.1.5.64.1 Method Result Set

See section 2.2.5.9. The result set MUST contain zero or one row.
3.1.5.65 proc_ar_GetMethodInstanceById

The proc_ar_GetMethodInstanceById stored procedure is called to retrieve the MethodInstance with the specified MetadataObjectId.

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

```sql
PROCEDURE proc_ar_GetMethodInstanceById (  
    @MetadataObjectId int
);
```

@MetadataObjectId: The MetadataObjectId of the MethodInstance that is to be retrieved. The value MUST be an Id, as specified in section 2.2.2.1.

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MUST return the MethodInstance result set, as specified in section 3.1.5.65.1.

3.1.5.65.1 MethodInstance Result Set

See section 2.2.5.10. The result set MUST contain zero or more rows.

3.1.5.66 proc_ar_GetMethodInstancesForDataClassWithCount

The proc_ar_GetMethodInstancesForDataClassWithCount stored procedure is called to retrieve the MethodInstances that are contained by the DataClass with the specified MetadataObjectId, excluding those MethodInstances that are Associations, along with the count of such MethodInstances.

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

```sql
PROCEDURE proc_ar_GetMethodInstancesForDataClassWithCount (  
    @ClassId int
);
```

@ClassId: The MetadataObjectId of the DataClass that contains the MethodInstances to be returned. The value MUST be an Id, as specified in section 2.2.2.1.

Return Values: An integer that the protocol client MUST ignore.

Results Sets: MUST return the Count result set, as specified in section 3.1.5.66.1, and the MethodInstance result set, as specified in section 3.1.5.66.2.

3.1.5.66.1 Count Result Set

See section 2.2.5.4. The result set MUST contain one row.

3.1.5.66.2 MethodInstance Result Set

See section 2.2.5.10. The result set MUST contain zero or more rows.
3.1.5.67 proc_ar_GetMethodInstancesForMethodWithCount

The proc_ar_GetMethodInstancesForMethodWithCount stored procedure is called to retrieve the MethodInstances that are contained by the Method with the specified MetadataObjectId, excluding those MethodInstances that are Associations, along with the count of such MethodInstances.

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

```sql
PROCEDURE proc_ar_GetMethodInstancesForMethodWithCount (  
    @MethodId                   int
);
```

@MethodId: The MetadataObjectId of the Method that contains the MethodInstances to be returned. The value MUST be an Id, as specified in section 2.2.2.1.

Return Values: An integer that the protocol client MUST ignore.

Results Sets: MUST return the following result sets.

3.1.5.67.1 Count Result Set

See section 2.2.5.4. The result set MUST contain 1 row.

3.1.5.67.2 MethodInstance Result Set

See section 2.2.5.10. The result set MUST contain zero or more rows.

3.1.5.68 proc_ar_GetMethodsForDataClassWithCount

The proc_ar_GetMethodsForDataClassWithCount stored procedure is called to retrieve the Methods contained by the DataClass with the specified MetadataObjectId, along with the count of such methods.

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

```sql
PROCEDURE proc_ar_GetMethodsForDataClassWithCount (  
    @ClassId                    int
);
```

@ClassId: The MetadataObjectId of the DataClass that contains the Methods to be returned. The value MUST be an Id, as specified in section 2.2.2.1.

Return Values: An integer that the protocol client MUST ignore.

Results Sets: MUST return the Count result set, as specified in section 3.1.5.68.1, and the Method result set, as specified in section 3.1.5.68.2.

3.1.5.68.1 Count Result Set

See section 2.2.5.4. The result set MUST contain one row.
3.1.5.68.2 Method Result Set

See section 2.2.5.9. The result set MUST contain zero or more rows.

3.1.5.69 proc_ar_GetParameterById

The proc_ar_GetParameterById stored procedure is called to retrieve the Parameter with the specified MetadataObjectId.

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

```sql
PROCEDURE proc_ar_GetParameterById (  
    @MetadataObjectId             int  
);  

@MetadataObjectId: The MetadataObjectId of the Parameter that is to be retrieved. The value MUST be an Id, as specified in section 2.2.2.1.

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MUST return the Parameter result set, as specified in section 3.1.5.69.1.

3.1.5.69.1 Parameter Result Set

See section 2.2.5.11. The result set MUST contain zero or more rows.

3.1.5.70 proc_ar_GetParametersForMethodWithCount

The proc_ar_GetParametersForMethodWithCount stored procedure is called to retrieve Parameters information for the Method with the given MetadataObjectId.

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

```sql
PROCEDURE proc_ar_GetParametersForMethodWithCount (  
    @MethodId                     int  
);  

@MethodId: The MetadataObjectId of the Method for which Parameter information is to be retrieved. The value MUST be an Id, as specified in section 2.2.2.1.

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MUST return the Count result set, as specified in section 3.1.5.70.1, and the Parameter result set, as specified in section 3.1.5.70.2.

3.1.5.70.1 Count Result Set

See section 2.2.5.4. The result set MUST contain one row.

3.1.5.70.2 Parameter Result Set

See section 2.2.5.11. The result set MUST contain zero or more rows.
3.1.5.71 proc_ar_GetPropertiesForMetadataObject

The proc_ar_GetPropertiesForMetadataObject stored procedure is invoked to retrieve Properties for the MetadataObject with the given MetadataObjectId.

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

```sql
PROCEDURE proc_ar_GetPropertiesForMetadataObject (  
    @MetadataObjectId          int,  
    @ErrorCode                 int OUTPUT  
);
```

@MetadataObjectId: The MetadataObjectId of the MetadataObject whose Properties are to be returned. The value MUST be an Id, as specified in section 2.2.2.1.

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer that is listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-2</td>
<td>Object not found.</td>
</tr>
</tbody>
</table>

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MUST return the Property result set, as specified in section 3.1.5.71.1.

3.1.5.71.1 Property Result Set

The Property result set contains the name and value of the Property associated with a MetadataObject. Each row represents one Property. The result set MUST contain zero or more rows.

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

```sql
Name               nvarchar(255),  
Value              sql_variant;  
```

Name: The programmatic name of the Property.

Value: The value of the Property.

3.1.5.72 proc_ar_GetRootTypeDescriptorForParameter

The proc_ar_GetRootTypeDescriptorForParameter stored procedure is called to retrieve the TypeDescriptor information that is contained by the Parameter with the specified MetadataObjectId and has no parent TypeDescriptor.

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

```sql
PROCEDURE proc_ar_GetRootTypeDescriptorForParameter (  
    @ParameterId                int  
);
```
@ParameterId: Provides the MetadataObjectId value of an existing Parameter that contains the TypeDescriptor to be returned. The value MUST be an Id, as specified in section 2.2.2.1.

Return Values: An integer that the protocol client MUST ignore.

Results Sets: MUST return the TypeDescriptor result set, as specified in section 3.1.5.72.1.

3.1.5.72.1 TypeDescriptor Result Set

See section 2.2.5.14. The result set MUST contain zero or one row.

3.1.5.73 proc_ar_GetSystemById

The proc_ar_GetSystemById stored procedure is invoked to retrieve the LobSystem with the specified MetadataObjectId. The T-SQL syntax for the stored procedure is as follows:

```sql
PROCEDURE proc_ar_GetSystemById (  
    @MetadataObjectId            int  
);  
```

@MetadataObjectId: The MetadataObjectId of the LobSystem to be retrieved. The value MUST be an Id, as specified in section 2.2.2.1.

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MUST return the System result set, as specified in section 3.1.5.73.1.

3.1.5.73.1 System Result Set

See section 2.2.5.13. The result set MUST contain zero or one row.

3.1.5.74 proc_ar_GetSystemDataBySystemName

The proc_ar_GetSystemDataBySystemName stored procedure is called to retrieve the binary an implementation-specific <37> business logic module associated with the given LobSystem. The business logic module can be used to provide implementation-specific <38> business logic (2) that MAY <39> be referenced by the TypeDescriptorTypeNames for the TypeDescriptors contained by the Parameters contained by the Methods contained by the Entities contained by the LobSystem specified by the value of @SystemName.

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

```sql
PROCEDURE proc_ar_GetSystemDataBySystemName (  
    @SystemName             nvarchar(255)  
);  
```

@SystemName: The programmatic name of the LobSystem. The value MUST be a Name, as specified in section 2.2.2.2.
Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MUST return the System Data result set, as specified in section 3.1.5.74.1.

3.1.5.74.1 System Data Result Set

The System Data result set contains the binary representation of the business logic module associated with a single LobSystem. The result set MUST contain zero or one row.

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

```
Length int,
Data image;
```

Length: The size of the binary business logic module in bytes.

Data: The binary business logic module.

3.1.5.75 proc_ar_GetSystemInstanceById

The proc_ar_GetSystemInstanceById stored procedure retrieves the LobSystemInstance with the specified MetadataObjectId.

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

```
PROCEDURE proc_ar_GetSystemInstanceById (
    @MetadataObjectId int
);
```

@MetadataObjectId: The MetadataObjectId of the LobSystemInstance to be retrieved. The value MUST be an Id, as specified in section 2.2.2.1.

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MUST return the System Instance result set, as specified in section 3.1.5.75.1.

3.1.5.75.1 System Instance Result Set

See section 2.2.5.12. The result set MUST contain zero or one row.

3.1.5.76 proc_ar_GetSystemInstancesForSystemWithCount

The proc_ar_GetSystemInstancesForSystemWithCount stored procedure is called to retrieve LobSystemInstances information contained by the LobSystem with the specified MetadataObjectId.

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

```
PROCEDURE proc_ar_GetSystemInstancesForSystemWithCount (
    @SystemId int
);
```
@SystemId: The MetadataObjectId of the LobSystem. The value MUST be an Id, as specified in section 2.2.2.1.

Return Code Values: An integer that the protocol client MUST ignore.

Results Sets: MUST return the Count result set, as specified in section 3.1.5.76.1, and the System Instance result set, as specified in section 3.1.5.76.2.

3.1.5.76.1 Count Result Set

See section 2.2.5.4. The result set MUST contain one row.

3.1.5.76.2 System Instance Result Set

See section 2.2.5.12. The result set MUST contain zero or more rows.

3.1.5.77 proc_ar_GetSystemsLikeNameWithCount

The proc_ar_GetSystemsLikeNameWithCount stored procedure is called to retrieve the count and the LobSystems that satisfy either one of the following constraints:

- Any LobSystems that have names that match the specified pattern.
- Any LobSystems whose localized names match the specified pattern and either have the given LCID or their LCID is zero.

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

```sql
PROCEDURE proc_ar_GetSystemsLikeNameWithCount (
    @MetadataObjectName nvarchar(255),
    @LCID int
);
```

@MetadataObjectName: The string that specifies the name pattern of the LobSystems to be returned. The characters in this string MUST be in upper case. It can include wildcard characters. For example, if the @MetadataObjectName is "A%", this stored procedure returns only the LobSystems with names beginning with either "A" or "a".

@LCID: The LCID of the localized names of the LobSystems to be fetched. In the following two cases, this parameter MUST be ignored:

- @MetadataObjectName matches the name of the LobSystem.
- @MetadataObjectName matches the localized name and the LCID of localized name is zero.

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MUST return the Count result set, as specified in section 3.1.5.77.1, and the System result set, as specified in section 3.1.5.77.2.

3.1.5.77.1 Count Result Set

See section 2.2.5.4. The result set MUST contain one row.
3.1.5.77.2 System Result Set

See section 2.2.5.13. The result set MUST contain zero or more rows.

3.1.5.78 proc_ar_GetTypeDescriptorById

The proc_ar_GetTypeDescriptorById stored procedure is called to retrieve the TypeDescriptor with the specified MetadataObjectId.

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

```
PROCEDURE proc_ar_GetTypeDescriptorById (
   @MetadataObjectId               int
);
```

@MetadataObjectId: The MetadataObjectId of the TypeDescriptor that is to be retrieved. The value MUST be an Id, as specified in section 2.2.2.1.

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MUST return the TypeDescriptor result set, as specified in section 3.1.5.78.1.

3.1.5.78.1 TypeDescriptor Result Set

See section 2.2.5.14. The result set MUST contain zero or one row.

3.1.5.79 proc_ar_GetTypeDescriptorsByNameAndParameter

The proc_ar_GetTypeDescriptorsByNameAndParameter stored procedure is called to retrieve TypeDescriptors that have the specified name and are either the root or the child TypeDescriptor of the specified Parameter.

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

```
PROCEDURE proc_ar_GetTypeDescriptorsByNameAndParameter (
   @ParameterId                 int,
   @Name                        nvarchar(255)
);
```

@ParameterId: The MetadataObjectId of an existing Parameter for which TypeDescriptors are to be returned. The value MUST be an Id, as specified in section 2.2.2.1.

@Name: The name of the TypeDescriptor to be returned. The value MUST be a Name, as specified in section 2.2.2.2.

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MUST return the TypeDescriptor result set, as specified in section 3.1.5.79.1.

3.1.5.79.1 TypeDescriptor Result Set

See section 2.2.5.14. The result set MUST contain zero or more rows.
3.1.5.80 proc_ar_GetTypeDescriptorsForFilterDescriptorWithCount

The proc_ar_GetTypeDescriptorsForFilterDescriptorWithCount stored procedure is called to retrieve TypeDescriptors that reference the specified FilterDescriptor.

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

```sql
PROCEDURE proc_ar_GetTypeDescriptorsForFilterDescriptorWithCount (  
    @FilterDescriptorId int
);
```

@FilterDescriptorId: The MetadataObjectId of the FilterDescriptor for which TypeDescriptors are to be returned. The value MUST be an Id, as specified in section 2.2.2.1.

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MUST return the following two result sets in the listed order:
1. A Count result set, as specified in section 3.1.5.80.1.
2. A TypeDescriptor result set, as specified in section 3.1.5.80.2.

3.1.5.80.1 Count Result Set

See section 2.2.5.4. The result set MUST contain one row.

3.1.5.80.2 TypeDescriptor Result Set

See section 2.2.5.14. The result set MUST contain zero or more rows.

3.1.5.81 proc_ar_SetAccessControlEntryForMetadataObject

The proc_ar_SetAccessControlEntryForMetadataObject stored procedure adds an ACE and associates it with the MetadataObject identified by the specified MetadataObjectId. If an ACE with @IdentityName already exists, it is replaced by the newly created ACE.

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

```sql
PROCEDURE proc_ar_SetAccessControlEntryForMetadataObject (  
    @MetadataObjectId int,  
    @IdentityName nvarchar(250),  
    @DisplayName nvarchar(250),  
    @RawSid varbinary(512),  
    @Rights bigint
);
```

@MetadataObjectId: The MetadataObjectId of the MetadataObject to which the ACE is to be added. The value MUST be an Id, as specified in section 2.2.2.1.

@IdentityName: The programmatic name of the security principal (2).

@DisplayName: The name of the security principal (2) used for display purposes.

@RawSid: The SID, if the security principal (2) is a Windows security principal. If the security principal (2) is not a Windows security principal, the value MUST be NULL.
@Rights: The permissions available to the security principal (2) for the MetadataObject identifier by the MetadataObjectId. It MUST be a MetadataRights, as specified in section 2.2.2.22.

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MUST NOT return any result sets.

3.1.5.82 proc_ar_SetDefaultAction

The proc_ar_SetDefaultAction stored procedure sets or clears the default Action on the specified Entity.

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

```sql
PROCEDURE proc_ar_SetDefaultAction (  
    @EntityId                  int,  
    @ActionName                nvarchar(255),  
    @ErrorCode                 int OUTPUT  
);  
```

@EntityId: The MetadataObjectId of the Entity to set or clear the DefaultAction for. This parameter MUST be non-NULL. The value MUST be an Id, as specified in section 2.2.1.1.

@ActionName: This parameter MUST take values defined in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NULL</td>
<td>This operation clears the DefaultAction for the Entity with MetadataObjectId equal to @EntityId.</td>
</tr>
<tr>
<td>Not NULL</td>
<td>This operation sets the DefaultAction for the Entity with MetadataObjectId equal to @EntityId to the Action with the name @ActionName, if such Action is contained by this Entity; otherwise, the @ErrorCode parameter MUST be set to &quot;-2&quot; and DefaultAction for this Entity MUST be unchanged.</td>
</tr>
</tbody>
</table>

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer that is listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-2</td>
<td>The value of @ActionName does not match the names of any of the Actions contained by the Entity with MetadataObjectId equal to @EntityId.</td>
</tr>
<tr>
<td>A positive integer</td>
<td>A system specific error with the given error code has occurred.</td>
</tr>
</tbody>
</table>

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MAY <40> return zero or more result sets, which the protocol client MUST ignore.

3.1.5.83 proc_ar_SetDefaultValuesForTypeDescriptor

The proc_ar_SetDefaultValuesForTypeDescriptor stored procedure is called to set a DefaultValue for TypeDescriptor identified by the specified MetadataObjectId in

---

Preliminary
@TypeDescriptorId and a MethodInstance identified by the specified MetadataObjectId in @MethodInstanceId.

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

```sql
PROCEDURE proc_ar_SetDefaultValuesForTypeDescriptor (  
    @TypeDescriptorId            int,  
    @MethodInstanceId            int,  
    @Value                       sql_variant,  
    @ErrorCode                   int OUTPUT  
);  
```

@TypeDescriptorId: The MetadataObjectId of the TypeDescriptor associated with the DefaultValue to be set. The value MUST be an Id, as specified in section 2.2.2.1.

@MethodInstanceId: The MetadataObjectId of the MethodInstance associated with the DefaultValue to be set. The value MUST be an Id.

@Value: The DefaultValue.

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer that is listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
<td></td>
</tr>
<tr>
<td>-2</td>
<td>At least one of the following is true:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• A TypeDescriptor with MetadataObjectId equal to @TypeDescriptorId is not found.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• A MethodInstance with MetadataObjectId equal to @MethodInstanceId is not found.</td>
<td></td>
</tr>
<tr>
<td>-3</td>
<td>The TypeDescriptor with MetadataObjectId equal to @TypeDescriptorId already has the implementation-specific maximum number of DefaultValues allowed.</td>
<td></td>
</tr>
<tr>
<td>-600</td>
<td>Parameter of the TypeDescriptor with MetadataObjectId equal to @TypeDescriptorId is not contained by the Method that contains MethodInstance with MetadataObjectId equal to @MethodInstanceId.</td>
<td></td>
</tr>
</tbody>
</table>

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MUST NOT return any result sets.

### 3.1.5.84 proc_ar_SetSystemDataBySystemName

The proc_ar_SetSystemDataBySystemName stored procedure is called to store the binary an implementation-specific business logic module associated with the given LobSystem. The business logic module can be used to provide implementation-specific business logic (2) that MAY be referenced by the TypeDescriptorTypeNames for the TypeDescriptors contained by the Parameters contained by the Methods contained by the Entities contained by the LobSystem specified by the value of @SystemName.

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

```sql
PROCEDURE proc_ar_SetSystemDataBySystemName (  
);  
```
@SystemName: The programmatic name of the LobSystem to set the business logic module for. The value of this parameter MUST match the programmatic name of an existing LobSystem in the metadata store. The value MUST be a Name, as specified in section 2.2.2.2.

@AssemblyName: The Business Logic Module Reference.

@Length: The size of the binary business logic module in bytes.

@Data: The binary business logic module.

Return Values: An integer that the protocol client MUST ignore.

Result Sets: MUST NOT return any result sets.

3.1.5.85 proc_ar_UpdateActionById

The proc_ar_UpdateActionById stored procedure is called to change the attributes of the Action identified by the specified MetadataObjectId.

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

```
PROCEDURE proc_ar_UpdateActionById (  
    @Id                      int,  
    @Name                    nvarchar(50),  
    @IsCached                bit,  
    @Version                 int OUTPUT,  
    @Position                int,  
    @IsDisplayed             bit,  
    @IsOpenedInNewWindow     bit,  
    @Icon                    nvarchar(2080),  
    @URL                     nvarchar(2080),  
    @ErrorCode               int OUTPUT
);
```

@Id: The MetadataObjectId of the Action that is to be updated. The value MUST be an Id, as specified in section 2.2.2.1.

@Name: The name of the Action. The value MUST be a Name, as specified in section 2.2.2.2.

@IsCached: A bit that specifies whether this Action is frequently used. The value MUST be an IsCached, as specified in section 2.2.2.3.

@Version: The value of version at the time Action with the specified MetadataObjectId was last read. This value MUST be incremented in the metadata store every time the Action is updated. The value MUST wrap around after reaching 2147483646.

@Position: The order of this Action among the other Actions displayed in a user interface for this Entity. The value MUST be a Position, as specified in section 2.2.2.6.
@IsDisplayed: A bit that provides a hint on whether the Action is displayed in the user interface presented to the user. The value MUST be an IsDisplayed, as specified in section 2.2.2.7.

@IsOpenedInNewWindow: A bit that a hint on whether the results of executing the Action are displayed in a new window in the user interface presented to the user. The value MUST be an IsOpenedInNewWindow, as specified in section 2.2.2.9.

@Icon: The URL of the icon associated with this Action. The value MUST be an Icon, as specified in section 2.2.2.10.

@URL: The URL associated with this Action. The value MUST be a URL, as specified in section 2.2.2.11.

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-1</td>
<td>The Entity that contains this Action already contains another Action with the specified @Name.</td>
</tr>
<tr>
<td>-2</td>
<td>An Action with the specified @Id does not exist.</td>
</tr>
<tr>
<td>-6</td>
<td>An Action with the specified MetadataObjectId has been updated by a context other than the one that it has been currently read by. This happens when the version specified does not match with the current version of the Action.</td>
</tr>
<tr>
<td>A positive integer</td>
<td>A T-SQL error code.</td>
</tr>
</tbody>
</table>

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MAY return zero or more result sets that the protocol client MUST ignore.

3.1.5.86 proc_ar_UpdateActionParameterById

The proc_ar_UpdateActionParameterById stored procedure is called to change the attributes of the ActionParameter identified by the specified MetadataObjectId.

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

```sql
PROCEDURE proc_ar_UpdateActionParameterById (  
    @Id                        int,  
    @IsCached                  bit,  
    @Version                   int OUTPUT,  
    @Name                      nvarchar(50),  
    @Index                     tinyint,  
    @ErrorCode                 int OUTPUT  
);  
```

@Id: The MetadataObjectId of the ActionParameter that is to be updated. The value MUST be an Id, as specified in section 2.2.2.1.

@IsCached: A bit that specifies whether this ActionParameter is frequently used. The value MUST be an IsCached, as specified in section 2.2.2.3.
@Version: The value of `version` at the time `ActionParameter` with the specified `MetadataObjectId` was last read. This value MUST be incremented in the metadata store every time the `ActionParameter` is updated. The value MUST wrap around after reaching 2147483646.

@Name: The name of the `ActionParameter`. The value MUST be a `Name`, as specified in section 2.2.2.2.

@Index: A value, indicating the position of this `ActionParameter` among the `ActionParameters` of the `Action` that contains this `ActionParameter`. It MUST be an `Index`, as specified in section 2.2.2.12.

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an `integer` listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-1</td>
<td>The <code>Action</code> that contains this <code>ActionParameter</code> already contains another <code>ActionParameter</code> with the specified <code>@Name</code>.</td>
</tr>
<tr>
<td>-2</td>
<td>An <code>ActionParameter</code> with the specified <code>@Id</code> does not exist.</td>
</tr>
<tr>
<td>-6</td>
<td>An <code>ActionParameter</code> with the specified <code>MetadataObjectId</code> has been updated by a context other than the one that it has been currently read by. This happens when the version specified does not match with the current version of the <code>ActionParameter</code>.</td>
</tr>
<tr>
<td>A positive integer</td>
<td>A T-SQL error code.</td>
</tr>
</tbody>
</table>

Return Code Values: An `integer` that the protocol client MUST ignore.

Result Sets: MAY return zero or more result sets that the protocol client MUST ignore.

### 3.1.5.87 proc_ar_UpdateAssociationById

The `proc_ar_UpdateAssociationById` stored procedure is called to change the attributes of the `Association` identified by its given `MetadataObjectId`.

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

```sql
PROCEDURE proc_ar_UpdateAssociationById (  
    @Id                      int,  
    @Name                    nvarchar(255),  
    @IsCached                bit,  
    @ReturnTypeDescriptorId  int,  
    @Type                    tinyint,  
    @Version                 int OUTPUT,  
    @ErrorCode               int OUTPUT  
);  
```

@Id: The `MetadataObjectId` of the `Association` that is to be updated. The value MUST be an `Id`, as specified in section 2.2.2.1.

@Name: The name of the `Association`. The value MUST be a `Name`, as specified in section 2.2.2.2.
@IsCached: A bit that specifies whether this Association is frequently used. The value MUST be an IsCached, as specified in section 2.2.2.3.

@ReturnTypeDescriptorId: The MetadataObjectId of the ReturnTypeDescriptor. The value MUST be an Id. It MUST be equal to the ReturnTypeDescriptor specified when the Association was created.

@Type: The type of the Association. This MUST be "4".

@Version: The value of version at the time Association with the specified MetadataObjectId was last read. This value MUST be incremented in the metadata store every time the Association is updated. The value MUST wrap around after reaching 2147483646.

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-1</td>
<td>The LobSystem already has another Association with MetadataObject name equal to @Name.</td>
</tr>
<tr>
<td>-2</td>
<td>An Association with MetadataObjectId equal to @Id is not available.</td>
</tr>
<tr>
<td>-6</td>
<td>Association with MetadataObjectId equal to @Id has been updated by a context other than the one that it has been currently read by. This happens when the value of @Version does not match with the version for the Association.</td>
</tr>
<tr>
<td>-500</td>
<td>This happens when the value of @ReturnTypeDescriptorId does not match with the MetadataObjectId of the Returntypedescriminator of the Association or if the value of @Type does not match with the MethodInstanceType for the Association, which is set to &quot;4&quot; on creation.</td>
</tr>
</tbody>
</table>

A positive integer A T-SQL error code.

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MAY <46> return zero or more result sets that the protocol client MUST ignore.

3.1.5.88 proc_ar_UpdateEntityById

The proc_ar_UpdateEntityById stored procedure is invoked to change the attributes of the Entity identified by the specified MetadataObjectId.

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

```sql
PROCEDURE proc_ar_UpdateEntityById ( 
    @Id                          int, 
    @Name                        nvarchar(255), 
    @IsCached                    bit, 
    @Version                     int OUTPUT, 
    @SystemId                    int, 
    @EstimatedInstanceCount      int, 
    @errorCode                   int OUTPUT 
); 
```
@Id: The identifier for the Entity that is to be updated. The value MUST be an Id, as specified in section 2.2.2.1.

@Name: The name of the Entity. The value MUST be a Name, as specified in section 2.2.2.2.

@IsCached: A bit that specifies whether this Entity is frequently used. The value MUST be an IsCached, as specified in section 2.2.2.3.

@Version: The value of version at the time the Entity with the specified MetadataObjectId was last read. This value MUST be incremented in the metadata store every time the Entity is updated. The value MUST wrap around after reaching 2147483646.

@SystemId: The MetadataObjectId of the LobSystem with which the Entity is associated. This MUST be an Id. This MUST be the MetadataObjectId of a LobSystem currently in the metadata store.

@EstimatedInstanceCount: The estimated number of instances of this Entity present within the LobSystemInstance. The value MUST be an EstimatedInstanceCount, as specified in section 2.2.2.4.

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-1</td>
<td>The LobSystem that contains this Entity already contains another Entity with @Name.</td>
</tr>
<tr>
<td>-2</td>
<td>An Entity with the specified @Id does not exist.</td>
</tr>
<tr>
<td>-3</td>
<td>The LobSystem already contains the implementation-specific maximum allowed number of Entities.</td>
</tr>
<tr>
<td>-6</td>
<td>An Entity with the specified MetadataObjectId has been updated by a context other than the one that it has been currently read by. This happens when the version specified does not match with the current version of the Entity.</td>
</tr>
</tbody>
</table>

A positive integer A T-SQL error code.

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MAY <47> return zero or more result sets that the protocol client MUST ignore.

3.1.5.89 proc_ar_UpdateFilterDescriptorById

The proc_ar_UpdateFilterDescriptorById stored procedure changes the attributes of the FilterDescriptor identified by the specified MetadataObjectId.

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

```sql
PROCEDURE proc_ar_UpdateFilterDescriptorById (  @Id                      int,  @Name                    nvarchar(255),  @IsCached                bit,  @Version                 int OUTPUT,  @TypeName                nvarchar(255),

)"`
@ErrorCode: The MetadataObjectId of the FilterDescriptor that is to be updated. The value MUST be an Id, as specified in section 2.2.2.1.

@Name: The name of the FilterDescriptor. The value MUST be a Name, as specified in section 2.2.2.2.

@IsCached: A bit that specifies whether this FilterDescriptor is frequently used. The value MUST be an IsCached, as specified in section 2.2.2.3.

@Version: The value of version at the time FilterDescriptor with the specified MetadataObjectId was last read. This value MUST be incremented in the metadata store every time the FilterDescriptor is updated. The value MUST wrap around after reaching 2147483646.

@TypeName: The type name of the FilterDescriptor. The value MUST be an FilterDescriptorTypeName, as specified in section 2.2.2.13.

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-1</td>
<td>The Method that contains this FilterDescriptor already contains another FilterDescriptor with the specified @Name.</td>
</tr>
<tr>
<td>-2</td>
<td>A FilterDescriptor with the specified @Id does not exist.</td>
</tr>
<tr>
<td>-6</td>
<td>A FilterDescriptor with the specified MetadataObjectId has been updated by a context other than the one that it has been currently read by. This happens when the version specified does not match with the current version of the FilterDescriptor.</td>
</tr>
<tr>
<td>A positive integer</td>
<td>A T-SQL error code.</td>
</tr>
</tbody>
</table>

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MAY <48> return zero or more result sets that the protocol client MUST ignore.

3.1.5.90 proc_ar_UpdateIdentifierById

The proc_ar_UpdateIdentifierById stored procedure is invoked to change the attributes of the Identifier identified by the specified MetadataObjectId.

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

```
PROCEDURE proc_ar_UpdateIdentifierById (
    @Id int,
    @Name nvarchar(255),
    @IsCached bit,
    @Version int OUTPUT,
    @TypeName nvarchar(255),
    @ErrorCode int OUTPUT
)
```
@Id: The MetadataObjectId of the Identifier that is to be updated. The value MUST be an Id, as specified in section 2.2.2.1.

@Name: The name of the Identifier. The value MUST be a Name, as specified in section 2.2.2.2.

@IsCached: A bit that specifies whether this Identifier is frequently used. The value MUST be an IsCached, as specified in section 2.2.2.3.

@Version: The value of version at the time the Identifier with the specified MetadataObjectId was last read. This value MUST be incremented in the metadata store every time the Identifier is updated. The value MUST wrap around after reaching 2147483646.

@TypeName: The type name of the Identifier. The value MUST be an IdentifierTypeName, as specified in section 2.2.2.14.

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-1</td>
<td>The Entity that contains this Identifier already contains another Identifier with the specified @Name.</td>
</tr>
<tr>
<td>-2</td>
<td>An Identifier with the specified @Id does not exist.</td>
</tr>
<tr>
<td>-6</td>
<td>An Identifier with the specified MetadataObjectId has been updated by a context other than the one that it has been currently read by. This happens when the version specified does not match with the current version of the Identifier.</td>
</tr>
</tbody>
</table>

A positive integer: A T-SQL error code.

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MAY return zero or more result sets that the protocol client MUST ignore.

3.1.5.91 proc_ar_UpdateMethodById

The proc_ar_UpdateMethodById stored procedure changes the attributes of the Method identified by the specified MetadataObjectId.

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

```sql
PROCEDURE proc_ar_UpdateMethodById ( 
    @Id int, 
    @Name nvarchar(255), 
    @IsCached bit, 
    @Version int OUTPUT, 
    @IsStatic bit, 
    @ErrorCode int OUTPUT
);```
@Id: The MetadataObjectId of the Method that is to be updated. The value MUST be an Id, as specified in section 2.2.2.1.

@Name: The name of the Method. The value MUST be a Name, as specified in section 2.2.2.2.

@IsCached: A bit that specifies whether this Method is frequently used. The value MUST be an IsCached, as specified in section 2.2.2.3.

@Version: The value of version at the time the Method with the specified MetadataObjectId was last read. This value MUST be incremented in the metadata store every time the Method is updated. The value MUST wrap around after reaching 2147483646.

@IsStatic: A bit specifying whether the Method is associated with an EntityInstance. The value MUST be an IsStatic, as specified in section 2.2.2.23.

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-1</td>
<td>The Entity that contains this Method already contains another Method with the specified @Name.</td>
</tr>
<tr>
<td>-2</td>
<td>A Method with the specified @Id does not exist.</td>
</tr>
<tr>
<td>-6</td>
<td>A Method with the specified MetadataObjectId has been updated by a context other than the one that it has been currently read by. This happens when the version specified does not match with the current version of the Method.</td>
</tr>
</tbody>
</table>

A positive integer A T-SQL error code.

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MAY <50> return zero or more result sets that the protocol client MUST ignore.

3.1.5.92 proc_ar_UpdateMethodInstanceById

The proc_ar_UpdateMethodInstanceById stored procedure is called to change attributes of MethodInstance with the given MetadataObjectId.

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

```sql
PROCEDURE proc_ar_UpdateMethodInstanceById (  
  @Id                          int,  
  @Name                        nvarchar(255),  
  @IsCached                    bit,  
  @Version                     int OUTPUT,  
  @ReturnTypeDescriptorId      int,  
  @Type                        tinyint,  
  @ErrorCode                   int OUTPUT  
);
```

@Id: This parameter MUST contain the MetadataObjectId of the MethodInstance to be updated. The value MUST be an Id, as specified in section 2.2.2.1.
@Name: This parameter MUST be used to set the value of the programmatic name of MethodInstance. The value MUST be a Name, as specified in section 2.2.2.

@IsCached: A bit that specifies whether this MethodInstance is frequently used. The value MUST be an IsCached, as specified in section 2.2.3.

@Version: The value of version at the time the MethodInstance with the specified MetadataObjectId was last read. This value MUST be incremented in the metadata store every time the MethodInstance is updated. The value MUST wrap around after reaching 2147483646.

@ReturnTypeDescriptorId: The MetadataObjectId of the ReturnTypeDescriptor. The TypeDescriptor MUST exist in the metadata store. The value MUST be an Id.

@Type: The type of the MethodInstance. The value MUST be a MethodInstanceType, as specified in section 2.2.2.15.

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer that is listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-1</td>
<td>The Method that contains this MethodInstance already contains another MethodInstance with the MetadataObject name equal to @Name.</td>
</tr>
<tr>
<td>-2</td>
<td>A MethodInstance with MetadataObjectId equal to @Id doesn’t exist.</td>
</tr>
<tr>
<td>-6</td>
<td>The MethodInstance with MetadataObjectId equal to @Id has been updated by a context other than the one that it has been currently read by. This happens when the value of @Version does not match with the version for the MethodInstance.</td>
</tr>
<tr>
<td>-200</td>
<td>An Entity cannot contain more than one Method that contains at most one MethodInstance of MethodInstanceType &quot;Finder&quot;, or a Method that contains more than one MethodInstance with MethodInstanceType &quot;Finder&quot;.</td>
</tr>
<tr>
<td>-201</td>
<td>An Entity cannot contain more than one Method that contains at most one MethodInstance with MethodInstanceType &quot;SpecificFinder&quot;, or a Method that contains more than one MethodInstance with MethodInstanceType &quot;SpecificFinder&quot;.</td>
</tr>
<tr>
<td>-202</td>
<td>An Entity cannot contain more than one Method that contains at most one MethodInstance with MethodInstanceType &quot;IdEnumerator&quot;, or a Method that contains more than one MethodInstance with MethodInstanceType &quot;IdEnumerator&quot;.</td>
</tr>
<tr>
<td>-203</td>
<td>Method that contains Parameter that contains TypeDescriptor with MetadataObjectId equal to @ReturnTypeDescriptorId, does not contain the MethodInstance with MetadataObjectId equal to @Id.</td>
</tr>
<tr>
<td>-204</td>
<td>Parameter that contains TypeDescriptor with MetadataObjectId equal to @ReturnTypeDescriptorId is a Parameter with Direction of &quot;1&quot;.</td>
</tr>
<tr>
<td>-205</td>
<td>An Entity cannot contain more than one Method that contains at most one MethodInstance with MethodInstanceType &quot;AccessChecker&quot;, or a Method that contains more than one MethodInstance with MethodInstanceType &quot;AccessChecker&quot;.</td>
</tr>
</tbody>
</table>

A positive integer A T-SQL error code.

Return Values: An integer that the protocol client MUST ignore.
Result Sets: MAY <51> return zero or more result sets that the protocol client MUST ignore.

3.1.5.93 proc_ar_UpdateParameterById

The proc_ar_UpdateParameterById stored procedure is called to change the attributes of the Parameter identified by the specified MetadataObjectId.

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

```sql
PROCEDURE proc_ar_UpdateParameterById (
    @Id                        int,
    @Name                      nvarchar(255),
    @IsCached                  bit,
    @Version                   int OUTPUT,
    @OrdinalNumber             tinyint OUTPUT,
    @Direction                 tinyint,
    @TypeReflectorTypeName     nvarchar(255),
    @ErrorCode                 int OUTPUT
);
```

@Id: This parameter MUST contain the MetadataObjectId of the Parameter to be updated.

@Name: This parameter MUST be used to set the value of the programmatic name of the Parameter.

@IsCached: A bit that specifies whether this Parameter is frequently used. The value MUST be an IsCached, as specified in section 2.2.2.3.

@Version: The value of version at the time the Parameter with the specified MetadataObjectId was last read. This value MUST be incremented in the metadata store every time the Parameter is updated. The value MUST wrap around after reaching 2147483646.

@OrdinalNumber: The position of the Parameter in the Parameter signature of the Method containing this Parameter. If the position is the same as another Parameter's position for the same parent Method, the other Parameter's position, along with all Parameters positioned subsequently are incremented. When the stored procedure returns, all Parameters of the Method containing this Parameter MUST have positions in the range 0 to X, where X+1 is the number of Parameters in the Method. Parameters in the Method other than this Parameter MUST NOT have their relative positioning altered.

@Direction: This parameter MUST be used to set the direction in which the Parameter is passed. It MUST be a Direction, as specified in section 2.2.2.16.

@TypeReflectorTypeName: The type name of the TypeReflector to be used to resolve the native type of this parameter. The value MUST be a TypeReflectorTypeName, as specified in section 2.2.2.17.

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer that is listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-1</td>
<td>The Method that contains this Parameter already contains another Parameter with MetadataObject name equal to @Name.</td>
</tr>
<tr>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>-2</td>
<td>A Parameter with MetadataObjectId equal to @Id doesn't exist.</td>
</tr>
<tr>
<td>-6</td>
<td>Parameter with MetadataObjectId equal to @Id has been updated by a context other than the one that it has been currently read by. This happens when the value of @Version does not match with the Version for the Parameter.</td>
</tr>
<tr>
<td>-100</td>
<td>The Method that contains this Parameter already contains another Parameter with Direction of &quot;4&quot;.</td>
</tr>
<tr>
<td>-102</td>
<td>The Parameter with MetadataObjectId equal to @Id cannot be updated to have Direction of &quot;1&quot;. There is a MethodInstance with ReturnTypeDescriptor whose MetadataObjectId is equal to the MetadataObjectId of a TypeDescriptor in the TypeDescriptor tree of root TypeDescriptor of this Parameter.</td>
</tr>
</tbody>
</table>

**Return Code Values:** An integer that the protocol client MUST ignore.

**Result Sets:** MAY &lt;52&gt; return zero or more result sets that the protocol client MUST ignore.

### 3.1.5.94 proc_ar_UpdateSystemById

The `proc_ar_UpdateSystemById` stored procedure is called to change the attributes of the LobSystem identified by the specified MetadataObjectId.

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

```sql
PROCEDURE proc_ar_UpdateSystemById (  
    @Id                          int,  
    @Name                        nvarchar(255),  
    @IsCached                    bit,  
    @Version                     int OUTPUT,  
    @SystemUtilityTypeName       nvarchar(255),  
    @ConnectionManagerTypeName   nvarchar(255),  
    @EntityInstanceTypeName      nvarchar(255),  
    @ErrorCode                   int OUTPUT  
);  
```

**@Id:** The MetadataObjectId of the LobSystem to be updated. The value MUST be an Id, as specified in section 2.2.2.1.

**@Name:** The name of the LobSystem. The value MUST be a Name, as specified in section 2.2.2.2.

**@IsCached:** A bit that specifies whether this LobSystem is frequently used. The value MUST be an IsCached, as specified in section 2.2.2.3.

**@Version:** The value of version at the time LobSystem with the specified MetadataObjectId was last read. This value MUST be incremented in the metadata store every time the LobSystem is updated. The value MUST wrap around after reaching 2147483646.

**@SystemUtilityTypeName:** The name of the system utility to be used to execute the Methods in this LobSystem. The value MUST be a SystemUtilityTypeName, as specified in section 2.2.2.20.

**@ConnectionManagerTypeName:** The name of the connection manager to be used while connecting to this LobSystem. The value MUST be a ConnectionManagerTypeName, as specified in section 2.2.2.19.
@EntityInstanceTypeName: The name of the unit of implementation-specific business logic to be used to create the objects to carry EntityInstance data to client applications. The value MUST be an EntityInstanceTypeName, as specified in section 2.2.2.21.

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-1</td>
<td>A LobSystem with the specified @Name already exists in the metadata store.</td>
</tr>
<tr>
<td>-2</td>
<td>A LobSystem with the specified @Id does not exist.</td>
</tr>
<tr>
<td>-6</td>
<td>A LobSystem with the specified MetadataObjectId has been updated by a context other than the one that it has been currently read by. This happens when the version specified does not match with the current version of the LobSystem.</td>
</tr>
</tbody>
</table>

Positive integer A T-SQL error code.

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MAY return zero or more result sets that the protocol client MUST ignore.

3.1.5.95 proc_ar_UpdateSystemInstanceById

The proc_ar_UpdateSystemInstanceById stored procedure is called to change the attributes of the LobSystemInstance identified by the specified MetadataObjectId.

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

```sql
PROCEDURE proc_ar_UpdateSystemInstanceById (  
    @Id                    int,  
    @Name                  nvarchar(255),  
    @IsCached              bit,  
    @Version               int OUTPUT,  
    @SystemId              int,  
    @ErrorCode             int OUTPUT  
);  
```

@Id: The MetadataObjectId of the LobSystemInstance to be updated. The value MUST be an Id, as specified in section 2.2.2.1.

@Name: The name of the LobSystemInstance. The value MUST be a Name, as specified in section 2.2.2.2.

@IsCached: A bit that specifies whether this LobSystemInstance is frequently used. The value MUST be an IsCached, as specified in section 2.2.2.3.

@Version: The value of version at the time the LobSystemInstance with the specified MetadataObjectId was last read. This value MUST be incremented in the metadata store every time the LobSystemInstance is updated. The value MUST wrap around after reaching 2147483646.
@SystemId: The MetadataObjectId of the LobSystem that contains this LobSystemInstance. The value MUST be a LobSystem that currently exists in the metadata store.

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-1</td>
<td>The LobSystemInstance already contains another Entity with the specified @Name.</td>
</tr>
<tr>
<td>-2</td>
<td>A LobSystemInstance with the specified @Id is not available.</td>
</tr>
<tr>
<td>-3</td>
<td>The LobSystem with @SystemId already contains implementation-specific maximum number of LobSystemInstances.</td>
</tr>
<tr>
<td>-6</td>
<td>A LobSystemInstance with the specified MetadataObjectId has been updated by a context other than the one that it has been currently read by. This happens when the version specified does not match with the current version of the LobSystemInstance.</td>
</tr>
<tr>
<td>A positive integer</td>
<td>A T-SQL error code.</td>
</tr>
</tbody>
</table>

Return Code Values: An integer that the protocol client MUST ignore.

Result Sets: MAY <55> return zero or more result sets that the protocol client MUST ignore.

3.1.5.96 proc_ar_UpdateTypeDescriptorById

The proc_ar_UpdateTypeDescriptorById stored procedure is called to change attributes of TypeDescriptor with the specified MetadataObjectId.

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

```sql
PROCEDURE proc_ar_UpdateTypeDescriptorById (  
    @Id int,  
    @Name nvarchar(255),  
    @IsCached bit,  
    @Version int OUTPUT,  
    @ParentTypeDescriptorId int,  
    @TypeName nvarchar(255),  
    @IdentifierId int,  
    @FilterDescriptorId int,  
    @IsCollection bit,  
    @ErrorCode int OUTPUT,  
    @ContainsIdentifier bit OUTPUT,  
    @ContainsFilterDescriptor bit OUTPUT  
);  
```

@Id: The MetadataObjectId of the TypeDescriptor to be updated. The value MUST be an Id, as specified in section 2.2.2.1.

@Name: The programmatic name of the TypeDescriptor to be updated. The value MUST be a Name, as specified in section 2.2.2.2.
@IsCached: A bit that specifies whether this TypeDescriptor is frequently used. The value MUST be an IsCached, as specified in section 2.2.2.3.

@Version: The value of version at the time the TypeDescriptor with the specified MetadataObjectId was last read. This value MUST be incremented in the metadata store every time the TypeDescriptor is updated. The value MUST wrap around after reaching 2147483646.

@ParentTypeDescriptorId: The MetadataObjectId of the parent TypeDescriptor that contains this TypeDescriptor. If not NULL, The value MUST be a TypeDescriptor that currently exists in the metadata store. The value MUST be an Id.

@TypeName: The programmatic name of the data type that is represented by this TypeDescriptor. The value MUST be a TypeDescriptorTypeName, as specified in section 2.2.2.18.

@IdentifierId: The MetadataObjectId of the Identifier referenced by this TypeDescriptor. The value MUST be an Id.

@FilterDescriptorId: The MetadataObjectId of the FilterDescriptor associated with this TypeDescriptor. The value MUST be an Id.

@IsCollection: A bit that specifies whether this TypeDescriptor is to be interpreted by protocol clients as a collection of native LOB system data structures. The value MUST be an IsCollection, as specified in section 2.2.2.8.

@ErrorCode: The error code. Upon return from this stored procedure, this parameter MUST be set to an integer that is listed in the following table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No errors encountered.</td>
</tr>
<tr>
<td>-1</td>
<td>The TypeDescriptor that contains this TypeDescriptor already contains another TypeDescriptor with MetadataObject name equal to @Name.</td>
</tr>
<tr>
<td>-2</td>
<td>A TypeDescriptor with MetadataObjectId equal to @Id doesn't exist.</td>
</tr>
<tr>
<td>-3</td>
<td>At least one of the following has happened:</td>
</tr>
<tr>
<td></td>
<td>- The TypeDescriptor with MetadataObjectId equal to @ParentTypeDescriptorId already contains implementation-specific maximum number of TypeDescriptors allowed.</td>
</tr>
<tr>
<td></td>
<td>- The FilterDescriptor with MetadataObjectId equal to @FilterDescriptorId already associated with implementation-specific maximum number of TypeDescriptors allowed.</td>
</tr>
<tr>
<td>-6</td>
<td>TypeDescriptor with MetadataObjectId equal to @Id has been updated by a context other than the one that it has been currently read by. This happens when the value of @Version does not match with the version for the TypeDescriptor.</td>
</tr>
<tr>
<td>-300</td>
<td>The Parameter of the TypeDescriptor with MetadataObjectId equal to @Id already has a TypeDescriptor hierarchy deeper than the implementation-specific maximum level allowed.</td>
</tr>
<tr>
<td>-302</td>
<td>The Parameter of the TypeDescriptor with MetadataObjectId equal to @Id already has a root TypeDescriptor.</td>
</tr>
<tr>
<td>-303</td>
<td>The Method that contains FilterDescriptor with MetadataObjectId equal to @FilterDescriptorId does not contain the Parameter of the TypeDescriptor with...</td>
</tr>
<tr>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>----------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>-304</td>
<td>The <code>TypeDescriptor</code> tree of root <code>TypeDescriptor</code> of Parameter that contains <code>TypeDescriptor</code> with <code>MetadataObjectId</code> equal to <code>@Id</code> does not contain the <code>TypeDescriptor</code> with <code>MetadataObjectId</code> equal to <code>@ParentTypeDescriptorId</code>.</td>
</tr>
<tr>
<td>-305</td>
<td>A <code>TypeDescriptor</code> with the <code>IsCollection</code> attribute set to &quot;true&quot; cannot contain another <code>TypeDescriptor</code> with the <code>IsCollection</code> attribute set to &quot;true&quot;.</td>
</tr>
<tr>
<td>-306</td>
<td>A <code>TypeDescriptor</code> with the <code>IsCollection</code> attribute set to &quot;true&quot; cannot contain more than one <code>TypeDescriptor</code>.</td>
</tr>
<tr>
<td></td>
<td>A positive integer</td>
</tr>
</tbody>
</table>

@`ContainsIdentifier`: A `Boolean` value specifying if any `TypeDescriptor` in the `TypeDescriptor` tree of this `TypeDescriptor` references an `Identifier`.

@`ContainsFilterDescriptor`: A `Boolean` value specifying if any `TypeDescriptor` in the `TypeDescriptor` tree of this `TypeDescriptor` has an associated `FilterDescriptor`.

Return Values: An integer that the protocol client MUST ignore.

Result Sets: MAY &lt;56&gt; return zero or more result sets that the protocol client MUST ignore.

### 3.1.6 Timer Events

None.

### 3.1.7 Other Local Events

None.

### 3.2 Metadata Client Details

The metadata client acts as a client when it calls the back-end database server requesting processing of stored procedures and optionally caching some of the data retrieved by the stored procedures.

#### 3.2.1 Abstract Data Model

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

The `MetadataObjects` stored in the metadata store may be maintained as object structures within the protocol client.

The protocol client sends messages to the protocol server to add, retrieve, change, and delete `MetadataObjects` stored in the protocol server.
3.2.1.1 MetadataObject Caching

The metadata client can cache the following sets of data for this protocol within object structures. Data within these structures may not be a complete representation of all data on the back-end database server, but can be populated as various requests to the back-end database server are fulfilled. Data may be cached at two levels independently - the MetadataObjects themselves as well as the relationships between MetadataObjects of different types.

Data maintained in the metadata client can be discarded after individual sequences of requests have completed as part of the cache invalidation mechanism. Cache invalidation can happen independently for objects and relationships. To trigger cache invalidation, the protocol client MUST call proc_ar_BumpCacheInvalidationCounter, as specified in section 3.1.5.3, with the type of the MetadataObject whose cache is to be invalidated along with the type (Object or Relationship) cache to be invalidated.

Note that the preceding conceptual data can be implemented using a variety of techniques. An implementation can implement such data in any way.

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

The Protocol Client handles each stored procedure with the same basic processing method of calling the stored procedure and waiting for the result code and any result sets that will be returned.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.
4 Protocol Examples

This section provides specific example scenarios for operations on stored MetadataObjects. These examples describe in detail the process of communication between the protocol server and protocol client. In conjunction with the detailed client and server protocol specification in section 3, this information is intended to provide a comprehensive view on how the protocol client operates with the protocol server when executing such an operation.

The examples manipulate Entities. However, the principals illustrated apply equally to other MetadataObjects.

4.1 Creating an Entity

This example illustrates how a user can create an Entity in the metadata store.

The example assumes that:

- A **LobSystem** that will contain the **Entity** is already created in the metadata store.
- The **LobSystem** is identified by a **MetadataObjectId** of "33".
- The **LobSystem** has two ACEs associated with it:
  - The first authorizes domain\user1 with **MetadataRights** "Edit", "Execute".
  - The second authorizes domain\user2 with **MetadataRights** "Execute".

The following actions are carried out:

1. The user requests the protocol client to create an **Entity** with name "Customer" and estimated instance count of "100".
2. The protocol client calls the **proc_ar_CreateEntity** stored procedure, as specified in section 3.1.5.9:

   ```
   exec @return_value = proc_ar_CreateEntity
   @Name = 'Customer',
   @IsCached = 1,
   @SystemId = 33,
   @EstimatedInstanceCount = 100,
   @CreatedId OUTPUT,
   @ErrorCode OUTPUT
   ```

3. The protocol server creates the **Entity** in the metadata store. It also copies the ACEs of the **LobSystem** and associates them with the newly created **Entity**. Finally, it sets **@ErrorCode** to zero ("0").
4. The protocol server returns a variable number of result sets that the protocol client ignores.
5. The protocol server returns a return code that the protocol client ignores.
6. The protocol client returns the **@createdId** and **@errorCode** values to the user.
7. The user inspects the **@errorCode** to see if the creation was successful.
8. The user saves the @createdId as the MetadataObjectId of the newly created Entity for subsequent use. Assume the value of @createdId is "34".

4.2 Reading the Security Information of a MetadataObject

This example shows how a user can read the ACEs of an Entity.

The example assumes that the preceding example has been successfully executed.

The following actions are carried out:

1. The user requests the protocol client to read ACEs for the Entity identified by MetadataObjectId "34".

2. The protocol client calls the proc_ar_GetAccessControlEntriesForMetadataObject stored procedure, as specified in section 3.1.5.36.

   ```
   exec @return_value = proc_ar_GetAccessControlEntriesForMetadataObject
   @MetadataObjectId = 34,
   @ErrorCode [int] OUTPUT
   ```

3. The protocol server checks whether a MetadataObject with MetadataObjectId "34" exists in the metadata store.

4. The protocol server retrieves the attributes of each of the two ACEs associated with Entity that were created in the previous example.

5. The protocol server returns an Access Control Entry result set with two rows to the protocol client.

6. The protocol server returns a return code that the protocol client ignores.

7. The user utilizes the ACE information to make an implementation-specific authorization decision.

4.3 Reading an Entity

This example shows how a user can read an Entity in the metadata store.

The example assumes that the preceding example has been successfully executed.

The following actions are carried out:

1. The user requests the protocol client to read Entity with MetadataObjectId equal to "34".

2. The protocol client calls the proc_ar_GetEntityById stored procedure, as specified in section 3.1.5.59.

   ```
   exec @return_value = proc_ar_ReadEntityById
   @MetadataObjectId = 34
   ```

3. The protocol server checks whether an Entity with MetadataObjectId "34" exists in the metadata store.

4. If it exists, the protocol server retrieves the attributes of the stored Entity.
5. The protocol server returns an **Entity** result set with one row to the protocol client. The columns in the row and the values are as follows:

<table>
<thead>
<tr>
<th>Column</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>34</td>
</tr>
<tr>
<td>EstimatedInstanceCount</td>
<td>100</td>
</tr>
<tr>
<td>SystemId</td>
<td>33</td>
</tr>
<tr>
<td>Name</td>
<td>Customer</td>
</tr>
<tr>
<td>IsCached</td>
<td>1</td>
</tr>
<tr>
<td>Version</td>
<td>0</td>
</tr>
</tbody>
</table>

6. The protocol server returns a return code that the protocol client ignores.

7. The user retrieves the **Entity** attributes from the result set.

### 4.4 Updating an Entity

This example shows how a user can update an **Entity** in the metadata store.

The example assumes that the preceding example has been successfully executed.

The following actions are carried out:

1. The user requests the protocol client to update the **Entity** with **MetadataObjectId** equal to "34" and change its name from "Customer" to "Buyer".

2. The protocol client calls the **proc_ar_UpdateEntityById** stored procedure, as specified in section 3.1.5.88. Attributes other than **name** are supplied with the values obtained when the **Entity** was read in the preceding example.

   ```sql
   exec @return_value = proc_ar_UpdateEntityById
       @Id = 34,
       @Name = 'Buyer',
       @IsCached = 1,
       @Version = 0 OUTPUT,
       @SystemId = 33,
       @EstimatedInstanceCount = 10,
       @ErrorCode OUTPUT
   ```

3. The protocol server checks whether an **Entity** with **MetadataObjectId** "34" exists in the metadata store.

4. If it exists, the protocol server compares the value of @Version with the value of the stored version for the **Entity** with **MetadataObjectId** "34". Because they are same, the protocol server updates the all the attribute of the **Entity** with the supplied values, increments the version counter from zero ("0") to "1" and sets the @ErrorCode to zero ("0").

5. The protocol server returns a variable number of result sets that the protocol client ignores.

6. The protocol server returns a return code that the protocol client ignores.

7. The protocol client returns the @errorCode and @version values to the user.

8. The user inspects the @errorCode to see if the update was successful. The user saves the @version value, with value of "1", for use in subsequent updates to the **Entity**.
4.5 Deleting an Entity

This example shows how a user can delete an Entity in the metadata store.

The example assumes that the preceding example has been successfully executed.

The following actions are carried out:

1. The user requests the protocol client to delete the Entity with MetadataObjectId equal to "34".
2. The protocol client calls the proc_ar_DeleteEntityById stored procedure, as specified in section 3.1.5.22.

   ```
   exec @return_value = proc_ar_DeleteEntityById
   @Id = 34,
   @Version = 1,
   @ErrorCode OUTPUT
   ```

3. The protocol server checks whether an Entity with MetadataObjectId "34" exists in the metadata store.
4. If it exists, the protocol server compares the value of @Version with the value of the stored version for the Entity with MetadataObjectId "34". Because they are same, the protocol server deletes the Entity along with the associated properties, localized names and ACEs and sets @ErrorCode to zero ("0").
5. The protocol server returns a variable number of result sets that the protocol client ignores.
6. The protocol server returns a return code that the protocol client ignores.
7. The protocol client returns the @errorCode values to the user.
8. The user inspects the @errorCode to see if the deletion was successful.

4.6 Cache Invalidation

This example shows how a user can invalidate cached metadata objects and relationships after one or more MetadataObjects have been created, updated, or deleted.

The example assumes that the preceding example has been successfully executed. The user wants the Entity named "Buyer", which is currently reflected in any in-memory cached metadata representations that may be maintained by a protocol client but has been deleted from the metadata store, to also be removed from the in-memory representations.

The following actions are carried out:

1. The user requests the protocol client to remove all cached Entities from memory.
2. The protocol client calls the proc_ar_BumpCacheInvalidationCounter stored procedure, as specified in section 3.1.5.3.

   ```
   exec @return_value = proc_ar_BumpCacheInvalidationCounter
   @MetadataObjectType =
   @ObjectCache = 1
   ```
3. The protocol server increments the object cache version stamp for the **Entity**
    **MetadataObjectType**.

4. The protocol server returns a return code that the protocol client ignores.

5. The user requests the protocol client to remove references to all **Entities** that are held by all cached **MetadataObjects**.

6. The protocol client calls the **proc_ar_BumpCacheInvalidationCounter** stored procedure.

   ```sql
   exec @return_value = proc_ar_BumpCacheInvalidationCounter
   @MetadataObjectType =
   @ObjectCache = 0
   ```

7. The protocol server increments the relationship cache version stamp for the **Entity**
    **MetadataObjectType**.

8. The protocol server returns a return code that the protocol client ignores.

   In parallel to the preceding process, a cache invalidation timer job is polling the cache version stamp values in the metadata store periodically. When the timer is signaled, the following actions are carried out:

   1. The protocol client timer event handler calls the **proc_ar_GetCacheInvalidationCountersWithCount** stored procedure.

      ```sql
      exec @return_value = proc_ar_GetCacheInvalidationCounters
      ```

   2. The protocol server retrieves the cache version stamp values for all **MetadataObjectTypes**, along with how many types there are counters for.

      The protocol server returns a **Count** result set with one row to the protocol client.

   3. The protocol server returns a **Cache Version Stamps** result set with as many rows as were indicated in the subsequent step to the protocol client.

   4. The protocol server returns a return code that the protocol client ignores.

   5. The protocol client compares the returned counter values with the values it read when the timer was previously signaled, and finds that the Object Cache Version Stamp and the Relationship Cache version stamp values are different. In response, the protocol client deletes the cached **Entity** references and the cached **Entity MetadataObjects** from memory.
5 Security

5.1 Security Considerations for Implementers

There are no additional security considerations for implementers. Security assumptions of this protocol are documented in section 1.5.

5.2 Index of Security Parameters

None.
6 Appendix A: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include released service packs:

- Microsoft® Office SharePoint® Server 2007
- Microsoft® SharePoint® Server 2010
- Microsoft® SQL Server® 2005
- Microsoft® SQL Server® 2008
- Microsoft® SQL Server® 2008 R2
- Microsoft® SQL Server® 2008 R2 SP1
- Microsoft® SQL Server® 2012
- 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.

<1> Section 2.2.2.18: A data type providing business logic (2) that conforms to the [ECMA-335] specification and can be executed by the .NET Framework.

<2> Section 2.2.2.18: A Business Logic Module that conforms to the [ECMA-335] specification and is understood by the .NET Framework.

<3> Section 2.2.5.13: A data type providing business logic (2) that conforms to the [ECMA-335] specification and can be executed by the .NET Framework.

<4> Section 2.2.5.14: A data type providing business logic (2) that conforms to the [ECMA-335] specification and can be executed by the .NET Framework.

<5> Section 2.2.5.14: A business logic (2) module that conforms to the [ECMA-335] specification and is understood by the .NET Framework.

<6> Section 3.1.1: Office SharePoint Server 2007 can only navigate a relationship.

<7> Section 3.1.5.6: Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

<8> Section 3.1.5.7: Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.
<9> **Section 3.1.5.8:** The current implementation does not verify this restriction. Instead, it verifies that the **Entity** with **MetadataObjectId** equal to @DestinationEntityId serves as a source for less than 1000 **Associations**.

<10> **Section 3.1.5.8:** Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

<11> **Section 3.1.5.9:** Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

<12> **Section 3.1.5.10:** Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

<13> **Section 3.1.5.11:** Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

<14> **Section 3.1.5.12:** Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

<15> **Section 3.1.5.13:** Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

<16> **Section 3.1.5.14:** Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

<17> **Section 3.1.5.15:** A database that is stored on a back-end database server and contains all stored procedures and storage for the MetadataObject types.

<18> **Section 3.1.5.15:** A data type providing business logic (2) that conforms to the [ECMA-335] specification and can be executed by the .NET Framework.

<19> **Section 3.1.5.15:** Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

<20> **Section 3.1.5.16:** Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

<21> **Section 3.1.5.17:** Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

<22> **Section 3.1.5.18:** Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

<23> **Section 3.1.5.19:** Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.
Section 3.1.5.20: Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

Section 3.1.5.21: Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

Section 3.1.5.22: Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

Section 3.1.5.23: Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

Section 3.1.5.24: The current implementation does not update Version numbers for the remaining Identifiers that are contained by the same Entity which contained the deleted Identifier.

Section 3.1.5.25: Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

Section 3.1.5.26: Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

Section 3.1.5.27: Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

Section 3.1.5.28: Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

Section 3.1.5.29: The current implementation does not update Version numbers for the remaining Parameters that are contained by the same Method which contained the deleted Parameter.

Section 3.1.5.30: Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

Section 3.1.5.31: Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

Section 3.1.5.32: Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

Section 3.1.5.33: Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

Section 3.1.5.34: Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

Section 3.1.5.74: A business logic module that conforms to the [ECMA-335] specification and is understood by the .NET Framework.

Section 3.1.5.75: A data type providing business logic (2) that conforms to the [ECMA-335] specification and can be executed by the .NET Framework.
Section 3.1.5.74: The current implementation uses this business logic module only for LobSystems that are physically represented by Web services.

Section 3.1.5.82: Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

Section 3.1.5.84: A business logic (2) module that conforms to the [ECMA-335] specification and is understood by the .NET Framework.

Section 3.1.5.85: Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

Section 3.1.5.86: Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

Section 3.1.5.87: Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

Section 3.1.5.88: Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

Section 3.1.5.89: Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

Section 3.1.5.90: Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

Section 3.1.5.91: Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

Section 3.1.5.92: Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

Section 3.1.5.93: Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

Section 3.1.5.94: A data type providing business logic (2) that conforms to the [ECMA-335] specification and can be executed by the .NET Framework.
<54> Section 3.1.5.94: Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

<55> Section 3.1.5.95: Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.

<56> Section 3.1.5.96: Zero or more result sets are returned that are concerned with implementation-specific locking and data integrity validation functionality that the protocol client MUST ignore.
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 102
  MetadataObject caching 103
  server 30

Access Control Entry result set 60

Action result set (section 2.2.5.1, section 3.1.5.37.1, section 3.1.5.40.2, section 3.1.5.35.61, section 3.1.5.35.39.2, section 3.1.5.35.40.2)

ActionParameter result set (section 2.2.5.2, section 3.1.5.38.1, section 3.1.5.39.2)

Association result set (section 2.2.5.3, section 3.1.5.45.1, section 3.1.5.46.1, section 3.1.5.47.2, section 3.1.5.48.2, section 3.1.5.49.2)

B

Binary structures - overview 20
Bit fields - overview 20

C

Cache invalidation example 107
Cache Version Stamps result set 68
Capability negotiation 12
Change tracking 115

Client
  abstract data model 102
  higher-layer triggered events 103
  initialization 103
  local events 103
  message processing 103
  metadata client details interface 102
  MetadataObject caching 103
  overview 102
  sequencing rules 103
  timer events 103
  timers 103

Common data types
  overview 13

Common fields
  ConnectionManagerTypeName 18
  Direction 18
  EstimatedInstanceCount 13
  FilterDescriptorTypeName 15
  Icon 15
  Id 13
  IdentifierTypeName 16
  Index 15
  IsCached 13
  IsDisplayed 15
  IsOpenedInNewWindow 15
  IsStatic 20
  MetadataObjectType 13
  MetadataRights 20
  MethodInstanceType 17
  Name 13

  Position 15
  SystemUtilityTypeName (section 2.2.2.20, section 2.2.2.21)
  TypeDescriptorTypeName 18
  TypeReflectorTypeName 18
  URI 15
  ConnectionManagerTypeName field 18
  Count result set (section 2.2.5.4, section 3.1.5.39.1, section 3.1.5.40.1, section 3.1.5.43.1, section 3.1.5.44.1, section 3.1.5.47.1, section 3.1.5.49.1, section 3.1.5.50.1, section 3.1.5.51.1, section 3.1.5.52.1, section 3.1.5.53.1, section 3.1.5.54.1, section 3.1.5.55.1, section 3.1.5.56.1, section 3.1.5.57.1, section 3.1.5.58.1, section 3.1.5.59.1, section 3.1.5.60.1, section 3.1.5.61.1, section 3.1.5.62.1, section 3.1.5.63.1, section 3.1.5.64.1, section 3.1.5.65.1, section 3.1.5.66.1, section 3.1.5.67.1, section 3.1.5.68.1, section 3.1.5.69.1, section 3.1.5.70.1, section 3.1.5.71.1, section 3.1.5.72.1, section 3.1.5.73.1, section 3.1.5.74.1, section 3.1.5.75.1, section 3.1.5.76.1, section 3.1.5.77.1, section 3.1.5.78.1, section 3.1.5.79.1)

Creating an Entity example 104

D

Data model - abstract
  client 102
  MetadataObject caching 103
  server 30

Data types
  common 13

Data types - simple
  overview 13

DataClass result set (section 2.2.5.5, section 3.1.5.52.1, section 3.1.5.53.2, section 3.1.5.54.1, section 3.1.5.55.1, section 3.1.5.56.1, section 3.1.5.57.1, section 3.1.5.58.1, section 3.1.5.59.1, section 3.1.5.60.1, section 3.1.5.61.1, section 3.1.5.62.1, section 3.1.5.63.1, section 3.1.5.64.1, section 3.1.5.65.1, section 3.1.5.66.1, section 3.1.5.67.1, section 3.1.5.68.1, section 3.1.5.69.1, section 3.1.5.70.1, section 3.1.5.71.1, section 3.1.5.72.1, section 3.1.5.73.1, section 3.1.5.74.1, section 3.1.5.75.1, section 3.1.5.76.1, section 3.1.5.77.1, section 3.1.5.78.1)

Default Values result set 71
Deleting an Entity example 107
Direction field 18

E

Entity result set (section 2.2.5.6, section 3.1.5.56.2, section 3.1.5.57.2, section 3.1.5.58.2, section 3.1.5.59.1, section 3.1.5.60.1, section 3.1.5.61.1, section 3.1.5.62.1, section 3.1.5.63.1, section 3.1.5.64.1, section 3.1.5.65.1, section 3.1.5.66.1, section 3.1.5.67.1, section 3.1.5.68.1, section 3.1.5.69.1, section 3.1.5.70.1, section 3.1.5.71.1, section 3.1.5.72.1, section 3.1.5.73.1, section 3.1.5.74.1, section 3.1.5.75.1, section 3.1.5.76.1, section 3.1.5.77.1, section 3.1.5.78.1)

EntityId result set 72

EstimatedInstanceCount field 13

Events
  local - client 103
  local - server 102
  timer - client 103
  timer - server 102

Examples
  cache invalidation 107
  creating an Entity 104
  deleting an Entity 107

  overview 104
  reading an Entity 105
  reading the security information of a MetadataObject 105
  updating an Entity 106

[MS-BDCSP] — v20120630
Business Data Catalog Database Protocol Specification

Copyright © 2012 Microsoft Corporation.

Release: July 16, 2012
Fields – common

- ConnectionManagerTypeName (18)
- Direction (18)
- EstimatedInstanceCount (13)
- FilterDescriptorTypeName (15)
- Icon (15)
- Id (13)
- IdentifierTypeName (16)
- Index (15)
- IsCached (13)
- IsDisplayed (15)
- IsOpenedInNewWindow (15)
- IsStatic (20)
- MetadataObjectType (13)
- MetadataRights (20)
- MethodInstanceType (17)
- Name (13)
- Position (15)
- SystemUtilityTypeName (section 2.2.2.20, section 2.2.2.21 (19))
- TypeDescriptorTypeName (18)
- TypeReflectorTypeName (18)
- URL (15)

Fields – vendor-extensible (12)

- FilterDescriptor result set (section 2.2.5.7 (24), section 3.1.5.60.1 (75), section 3.1.5.61.2 (75))
- FilterDescriptorTypeName field (15)
- Flag structures – overview (20)

G

Glossary (9)

H

Higher-layer triggered events

- client (103)
- server (31)

I

- Icon field (15)
- Id field (13)
- Identifier result set (section 2.2.5.8 (24), section 3.1.5.62.1 (75), section 3.1.5.63.2 (76))
- IdentifierTypeName field (16)
- Implementer – security considerations (109)
- Index field (15)
- Index of security parameters (109)
- Informative references (11)
- Initialization
  - client (103)
  - server (31)
- Interfaces - client
  - metadata client details (102)
- Introduction (9)
- IsCached field (13)
- IsOpenedInNewWindow field (15)
- IsStatic field (20)

L

Local events

- client (103)
- server (102)
- Localized Name result set (63)

M

Message processing

- client (103)
- server (31)

Messages

- Access Control Entry result set (60)
- Action result set (20)
- ActionParameter result set (section 2.2.5.2 (21), section 3.1.5.38.1 (61), section 3.1.5.39.2 (62))
- Association result set (section 2.2.5.3 (22), section 3.1.5.45.1 (65), section 3.1.5.46.1 (66), section 3.1.5.47.2 (67), section 3.1.5.49.2 (68))
- binary structures (20)
- bit fields (20)
- Cache Version Stamps result set (68)
- common data types (13)
- Count result set (section 2.2.5.4 (22), section 3.1.5.39.1 (62), section 3.1.5.40.1 (62), section 3.1.5.41.1 (63), section 3.1.5.42.1 (64), section 3.1.5.43.1 (65), section 3.1.5.44.1 (66), section 3.1.5.47.1 (67), section 3.1.5.48.1 (68), section 3.1.5.49.1 (69), section 3.1.5.50.1 (70), section 3.1.5.51.1 (71), section 3.1.5.52.1 (72), section 3.1.5.53.1 (73), section 3.1.5.54.1 (74), section 3.1.5.55.1 (75), section 3.1.5.56.1 (76), section 3.1.5.57.1 (77), section 3.1.5.58.1 (78), section 3.1.5.59.1 (79), section 3.1.5.60.1 (80), section 3.1.5.61.1 (81), section 3.1.5.62.1 (82), section 3.1.5.63.1 (83), section 3.1.5.64.1 (84), section 3.1.5.65.1 (85), section 3.1.5.66.1 (86), section 3.1.5.67.1 (87), section 3.1.5.68.1 (88), section 3.1.5.69.1 (89), section 3.1.5.70.1 (90), section 3.1.5.71.1 (91), section 3.1.5.72.1 (92), section 3.1.5.73.1 (93), section 3.1.5.74.1 (94), section 3.1.5.75.1 (95), section 3.1.5.76.1 (96), section 3.1.5.77.1 (97), section 3.1.5.78.1 (98), section 3.1.5.79.1 (99), section 3.1.5.80.1 (100))
- DataClass result set (section 2.2.5.5 (23), section 3.1.5.52.1 (69), section 3.1.5.53.2 (70))
- Default Values result set (71)
- Entity result set (section 2.2.5.6 (23), section 3.1.5.56.2 (72), section 3.1.5.57.2 (73), section 3.1.5.58.2 (74), section 3.1.5.59.1 (74))
- EntityId result set (72)
- enumerations (13)
- FilterDescriptor result set (section 2.2.5.7 (24), section 3.1.5.60.1 (75), section 3.1.5.61.2 (75))
- flag structures (20)
- Identifier result set (section 2.2.5.8 (24), section 3.1.5.62.1 (75), section 3.1.5.63.2 (76))
- IdentifierTypeName field (16)
- Implementer – security considerations (109)
- Index field (15)
- Index of security parameters (109)
- Informative references (11)
- Initialization
  - client (103)
  - server (31)
- Interfaces - client
  - metadata client details (102)
- Introduction (9)
- IsCached field (13)
- IsOpenedInNewWindow field (15)
- IsStatic field (20)
simple data types 13
System Data result set 82
System Instance result set (section 2.2.5.12 27, section 3.1.5.42.2 64, section 3.1.5.43.2 65, section 3.1.5.75.1 82, section 3.1.5.76.2 83)
System result set (section 2.2.5.13 27, section 3.1.5.44.2 65, section 3.1.5.45.2 82, section 3.1.5.46.2 83, section 3.1.5.47.2 84)
table structures 29
transport 13
TypeDescriptor result set (section 2.2.5.14 28, section 3.1.5.51.2 69, section 3.1.5.72.1 81, section 3.1.5.73.2 84, section 3.1.5.74.2 85)
view structures 29
XML structures 29
Messages - common fields
  ConnectionManager TypeName 18
  Direction 18
  EstimatedInstanceCount 13
  FilterDescriptorTypeName 15
  Icon 15
  Id 13
  IdentifierTypeName 16
  Index 15
  IsCached 13
  IsDisplayed 15
  IsOpenedInNewWindow 15
  IsStatic 20
  MetadataObjectType 13
  MetadataRights 20
  MethodInstanceType 17
  Name 13
  Position 15
  SystemUtilityTypeName (section 2.2.2.20 19, section 2.2.2.21 19)
  TypeDescriptorTypeName 18
  TypeReflectorTypeName 18
  URL 15
Metadata client details interface 102
MetadataObject field 13
MetadataRights field 20
Method result set (section 2.2.5.9 25, section 3.1.5.64.1 76, section 3.1.5.68.2 79)
MethodInstance result set (section 2.2.5.10 25, section 3.1.5.65.1 77, section 3.1.5.66.2 77, section 3.1.5.67.2 78)
MethodInstanceType field 17
Methods
  proc_ar_CreateEntity 38
  proc_ar_CreateFilterDescriptor 39
  proc_ar_CreateIdentifier 40
  proc_ar_CreateMethod 41
  proc_ar_CreateMethodInstance 42
  proc_ar_CreateParameter 43
  proc_ar_CreateSystem 44
  proc_ar_CreateSystemInstance 45
  proc_ar_CreateTypeDescriptor 46
  proc_ar_DeleteActionById 47
  proc_ar_DeleteActionParameterById 48
  proc_ar_DeleteAssociationById 49
  proc_ar_DeleteDefaultValue 50
  proc_ar_DeleteEntityById 50
  proc_ar_DeleteFilterDescriptorById 51
  proc_ar_DeleteIdentifierById 52
  proc_ar_DeleteLocalizedObjectNameForMetadataObjectByLCID 53
  proc_ar_DeleteLocalizedNamesByMetadataObjectId 54
  proc_ar_DeleteMethodParameterById 55
  proc_ar_DeletePropertiesById 56
  proc_ar_DeletePropertyForMetadataObjectId 57
  proc_ar_DeleteSystemById 57
  proc_ar_DeleteSystemInstanceById 58
  proc_ar_DeleteTypeDescriptorById 59
  proc_ar_DeleteValueForTypeDescriptor 60
  proc_ar_GetActionById 61
  proc_ar_GetActionParameterById 61
  proc_ar_GetActionParametersForActionWithCount 62
  proc_ar_GetActionsForEntityWithCount 62
  proc_ar_GetAllLocalizedNamesForMetadataObjectWithCount 63
  proc_ar_GetAllSystemInstancesLikeNameWithCount 64
  proc_ar_GetAllSystemInstancesWithCount 65
  proc_ar_GetAllSystemsWithCount 65
  proc_ar_GetAssociationById 65
  proc_ar_GetAssociationByName 65
  proc_ar_GetAssociationsForDataClassWithCount 66
  proc_ar_GetAssociationsForEntityAndRoleWithCount 66
  proc_ar_GetAssociationsForEntityWithCount 67
  proc_ar_GetAssociationsForMethodWithCount 67
  proc_ar_GetCacheInvalidationCountersWithCount 68
  proc_ar_GetChildTypeDescriptorsForTypeDescriptorWithCount 69
  proc_ar_GetDataClassById 69
  proc_ar_GetDataClassesForSystemWithCount 70
  proc_ar_GetDefaultValuesForTypeDescriptor 70
  proc_ar_GetDependentEntitiesForEntity 71
  proc_ar_GetEntitiesForAssociationAndRoleWithCount 72
  proc_ar_GetEntitiesForSystemLikeNameWithCount 73
Reading the security information of a MetadataObject example 105

References 10

Informative 11

normative 10

Relationship to other protocols 11

Result sets - messages

Access Control Entry 60

Action 20

ActionParameter (section 2.2.5.2 21, section 3.1.5.39.1 61, section 3.1.5.39.2 62)

Association (section 2.2.5.3 22, section 3.1.5.45.1 65, section 3.1.5.46.1 66, section 3.1.5.47.2 66, section 3.1.5.48.2 67, section 3.1.5.49.2 68)

Cache Version Stamps 68

Count (section 2.2.5.4 22, section 3.1.5.39.1 62, section 3.1.5.40.1 62, section 3.1.5.41.1 63, section 3.1.5.42.1 64, section 3.1.5.43.1 64, section 3.1.5.44.1 65, section 3.1.5.45.1 66, section 3.1.5.46.1 66, section 3.1.5.47.1 73, section 3.1.5.48.1 73, section 3.1.5.49.1 74, section 3.1.5.50.1 75, section 3.1.5.51.1 76, section 3.1.5.52.1 77, section 3.1.5.53.1 78, section 3.1.5.54.1 79, section 3.1.5.55.1 80, section 3.1.5.56.1 81, section 3.1.5.57.1 82, section 3.1.5.58.1 83, section 3.1.5.59.1 84)

DataClass (section 2.2.5.5 23, section 3.1.5.52.1 69, section 3.1.5.53.2 70)

Default Values 71

Entity (section 2.2.5.6 23, section 3.1.5.56.2 72, section 3.1.5.57.2 73, section 3.1.5.58.2 74, section 3.1.5.59.1 74)

EntityId 72

FilterDescriptor (section 2.2.5.7 24, section 3.1.5.60.1 75, section 3.1.5.61.2 75)

Identifier (section 2.2.5.8 24, section 3.1.5.62.1 75, section 3.1.5.63.2 76)

Localized Name 63

Method (section 2.2.5.9 25, section 3.1.5.64.1 76, section 3.1.5.68.2 79)
MethodInstance (section 2.2.5.10 25, section 3.1.5.65.1 77, section 3.1.5.66.2 77, section 3.1.5.67.2 78)
Parameter (section 2.2.5.11 26, section 3.1.5.69.1 79, section 3.1.5.70.2 79)
Property 80
System (section 2.2.5.13 27, section 3.1.5.44.2 65, section 3.1.5.73.1 81, section 3.1.5.77.2 84)
System Data 82
System Instance (section 2.2.5.12 27, section 3.1.5.42.2 64, section 3.1.5.43.2 65, section 3.1.5.75.1 82, section 3.1.5.76.2 83, section 3.1.5.77.1 84, section 3.1.5.78.1 84, section 3.1.5.80.2 85)
Result sets - overview 20
Result sets – server
Action (section 3.1.5.37.1 61, section 3.1.5.40.2 62)
S
Security implementer considerations 109
parameter index 109
Sequencing rules client 103
server 31
Server abstract data model 30
Action result set (section 3.1.5.37.1 61, section 3.1.5.40.2 62)
higher-layer triggered events 31
initialization 31
local events 102
message processing 31
proc_ar_AddOrInsertLocalizedNameForMetadataObject method 31
proc_ar_AddOrInsertPropertyForMetadataObject method 31
proc_ar_BumpCacheInvalidationCounter method 32
proc_ar_ClearAccessControlEntriesForMetadataObject method 34
proc_ar_CopyAccessControlEntriesForMetadataObject method 34
proc_ar_CreateAction method 34
proc_ar_CreateActionParameter method 36
proc_ar_CreateAssociation method 37
proc_ar_CreateEntity method 38
proc_ar_CreateFilterDescriptor method 39
proc_ar_CreateIdentifier method 40
proc_ar_CreateMethod method 41
proc_ar_CreateMethodInstanceId method 42
proc_ar_CreateParameter method 43
proc_ar_CreateSystem method 44
proc_ar_CreateSystemInstanceId method 45
proc_ar_CreateTypeDescriptor method 46
proc_ar_DeleteActionById method 47
proc_ar_DeleteActionParameterById method 48
proc_ar_DeleteAssociationById method 49
proc_ar_DeleteDefaultValue method 50
proc_ar_DeleteEntityById method 50
proc_ar_DeleteFilterDescriptorById method 51
proc_ar_DeleteIdentifierById method 52
proc_ar_DeleteLocalizedNameForMetadataObjectById method 53
proc_ar_DeleteLocalizedNamesByMetadataObjectId method 53
proc_ar_DeleteMethodById method 54
proc_ar_DeleteMethodInstanceId method 55
proc_ar_DeleteParameterById method 55
proc_ar_DeletePropertiesById method 56
proc_ar_DeletePropertyForMetadataObjectId method 57
proc_ar_DeleteSystemById method 57
proc_ar_DeleteSystemInstanceId method 58
proc_ar_DeleteTypeDescriptorById method 59
proc_ar_EnsureApplicationRegistryExists method 59
proc_ar_GetAccessControlEntriesForMetadataObject method 60
proc_ar_GetActionById method 61
proc_ar_GetActionParameterById method 61
proc_ar_GetActionParametersForActionWithCount method 62
proc_ar_GetActionsForEntityWithCount method 62
proc_ar_GetAllLocalizedNamesForMetadataObjectWithCount method 63
proc_ar_GetAllSystemInstancesLikeNameWithCount method 63
proc_ar_GetAllSystemInstancesWithCount method 64
proc_ar_GetAllSystemsWithCount method 65
proc_ar_GetAssociationById method 65
proc_ar_GetAssociationByName method 65
proc_ar_GetAssociationsForDataClassWithCount method 66
proc_ar_GetAssociationsForEntityAndRoleWithCount method 67
proc_ar_GetAssociationsForMethodWithCount method 67
proc_ar_GetCacheInvalidationCountersWithCount method 68
proc_ar_GetChildTypeDescriptorsForTypeDescriptorWithCount method 69
proc_ar_GetDataClassById method 69
proc_ar_GetDataClassesForSystemWithCount method 70
proc_ar_GetDefaultValuesForTypeDescriptor method 70
proc_ar_GetDependentEntitiesForEntity method 71
proc_ar_GetEntitiesForAssociationAndRoleWithCount method 72
proc_ar_GetEntitiesForSystemLikeNameWithCount method 73
proc_ar_GetEntitiesForSystemWithCount method 73
proc_ar_GetEntityById method 74
proc_ar_GetFilterDescriptorById method 74
proc_ar_GetFilterDescriptorsForMethodWithCount method 75
proc_ar_GetIdentifierById method 75
proc_ar_GetIdentifiersForEntityWithCount method 76
proc_ar_GetMethodById method 76
proc_ar_GetMethodInstanceById method 77
proc_ar_GetMethodInstancesForDataClassWithCount method 77
proc_ar_GetMethodInstancesForMethodWithCount method 77
proc_ar_GetMethodsForDataClassWithCount method 78
proc_ar_GetParameterById method 79
proc_ar_GetParametersForMethodWithCount method 79
proc_ar_GetPropertiesForMetadataObject method 80
proc_ar_GetRootTypeDescriptorForParameter method 80
proc_ar_GetSystemById method 81
proc_ar_GetSystemDataBySystemName method 81
proc_ar_GetSystemInstanceById method 82
proc_ar_GetSystemInstancesForSystemWithCount method 82
proc_ar_GetSystemsLikeNameWithCount method 83
proc_ar_GetTypeDescriptorById method 84
proc_ar_GetTypeDescriptorsByNameAndParameter method 84
proc_ar_GetTypeDescriptorsForFilterDescriptorWithCount method 85
proc_ar_SetAccessControlEntryForMetadataObject method 85
proc_ar_SetDefaultAction method 86
proc_ar_SetDefaultValuesForTypeDescriptor method 86
proc_ar_SetSystemDataBySystemName method 87
proc_ar_UpdateActionById method 88
proc_ar_UpdateActionParameterById method 89
proc_ar_UpdateAssociationById method 90
proc_ar_UpdateEntityById method 91
proc_ar_UpdateFilterDescriptorById method 92
proc_ar_UpdateIdentifierById method 93
proc_ar_UpdateMethodById method 94
proc_ar_UpdateMethodInstanceById method 95
proc_ar_UpdateParameterById method 97
proc_ar_UpdateSystemById method 98
proc_ar_UpdateSystemInstanceById method 99
proc_ar_UpdateTypeDescriptorById method 100

Table and view 29
XML 29
System Data result set 82
System Instance result set (section 2.2.5.12 27,
section 3.1.5.42.2 64, section 3.1.5.43.2 65,
section 3.1.5.75.1 82, section 3.1.5.76.2 83)
System result set (section 2.2.5.13 27, section 3.1.5.44.2 65, section 3.1.5.73.1 81, section 3.1.5.77.2 84)
SystemUtilityTypeName field (section 2.2.2.20 19, section 2.2.2.21 19)

T
Table structures - overview 29
Timer events
  client 103
  server 102
Timers
  client 103
  server 31
Tracking changes 115
Transport 13
Triggered events - higher-layer
  client 103
  server 31
TypeDescriptor result set (section 2.2.5.14 28,
section 3.1.5.51.2 69, section 3.1.5.72.1 81,
section 3.1.5.79.1 84, section 3.1.5.80.2 85)
TypeDescriptorTypeName field 18
TypeReflectorTypeName field 18

U
Updating an Entity example 106
URL field 15

V
Vendor-extensible fields 12
Versioning 12
View structures - overview 29

X
XML structures 29

[MS-BDCSP] — v20120630
Business Data Catalog Database Protocol Specification
Copyright © 2012 Microsoft Corporation.
Release: July 16, 2012