

[MS-SSAS9]:

SQL Server Analysis Services Protocol Specification

Intellectual Property Rights Notice for Protocol Documentation

- **Copyrights.** This protocol 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 protocols, and may distribute portions of it in your implementations of the protocols 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 protocol documentation.
- **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 protocols. Neither this notice nor Microsoft's delivery of the documentation grants any licenses under those or any other Microsoft patents. However, the protocols may be covered by Microsoft's Open Specification Promise (available here: <http://www.microsoft.com/interop/osp>). If you would prefer a written license, or if the protocols are not covered by the OSP, patent licenses are available by contacting protocol@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.

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. This protocol documentation is 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. A protocol specification does 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.

Revision Summary			
Author	Date	Version	Comments
Microsoft Corporation	April 4, 2008	0.1	Initial Availability
Microsoft Corporation	June 27, 2008	1.0	Revised and edited the technical content
Microsoft Corporation	October 6, 2008	1.01	Revised and edited the technical content
Microsoft Corporation	December 12, 2008	1.02	Revised and edited the technical content

Table of Contents

1	Introduction.....	5
1.1	Glossary	5
1.2	References.....	6
1.2.1	Normative References	6
1.2.2	Informative References.....	8
1.3	Protocol Overview (Synopsis)	8
1.4	Relationship to Other Protocols.....	9
1.5	Prerequisites/Preconditions	11
1.6	Applicability Statement	11
1.7	Versioning and Capability Negotiation.....	11
1.8	Vendor-Extensible Fields.....	11
1.9	Standards Assignments	11
2	Messages.....	11
2.1	Transport.....	11
2.2	Common Message Syntax	12
2.2.1	Namespaces	12
2.2.2	Messages.....	12
2.2.3	Elements	12
2.2.4	Complex Types	12
2.2.5	Simple Types.....	38
2.2.6	Attributes	38
2.2.7	Groups	38
2.2.8	Attribute Groups.....	39
2.3	Transport Specific Message Details	39
2.3.1	TCP.....	39
2.3.2	HTTP/HTTPS	42
2.3.3	Encryption.....	43
2.3.4	Compression	44
2.3.5	Binary XML.....	44
3	Protocol Details.....	46
3.1	Server Details	46
3.1.1	Abstract Data Model.....	46
3.1.2	Timers.....	46
3.1.3	Initialization	46
3.1.4	Message Processing Events and Sequencing Rules.....	48
3.1.5	Timer Events	141
3.1.6	Other Local Events	142
3.2	Transport Specific Protocol Details	143

3.2.1	Connection	143
3.2.2	Authentication and Encryption.....	143
3.2.3	Content Type Negotiation.....	143
3.2.4	Generating and Parsing Messages	144
3.2.5	Compression	145
4	<i>Protocol Examples.....</i>	<i>147</i>
4.1	Client obtains a list of databases from the server over TCP	147
4.1.1	Connection	147
4.1.2	Authentication	147
4.1.3	New Session Request.....	155
4.1.4	Request for List of Catalogs	159
4.1.5	End of Session.....	165
4.2	Client obtains a list of cubes from the server over HTTP	169
4.2.1	Connection	169
4.2.2	New Session Request.....	169
4.2.3	Request for List of Cubes.....	170
4.2.4	End of Session.....	171
5	<i>Security.....</i>	<i>173</i>
6	<i>Appendix A: Product Behavior.....</i>	<i>173</i>
	<i>Index.....</i>	<i>178</i>

1 Introduction

The SQL Server Analysis Services Protocol [MS-SSAS9] provides methods for a client to communicate with, and perform operations on, an **OLAP** server.

1.1 Glossary

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

- flag**
- globally unique identifier (GUID)**
- handshake**
- object**
- padding**
- security token**
- session**
- SOAP Action**
- SOAP Body**
- SOAP Fault**
- SOAP Fault code**
- SOAP Header**
- SOAP message**
- WSDL message**
- uniform resource locator (URL)**
- XML**
- XML Schema**
- XML namespace**

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

- complex type**
- cube**
- data block**
- data definition language (DDL)**
- dataset**
- dimension**
- element**
- hierarchy**
- key performance indicator (KPI)**
- measure**
- measure group**
- MIME**
- member**
- multidimensional expression (MDX)**

online analytical processing (OLAP)
perspective
ragged hierarchy
record
simple type
 slicer axis
tuple

The following terms are specific to this document:

role-playing dimension: A single database **dimension** joined to the fact table on a different foreign key to produce multiple **cube dimensions**.

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

1.2 References

1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information. Please check the archive site, <http://msdn.microsoft.com/en-us/library/cc136647.aspx>, as an additional source.

[DIME] Nielsen, H. F., Sanders, H., and Christensen, E., "Direct Internet Message Encapsulation (DIME)", February 2002, <http://xml.coverpages.org/draft-nielsen-dime-01.txt>.

[MS-BINXML] Microsoft Corporation, "[SQL Server Binary XML Structure Specification](#)", June 2008.

[MS-GLOS] Microsoft Corporation, "[Windows Protocols Master Glossary](#)", June 2008.

[MS-LCID] Microsoft Corporation, "[Windows Language Code Identifier \(LCID\) Reference](#)", March 2008.

[MS-OFSGLOS] Microsoft Corporation, "[Microsoft Office Server Master Glossary](#)", June 2008.

[RFC793] Postel, J., "Transmission Control Protocol: DARPA Internet Program Protocol Specification", RFC 793, September 1981, <http://www.ietf.org/rfc/rfc0793.txt>.

- [RFC2279] Yergeau, F., "UTF-8, A Transformation Format of ISO10646", RFC 2279, January 1998, <http://www.ietf.org/rfc/rfc2279.txt>.
- [RFC2396] Berners-Lee, T., Fielding, R., and Masinter, L., "Uniform Resource Identifiers (URI): Generic Syntax", RFC 2396, August 1998, <http://www.ietf.org/rfc/rfc2396.txt>.
- [RFC2616] Fielding, R., et al., "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999, <http://www.ietf.org/rfc/rfc2616.txt>.
- [RFC2743] Linn, J., "Generic Security Service Application Program Interface Version 2, Update 1", RFC 2743, January 2000, <http://www.ietf.org/rfc/rfc2743.txt>.
- [RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, May 2000, <http://www.ietf.org/rfc/rfc2818.txt>.
- [RFC4178] Zhu, L., Leach, P., Jaganathan, K., and Ingersoll, W., "The Simple and Protected Generic Security Service Application Program Interface (GSS-API) Negotiation Mechanism", RFC 4178, October 2005, <http://www.ietf.org/rfc/rfc4178.txt>.
- [SOAP1.1] Box, D., Ehnebuske, D., Kakivaya, G., Layman, A., Mendelsohn, N., Nielsen, H. F., Thatté, S., and Winer, D., "Simple Object Access Protocol (SOAP) 1.1", May 2000, <http://www.w3.org/TR/2000/NOTE-SOAP-20000508/>.
- [SOAP1.2/1] Gudgin, M., Hadley, M., Mendelsohn, N., Moreau, J., and Nielsen, H.F., "SOAP Version 1.2 Part 1: Messaging Framework", W3C Recommendation, June 2003, <http://www.w3.org/TR/2003/REC-soap12-part1-20030624>.
- [SOAP1.2/2] Gudgin, M., Hadley, M., Mendelsohn, N., Moreau, J., and Nielsen, H.F., "SOAP Version 1.2 Part 2: Adjuncts", W3C Recommendation, June 2003, <http://www.w3.org/TR/2003/REC-soap12-part2-20030624>.
- [WSDL] Christensen, E., Curbera, F., Meredith, G., and Weerawarana, S., "Web Services Description Language (WSDL) 1.1", W3C Note, March 2001, <http://www.w3.org/TR/2001/NOTE-wsdl-20010315>.
- [XML10] World Wide Web Consortium, "Extensible Markup Language (XML) 1.0 (Third Edition)", February 2004, <http://www.w3.org/TR/REC-xml>.
- [XMLNS] World Wide Web Consortium, "Namespaces in XML 1.0 (Second Edition)", August 2006, <http://www.w3.org/TR/REC-xml-names/>.
- [XMLSCHEMA1] Thompson, H.S., Ed., Beech, D., Ed., Maloney, M., Ed., and Mendelsohn, N., Ed., "XML Schema Part 1: Structures", W3C Recommendation, May 2001, <http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/>.

[XMLSCHEMA2] Biron, P.V., Ed. and Malhotra, A., Ed., "XML Schema Part 2: Datatypes", W3C Recommendation, May 2001, <http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/>.

[RFC2119] Bradner, S., "Key Words for Use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, <http://www.ietf.org/rfc/rfc2119.txt>.

1.2.2 Informative References

[MSDN-MDPROP] Microsoft Corporation, "MDPROP_Flattening_Support", <http://msdn.microsoft.com/en-us/library/ms720900.aspx>.

[MSDN-SSAS] Microsoft Corporation, "Analysis Services Concepts and Objects", <http://msdn.microsoft.com/en-us/library/ms174578.aspx>.

[MSDN-SSPI] Microsoft Corporation, "Security Support Provider Interface", [http://msdn.microsoft.com/en-us/library/aa378663\(VS.85\).aspx](http://msdn.microsoft.com/en-us/library/aa378663(VS.85).aspx).

[MSDN-SQLXML] Microsoft Corporation, "SQLXML", <http://msdn.microsoft.com/en-us/library/aa286527.aspx>.

[XMLA] Microsoft Corporation, Hyperion Solutions Corporation, SAS Institute Inc., "XML for Analysis", 2008, <http://www.xmla.org/xmla1.1.pdf>.

1.3 Protocol Overview (Synopsis)

The SQL Server Analysis Services Protocol [MS-SSAS9] provides methods for a client to communicate with, and perform operations on an [online analytical processing \(OLAP\)](#) server. This protocol is based on Simple Object Access Protocol (SOAP) and XML for Analysis (XMLA) [\[XMLA\]](#). This protocol supports TCP/IP as an underlying transport mechanism in addition to HTTP/HTTPS.

The SQL Server Analysis Services Protocol defines these operations: Authenticate, Discover, and Execute.

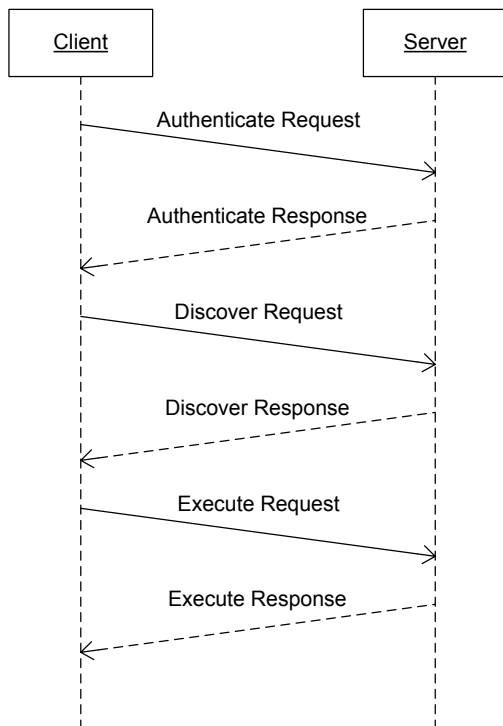
Authenticate is used by the client and server to exchange UTF-8 ([\[RFC2279\]](#)) encoded [security token data blocks](#) as part of the authentication process. For more information on authentication, see section [Authentication](#).

Discover is used to obtain information from an OLAP server, such as a list of catalogs on a server. Properties are used to control what data is obtained. This generic interface and the use of properties allow extensibility without rewriting existing functions. For more information, see section [Discover](#).

Execute is used to execute [multidimensional expressions \(MDX\)](#) or other server-specific commands against a particular OLAP server and get back a result set either in a tabular or multidimensional form. For more information, see section [Execute](#).

By using the Authenticate, Discover, and Execute operations, the transfer of data between a client and OLAP server can be achieved.

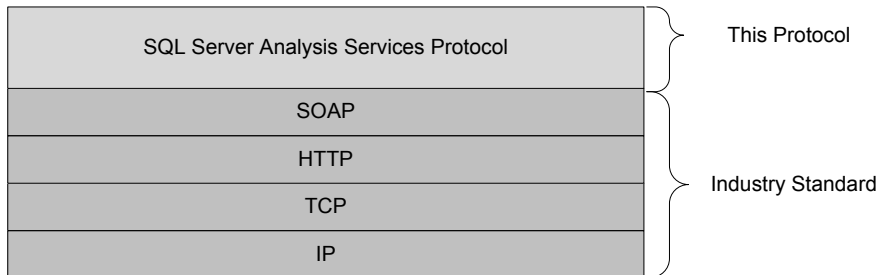
The following diagram illustrates this concept:



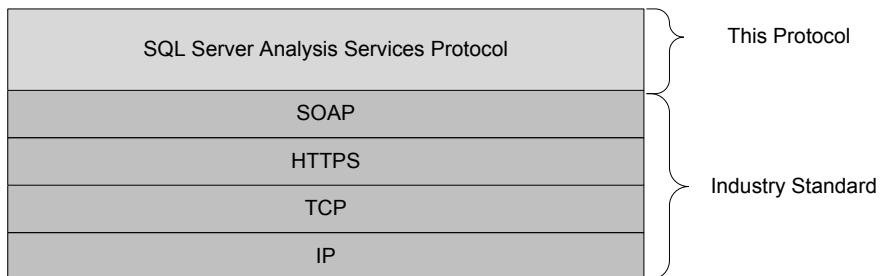
1.4 Relationship to Other Protocols

The SQL Server Analysis Services protocol uses the SOAP messaging protocol for formatting requests and responses as specified either in [\[SOAP1.1\]](#) or in [\[SOAP1.2/1\]](#) and [\[SOAP1.2/2\]](#). It transmits these messages using HTTP [\[RFC2616\]](#), HTTPS [\[RFC2818\]](#), or TCP [\[RFC793\]](#).

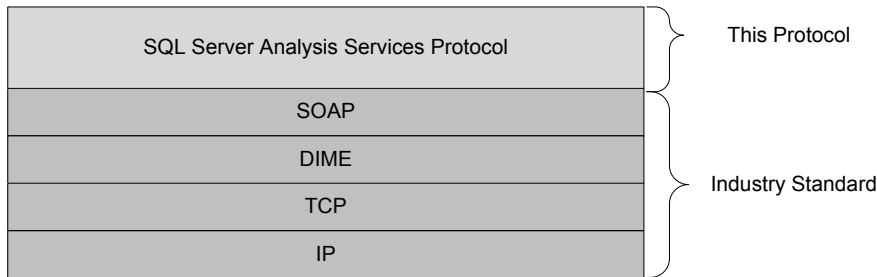
The SQL Server Analysis Services protocol uses SOAP over HTTP as shown in the following layering diagram:



The SQL Server Analysis Services protocol uses SOAP over HTTPS as shown in the following diagram:



The SQL Server Analysis Services protocol uses SOAP over Direct Internet Message Encapsulation (DIME) [\[DIME\]](#) and TCP/IP as shown in the following diagram:



1.5 Prerequisites/Preconditions

None.

1.6 Applicability Statement

This protocol supports the exchange of messages between a client and an OLAP server.

1.7 Versioning and Capability Negotiation

This protocol does not support the use of versioning. This protocol does explicit negotiation as specified below in section [Content Type Negotiation](#).

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

The conversation between a client and a server **MUST** be performed either over TCP or HTTP/HTTPS with a number of layers added on top of them. The message format is a clear text XML [\[XML10\]](#) or binary XML [\[MS-BINXML\]](#) that **MAY** be compressed. By the choice of the client, the message **MAY** also be encrypted using GSS-API [\[RFC4178\]](#) if the conversation is performed over TCP and using SSL in case of HTTPS<[1](#)><[2](#)>. In addition, DIME [\[DIME\]](#) is used for messages transmitted using TCP, and all data transferred between client and server is encoded using UTF-8 [\[RFC2279\]](#). Section [Common Message Syntax](#) specifies the [SOAP Message](#) syntax, regardless of the underlying transport. Section [Transport-Specific Message Details](#) describes transport specific message details.

2.2 Common Message Syntax

This section contains common definitions used by this protocol. The syntax of the definitions uses [XML Schema](#) as defined in [\[XMLSCHEMA1\]](#) and [\[XMLSCHEMA2\]](#), and Web Services Description Language as defined in [\[WSDL\]](#).

2.2.1 Namespaces

This specification defines and references various [XML namespaces](#) using the mechanisms specified in [\[XMLNS\]](#). Although this specification associates a specific XML namespace prefix for each XML namespace that is used, the choice of any particular XML namespace prefix is implementation-specific and not significant for interoperability.

Prefix	Namespace URI	Reference
xsd	http://www.w3.org/2001/XMLSchema	[XMLSCHEMA1] [XMLSCHEMA2]
xsi	http://www.w3.org/2001/XMLSchema-instance	[XMLSCHEMA1] [XMLSCHEMA2]
sql	urn:schemas-microsoft-com:xml-sql	[MSDN-SQLXML]
xmldb	urn:schemas-microsoft-com:xml-analysis	[XMLA]
xmldb-ds	urn:schemas-microsoft-com:xml-analysis:mdataset	[XMLA]
xmldb-rs	urn:schemas-microsoft-com:xml-analysis:rowset	[XMLA]
xmldb-e	urn:schemas-microsoft-com:xml-analysis:empty	[XMLA]
xmldb-x	urn:schemas-microsoft-com:xml-analysis:exception	[XMLA]

2.2.2 Messages

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

2.2.3 Elements

This specification does not define any common XML schema [element](#) definitions.

2.2.4 Complex Types

The following table summarizes the set of common XML schema [complex type](#) definitions defined by this specification. XML schema complex type definitions that are specific to a particular operation are described with the operation.

Complex Type	Description
xmla-ds:root	Describes data from a cube in a representation consisting of information about the origin of the data, the axes of the cube and the data cells.
xmla-e:root	Describes a response to an exception that has occurred at the server during the process of the client request and no further data is available to the client.
xmla-rs:root	Describes data from a cube in a tabular representation.
xmla-x:messages	Describes the structure of error and warning messages sent by the server to the client.

2.2.4.1 xmla-ds:root complex type

Describes the structure of response received from the Execute xmla-ds:root complex type.

The xmla-ds:root complex type describes data from a cube in a representation consisting of information about the origin of the data, the axes of the cube and the data cells.

The xmla-ds:root complex type consists of these main sections:

Schema	Contains the complete schema definition for this message, particular to the data being sent.
OlapInfo	Describes the structure of the current result set.
Axes	Contains the data for the axes as defined in the OlapInfo structure
CellData	Contains the data for the cube cells as defined in the OlapInfo structure

```
<xsd:element name="root">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="schema" namespace="http://www.w3.org/2001/XMLSchema" target
namespace="http://www.w3.org/2001/XMLSchema" processContents="strict" minOccurs="1" />
      <xsd:element name="OlapInfo" type="xmla-ds:OlapInfo" minOccurs="0" />
      <xsd:element name="Axes" type="xmla-ds:Axes" minOccurs="0" />
      <xsd:element name="CellData" type="xmla-ds:CellData" minOccurs="0" />
      <xsd:element name="Exception" type="xmla-x:exception" minOccurs="0" />
      <xsd:element name="Messages" type="xmla-x:Messages" minOccurs="0" />
    
```

13 of 178

```

    </xsd:sequence>
  </xsd:complexType>
</xsd:element>

```

2.2.4.1.1 Schema sub-element

The Schema sub-element contains the complete description of the structure of the data being sent within this Root element. By reading the schema sub-element the client can fully parse the response received. This element **MUST** come with any response that contains data for the client. The schema only describes the data parts of the response; messages or alerts are not documented by the Schema element.

The Schema sub-element describes the elements and types, starting with the most elemental one and progressing to the most complex. Any element or data type that depends on another defined element or data type **MUST** appear after the one or ones that are part of its definition. In that order of ideas, the root element itself would be the last one to appear in the schema sub-element, preceded by the *OlapInfo*, *Axes*, and *CellData* element or data type definition, and so on.

Reading the following sample from the bottom up makes it easier to find the place where the schema definition for the *xmla-ds:root* element appears. This definition contains the three data elements mentioned in [xmla-ds:root complex type](#): *OlapInfo*, *Axes*, and *CellData*.

The following sample describes the schema sent as part of a response to an [MDX](#) Statement Command request by the client:

```

<root xmlns="urn:schemas-microsoft-com:xml-analysis:mddataset"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:schema targetNamespace="urn:schemas-microsoft-com:xml-analysis:mddataset"
elementFormDefault="qualified" xmlns="urn:schemas-microsoft-com:xml-analysis:mddataset"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xsd:complexType name="MemberType">
      <xsd:sequence>
        <xsd:any namespace="##targetNamespace" minOccurs="0" maxOccurs="unbounded"
processContents="skip" />
      </xsd:sequence>
      <xsd:attribute name="Hierarchy" type="xsd:string" />
    </xsd:complexType>
    <xsd:complexType name="PropType">
      <xsd:sequence>
        <xsd:element name="Default" minOccurs="0" />
      </xsd:sequence>
      <xsd:attribute name="name" type="xsd:string" use="required" />
      <xsd:attribute name="type" type="xsd:QName" />
    </xsd:complexType>
    <xsd:complexType name="TupleType">
      <xsd:sequence>

```

```

        <xsd:element name="Member" type="MemberType" maxOccurs="unbounded" />
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="MembersType">
    <xsd:sequence>
        <xsd:element name="Member" type="MemberType" minOccurs="0" maxOccurs="unbounded"
/>
    </xsd:sequence>
    <xsd:attribute name="Hierarchy" type="xsd:string" use="required" />
</xsd:complexType>
<xsd:complexType name="TuplesType">
    <xsd:sequence>
        <xsd:element name="Tuple" type="TupleType" minOccurs="0" maxOccurs="unbounded" />
    </xsd:sequence>
</xsd:complexType>
<xsd:group name="SetType">
    <xsd:choice>
        <xsd:element name="Members" type="MembersType" />
        <xsd:element name="Tuples" type="TuplesType" />
        <xsd:element name="CrossProduct" type="SetListType" />
        <xsd:element name="Union">
            <xsd:complexType>
                <xsd:group ref="SetType" minOccurs="0" maxOccurs="unbounded" />
            </xsd:complexType>
        </xsd:element>
    </xsd:choice>
</xsd:group>
<xsd:complexType name="SetListType">
    <xsd:group ref="SetType" minOccurs="0" maxOccurs="unbounded" />
    <xsd:attribute name="Size" type="xsd:unsignedInt" />
</xsd:complexType>
<xsd:complexType name="OlapInfo">
    <xsd:sequence>
        <xsd:element name="CubeInfo">
            <xsd:complexType>
                <xsd:sequence>
                    <xsd:element name="Cube" maxOccurs="unbounded">
                        <xsd:complexType>
                            <xsd:sequence>
                                <xsd:element name="CubeName" type="xsd:string" />
                                <xsd:element name="LastDataUpdate" minOccurs="0" type="xsd:dateTime"
/>
                                <xsd:element name="LastSchemaUpdate" minOccurs="0"
type="xsd:dateTime" />
                            </xsd:sequence>
                        </xsd:complexType>
                    </xsd:element>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
    </xsd:sequence>
</xsd:complexType>

```

```

<xsd:element name="AxesInfo">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="AxisInfo" maxOccurs="unbounded">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name="HierarchyInfo" minOccurs="0"
maxOccurs="unbounded">
              <xsd:complexType>
                <xsd:sequence>
                  <xsd:any namespace="##targetNamespace" minOccurs="0"
maxOccurs="unbounded" processContents="skip" />
                </xsd:sequence>
                <xsd:attribute name="name" type="xsd:string" use="required" />
              </xsd:complexType>
            </xsd:element>
          </xsd:sequence>
          <xsd:attribute name="name" type="xsd:string" />
        </xsd:complexType>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
<xsd:element name="CellInfo">
  <xsd:complexType>
    <xsd:choice minOccurs="0" maxOccurs="unbounded">
      <xsd:any namespace="##targetNamespace" minOccurs="0" maxOccurs="unbounded"
processContents="skip" />
    </xsd:choice>
  </xsd:complexType>
</xsd:element>
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="Axes">
  <xsd:sequence>
    <xsd:element name="Axis" maxOccurs="unbounded">
      <xsd:complexType>
        <xsd:group ref="SetType" minOccurs="0" maxOccurs="unbounded" />
        <xsd:attribute name="name" type="xsd:string" />
      </xsd:complexType>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="CellData">
  <xsd:sequence>
    <xsd:element name="Cell" minOccurs="0" maxOccurs="unbounded">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:any namespace="##targetNamespace" minOccurs="0" maxOccurs="unbounded"
processContents="skip" />
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>

```

```

        </xsd:sequence>
        <xsd:attribute name="CellOrdinal" type="xsd:unsignedInt" use="required" />
    </xsd:complexType>
</xsd:element>
</xsd:sequence>
</xsd:complexType>
<xsd:element name="root">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:any namespace="http://www.w3.org/2001/XMLSchema" processContents="strict"
minOccurs="0" />
            <xsd:element name="OlapInfo" type="OlapInfo" minOccurs="0" />
            <xsd:element name="Axes" type="Axes" minOccurs="0" />
            <xsd:element name="CellData" type="CellData" minOccurs="0" />
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>
</xsd:schema>
...
</root>

```

2.2.4.1.2 *OlapInfo sub-element*

The OlapInfo sub-element contains information describing the multidimensional data being sent in this response. It **MUST** contain information about the cube or cubes from which these results were extracted, information about all axes involved and the cells returned.

The OlapInfo sub-element defines the schema for the incoming data; it explicitly defines the structure for the Axes sub-element in terms of the incoming data and the CellData sub-element.

The following samples are an introduction to the different sub-elements that compose the OlapInfo sub-element. For more information about the OlapInfo data type, see [xmlds:OlapInfo complex type](#).

The OlapInfo sub-element **MUST** contain the following child elements:

- CubeInfo, which specifies the sources of the result set
- AxesInfo, which specifies the structure of each of the axis of the cube including the [slicer axis](#)
- CellInfo, which specifies the structure of the cells

```

...
<OlapInfo>

```

```

...
<CubeInfo>
  <Cube>
    <CubeName>Adventure Works</CubeName>
    <LastDataUpdate
xmlns="http://schemas.microsoft.com/analysiservices/2003/engine">2008-02-
11T19:34:28</LastDataUpdate>
    <LastSchemaUpdate
xmlns="http://schemas.microsoft.com/analysiservices/2003/engine">2008-02-
11T19:33:10</LastSchemaUpdate>
  </Cube>
</CubeInfo>
...
</OlapInfo>
...

```

The following text shows how the OlapInfo sub-element describes the structure of one of the axis in terms of the specific data of the response:

```

...
<OlapInfo>
  ...
  <AxesInfo>
    <AxisInfo name="Axis0">
      <HierarchyInfo name="[Measures]">
        <UName name="[Measures].[MEMBER_UNIQUE_NAME]" type="xsd:string" />
        <Caption name="[Measures].[MEMBER_CAPTION]" type="xsd:string" />
        <LName name="[Measures].[LEVEL_UNIQUE_NAME]" type="xsd:string" />
        <LNum name="[Measures].[LEVEL_NUMBER]" type="xsd:int" />
        <DisplayInfo name="[Measures].[DISPLAY_INFO]" type="xsd:unsignedInt" />
      </HierarchyInfo>
    </AxisInfo>
  </AxesInfo>
  ...
</OlapInfo>
...

```

The following text shows how the OlapInfo sub-element describes the structure of the cells in terms of the specific data of the response. Observe how the AxesInfo element describes the existence of an Axis element named 'Axis0' through the AxisInfo element; within the AxisInfo element only one HierarchyInfo element is described with the name 'Measures'. Finally, the HierarchyInfo element is described as composed of several elements that reflect the properties of the [hierarchy member](#) to be returned. For each of the child elements a name and type are given; the name is the property name of the hierarchy member and the type is the type of the hierarchy member property.

...

```

<OlapInfo>
  ...
  <CellInfo>
    <Value name="VALUE" />
    <FmtValue name="FORMATTED_VALUE" type="xsd:string" />
    <CellOrdinal name="CELL_ORDINAL" type="xsd:unsignedInt" />
  </CellInfo>
</OlapInfo>
...

```

2.2.4.1.2.1 xmla-ds:OlapInfo complex type

The xmla-ds:OlapInfo complex type describes the current schema definition for the result set requested by the client. The schema element, described in section [Schema sub-element](#), describes the general syntax to parse any response. The OlapInfo complex type contains the description of the specific structure of the data being returned. The result set is described in three major sections:

- CubeInfo: Describes the source of the result set.
- AxesInfo: Describes the data structure of all axes in the result set, including the slicer axis.
- CellInfo Describes the data structure for the cells.

The OlapInfo data type MUST contain all three elements: CubeInfo, AxesInfo, and CellInfo.

```

<xsd:complexType name="OlapInfo">
  <xsd:sequence>
    <xsd:element name="CubeInfo" type="CubeInfo" />
    <xsd:element name="AxesInfo" type="AxesInfo" />
    <xsd:element name="CellInfo" type="CellInfo" />
  </xsd:sequence>
</xsd:complexType>

```

2.2.4.1.2.1.1 CubeInfo element

The CubeInfo element describes all sources involved in providing the current result set. The CubeInfo element is a collection of cube elements, each of them containing the following information about the cube: the cube name, the last time data was updated in the cube, and the last time the cube definition was updated.

```

<xsd:complexType name="CubeInfo">
  <xsd:sequence>
    <xsd:element name="Cube" maxOccurs="unbounded">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="CubeName" type="xsd:string" />

```

```

        <xsd:element name="LastDataUpdate" minOccurs="0" type="xsd:dateTime" />
        <xsd:element name="LastSchemaUpdate" minOccurs="0" type="xsd:dateTime" />
    </xsd:sequence>
</xsd:complexType>
</xsd:element>
</xsd:sequence>
</xsd:complexType>

```

2.2.4.1.2.1.2 AxesInfo element

The AxesInfo element describes a collection of AxisInfo sub-elements. The AxesInfo element contains one AxisInfo element per each axis in the result set, including the slicer axis of the result set.

```

<xsd:complexType name="AxesInfo">
    <xsd:sequence>
        <xsd:element name="AxisInfo" maxOccurs="unbounded">
            <xsd:complexType>
                <xsd:sequence>
                    <xsd:element name="HierarchyInfo" minOccurs="0" maxOccurs="unbounded">
                        <xsd:complexType>
                            <xsd:sequence>
                                <xsd:any namespace="##targetNamespace" minOccurs="0"
maxOccurs="unbounded" processContents="skip" />
                            </xsd:sequence>
                                <xsd:attribute name="name" type="xsd:string" use="required" />
                            </xsd:complexType>
                        </xsd:element>
                    </xsd:sequence>
                    <xsd:attribute name="name" type="xsd:string" />
                </xsd:complexType>
            </xsd:element>
        </xsd:sequence>
    </xsd:complexType>
</xsd:complexType>

```

The AxisInfo sub-element describes the axis hierarchy in terms of the member elements that constitute the axis point. The AxisInfo sub-element contains one HierarchyInfo element for each member that is a part of the axis hierarchy. The HierarchyInfo sub-element is a collection of properties and values that describe the member at this point in the axis.

As an example, for this MDX query:

```

<?xml version="1.0" encoding="utf-8"?>
<Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
    <Body>
        <Execute xmlns="urn:schemas-microsoft-com:xml-analysis">
            <Command>
                <Statement>

```

```

        Select nonemptycrossjoin([Geography].[Country].members,
[Customer].[Gender].members) on 0,
        [Measures].members on 1
        from [Adventure Works]
    </Statement>
</Command>
<Properties>
    <PropertyList>
        <Catalog>Adventure Works DW</Catalog>
    </PropertyList>
</Properties>
</Execute>
</Body>
</Envelope>

```

The following text shows parts the AxesInfo returned and the corresponding Axes element described by this information. Note the matching values of the name attribute of the HierarchyInfo element and the Hierarchy attribute in the Member element. Also, notice how the elements UName, Caption, LName, LNum, and DisplayInfo are defined within the HierarchyInfo element, and later, in the Member element of the corresponding hierarchy, are used to return the actual values.

```

...
<OlapInfo>
    ...
    <AxesInfo>
        <AxisInfo name="Axis0">
            <HierarchyInfo name="[Geography].[Country]">
                <UName name="[Geography].[Country].[MEMBER_UNIQUE_NAME]" type="xsd:string" />
                <Caption name="[Geography].[Country].[MEMBER_CAPTION]" type="xsd:string" />
                <LName name="[Geography].[Country].[LEVEL_UNIQUE_NAME]" type="xsd:string" />
                <LNum name="[Geography].[Country].[LEVEL_NUMBER]" type="xsd:int" />
                <DisplayInfo name="[Geography].[Country].[DISPLAY_INFO]" type="xsd:unsignedInt" />
            />
        </HierarchyInfo>
        <HierarchyInfo name="[Customer].[Gender]">
            <UName name="[Customer].[Gender].[MEMBER_UNIQUE_NAME]" type="xsd:string" />
            <Caption name="[Customer].[Gender].[MEMBER_CAPTION]" type="xsd:string" />
            <LName name="[Customer].[Gender].[LEVEL_UNIQUE_NAME]" type="xsd:string" />
            <LNum name="[Customer].[Gender].[LEVEL_NUMBER]" type="xsd:int" />
            <DisplayInfo name="[Customer].[Gender].[DISPLAY_INFO]" type="xsd:unsignedInt" />
        </HierarchyInfo>
    </AxisInfo>
    <AxisInfo name="Axis1">
        <HierarchyInfo name="[Measures]">
            <UName name="[Measures].[MEMBER_UNIQUE_NAME]" type="xsd:string" />
            <Caption name="[Measures].[MEMBER_CAPTION]" type="xsd:string" />
            <LName name="[Measures].[LEVEL_UNIQUE_NAME]" type="xsd:string" />
            <LNum name="[Measures].[LEVEL_NUMBER]" type="xsd:int" />
        </HierarchyInfo>
    </AxisInfo>

```

```

        <DisplayInfo name="[Measures].[DISPLAY_INFO]" type="xsd:unsignedInt" />
    </HierarchyInfo>
</AxisInfo>
<AxisInfo name="SlicerAxis">
...
    <Axes>
        <Axis name="Axis0">
            <Tuples>
                <Tuple>
                    <Member Hierarchy="[Geography].[Country]">
                        <UName>[Geography].[Country].[All Geographies]</UName>
                        <Caption>All Geographies</Caption>
                        <LName>[Geography].[Country].[ (All) ]</LName>
                        <LNum>0</LNum>
                        <DisplayInfo>6</DisplayInfo>
                    </Member>
                    <Member Hierarchy="[Customer].[Gender]">
                        <UName>[Customer].[Gender].[All Customers]</UName>
                        <Caption>All Customers</Caption>
                        <LName>[Customer].[Gender].[ (All) ]</LName>
                        <LNum>0</LNum>
                        <DisplayInfo>65538</DisplayInfo>
                    </Member>
                </Tuple>
                ...
                <Tuple>
                    <Member Hierarchy="[Geography].[Country]">
                        <UName>[Geography].[Country].&[United States]</UName>
                        <Caption>United States</Caption>
                        <LName>[Geography].[Country].[Country]</LName>
                        <LNum>1</LNum>
                        <DisplayInfo>131072</DisplayInfo>
                    </Member>
                    <Member Hierarchy="[Customer].[Gender]">
                        <UName>[Customer].[Gender].[All Customers]</UName>
                        <Caption>All Customers</Caption>
                        <LName>[Customer].[Gender].[ (All) ]</LName>
                        <LNum>0</LNum>
                        <DisplayInfo>65538</DisplayInfo>
                    </Member>
                </Tuple>
                ...
            </Tuples>
        </Axis>
    </Axes>

```

2.2.4.1.2.1.3 CellInfo element

The CellInfo element describes the properties of a data cell. All elements appearing in the element compose the schema definition for the Cell element described at 2.2.4.1.4. The value of attribute Name contains the property name of the cell at the server.

```
<xsd:complexType name="CellInfo">
```

```

    <xsd:sequence minOccurs="0" maxOccurs="unbounded">
      <xsd:any namespace="##targetNamespace" minOccurs="0" />
    </xsd:choice>
  </xsd:complexType>

```

The following sample shows how the CellInfo elements describe the contents of the Cell elements in the CellData element.

```

<CellInfo>
  <Value name="VALUE" />
  <FmtValue name="FORMATTED_VALUE" type="xsd:string" />
</CellInfo>
...
<CellData>
  ...
  <Cell CellOrdinal="628">
    <Value xsi:type="xsd:double">1.</Value>
    <FmtValue>1.00</FmtValue>
  </Cell>
  <Cell CellOrdinal="629">
    <Value xsi:type="xsd:double">1.</Value>
    <FmtValue>1.00</FmtValue>
  </Cell>
</CellData>

```

2.2.4.1.3 Axes sub-element

The Axes sub-element describes the structure of the axes in the result cube. The Axes sub-element contains one or more Axis sub-elements as child elements. There is one Axis sub-element per each axis of the results cube, including the slicer axis.

The Axis sub-element contains all the data values, in this response, for each member of the axis. For more information about this element, see [xm1a-ds:Axes complex type](#).

The following text shows the first data element for the Axis0 axis:

```

<Axes>
  <Axis name="Axis0">
    <Tuples>
      <Tuple>
        <Member Hierarchy="[Measures]">
          <UName>[Measures].[Internet Sales Amount]</UName>
          <Caption>Internet Sales Amount</Caption>
          <LName>[Measures].[MeasuresLevel]</LName>
          <LNum>0</LNum>
          <DisplayInfo>0</DisplayInfo>
        </Member>
      </Tuple>
    </Tuples>
  </Axis>
</Axes>

```

```

        </Tuple>
        ...
    </Tuples>
</Axis>
...
</Axes>

```

2.2.4.1.3.1 xmla-ds:Axes complex type

The xmla-ds:Axes data type describes the values and properties of each member in the hierarchy of an axis in the results cube.

The Axes type is composed of elements of a Set type as shown by the following schema definition. For more information about the set types defined for the Axes data type, see [SetType Model Group](#).

```

<xsd:complexType name="Axes">
  <xsd:sequence>
    <xsd:element name="Axis" maxOccurs="unbounded">
      <xsd:complexType>
        <xsd:group ref="SetType" minOccurs="0" maxOccurs="unbounded" />
        <xsd:attribute name="name" type="xsd:string" />
      </xsd:complexType>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>

```

2.2.4.1.3.1.1 SetType Model Group

The SetType model group describes the possible types of sets that belong to the group. For more information about model groups, see [XMLSCHEMA1](#).

```

<xsd:group name="SetType">
  <xsd:choice>
    <xsd:element name="Members" type="MembersType" />
    <xsd:element name="Tuples" type="TuplesType" />
    <xsd:element name="CrossProduct" type="SetListType" />
    <xsd:element name="Union">
      <xsd:complexType>
        <xsd:group ref="SetType" minOccurs="0" maxOccurs="unbounded" />
      </xsd:complexType>
    </xsd:element>
  </xsd:choice>
</xsd:group>

```

Each set type in the group is described independently in its own section:

- [MembersType complex type](#)

- [TuplesType complex type](#)
- [SetListType complex type](#)

The Union sub-element is an invocation of the [SetType model group](#) defined here. <3>

2.2.4.1.3.1.1.1 MembersType complex type

The MembersType complex type is a collection of MemberType elements. For more information, see [MemberType complex type](#).

```
<xsd:complexType name="MembersType">
  <xsd:sequence>
    <xsd:element name="Member" type="MemberType" minOccurs="0" maxOccurs="unbounded" />
  </xsd:sequence>
  <xsd:attribute name="Hierarchy" type="xsd:string" use="required" />
</xsd:complexType>
```

2.2.4.1.3.1.1.1.1 MemberType complex type

The MemberType type describes the properties of a simple element in the axis hierarchy. For example, in the Year:Quarter:Month:MonthDay axis [dimension](#), the member type describes the Year, the Quarter, the Month, or the MonthDay member of the hierarchy.

The schema definition of the MemberType type is defined in the AxisInfo specification of this message.

```
<xsd:complexType name="MemberType">
  <xsd:sequence>
    <xsd:any namespace="##targetNamespace" minOccurs="0" maxOccurs="unbounded"
processContents="skip" />
  </xsd:sequence>
  <xsd:attribute name="Hierarchy" type="xsd:string" />
</xsd:complexType>
```

The MemberType type is a collection of properties and values. The following sample shows the how the member values match the given specification in the AxisSection. Notice the matching link between <HierarchyInfo name="[Date].[Calendar Quarter]"> and <Member Hierarchy="[Date].[Calendar Quarter]">.

```
<AxisInfo name="Axis1">
  <HierarchyInfo name="[Date].[Calendar Quarter]">
    <UName name="[Date].[Calendar Quarter].[MEMBER_UNIQUE_NAME]" type="xsd:string" />
    <Caption name="[Date].[Calendar Quarter].[MEMBER_CAPTION]" type="xsd:string" />
    <LName name="[Date].[Calendar Quarter].[LEVEL_UNIQUE_NAME]" type="xsd:string" />
    <LNum name="[Date].[Calendar Quarter].[LEVEL_NUMBER]" type="xsd:int" />
    <DisplayInfo name="[Date].[Calendar Quarter].[DISPLAY_INFO]" type="xsd:unsignedInt"
  />
</HierarchyInfo>
```

```

</AxisInfo>
...
<Axis name="Axis1">
  <Tuples>
    <Tuple>
      <Member Hierarchy="[Date].[Calendar Quarter]">
        <UName>[Date].[Calendar Quarter].&[2001]&[3]</UName>
        <Caption>Q3 CY 2001</Caption>
        <LName>[Date].[Calendar Quarter].[Calendar Quarter]</LName>
        <LNum>1</LNum>
        <DisplayInfo>0</DisplayInfo>
      </Member>
    </Tuple>
    <Tuple>
      <Member Hierarchy="[Date].[Calendar Quarter]">
        <UName>[Date].[Calendar Quarter].&[2001]&[4]</UName>
        <Caption>Q4 CY 2001</Caption>
        <LName>[Date].[Calendar Quarter].[Calendar Quarter]</LName>
        <LNum>1</LNum>
        <DisplayInfo>131072</DisplayInfo>
      </Member>
    </Tuple>
  </Tuples>
  ...

```

2.2.4.1.3.1.1.2 *TuplesType complex type*

The **TuplesType** complex type is a collection of **TupleType** elements. For more information, see [TupleType complex type](#).

```

<xsd:complexType name="TuplesType">
  <xsd:sequence>
    <xsd:element name="Tuple" type="TupleType" minOccurs="0" maxOccurs="unbounded" />
  </xsd:sequence>
</xsd:complexType>

```

2.2.4.1.3.1.1.2.1 *TupleType complex type*

The **TupleType** type describes the set of members that identifies a point in the axis hierarchy. A **TupleType** type is a collection of **MemberType** [objects](#).

```

<xsd:complexType name="TupleType">
  <xsd:sequence>
    <xsd:element name="Member" type="MemberType" maxOccurs="unbounded" />
  </xsd:sequence>
</xsd:complexType>

```

2.2.4.1.3.1.1.3 *SetListType complex type*

A **SetListType** type describes a complex type composed of elements that belong to the **SetType** group.

```

<xsd:complexType name="SetListType">
  <xsd:group ref="SetType" minOccurs="0" maxOccurs="unbounded" />
  <xsd:attribute name="Size" type="xsd:unsignedInt" />
</xsd:complexType>

```

2.2.4.1.4 *CellData sub-element*

The CellData sub-element contains all the data values for each of the cells in the response. For more information, see [xmla-ds:CellData complex type](#).

The following text shows the first data element of all the cells:

```

<CellData>
  <Cell CellOrdinal="0">
    <Value xsi:type="xsd:decimal">1453522.8854</Value>
    <FmtValue>$1,453,522.89</FmtValue>
  </Cell>
  ...
</CellData>

```

2.2.4.1.4.1 **xmla-ds:CellData complex type**

The CellData type describes a collection of Cell elements. The Cell element describes the properties of a single cell of data.

The schema definition for the Cell element is defined in the [CellInfo element](#) specification of the current instance of this message.

```

<xsd:complexType name="CellData">
  <xsd:sequence>
    <xsd:element name="Cell" minOccurs="0" maxOccurs="unbounded">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="Value" minOccurs="0" maxOccurs="1">
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="Error" minOccurs="0" maxOccurs="unbounded">
                  <xsd:element name="ErrorCode" minOccurs="0" maxOccurs="1" />
                  <xsd:element name="Description" minOccurs="0" maxOccurs="1" />
                  <xsd:any namespace="##targetNamespace" minOccurs="0"
maxOccurs="unbounded" />
                </xsd:element>
              </xsd:sequence>
            </xsd:complexType>
          </xsd:element>
          <xsd:any namespace="##targetNamespace" minOccurs="0" maxOccurs="unbounded"
processContents="skip" />
        </xsd:sequence>
        <xsd:attribute name="CellOrdinal" type="xsd:unsignedInt" use="required" />
      </xsd:complexType>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>

```

```

    </xsd:element>
  </xsd:sequence>
</xsd:complexType>

```

The following sample shows the relationship between the CellInfo element in the OlapInfo type and the cell data.

```

...
  <CellInfo>
    <Value name="VALUE" />
    <FmtValue name="FORMATTED_VALUE" type="xsd:string" />
  </CellInfo>
...
  <CellData>
    ...
    <Cell CellOrdinal="385">
      <Value xsi:type="xsd:int">976</Value>
      <FmtValue>976</FmtValue>
    </Cell>
    <Cell CellOrdinal="388">
      <Value xsi:type="xsd:double">1.</Value>
      <FmtValue>1.00</FmtValue>
    </Cell>
    <Cell CellOrdinal="389">
      <Value xsi:type="xsd:double">1.</Value>
      <FmtValue>1.00</FmtValue>
    </Cell>
  </CellData>
</root>

```

2.2.4.1.4.1.1 CellOrdinal attribute

The CellOrdinal attribute **MUST** be specified and indicates the ordinal of the cell. CellOrdinal is numbered 0 to $n-1$, for n cells.

The axis reference for a cell can be calculated based on CellOrdinal. Conceptually, cells are numbered in a [dataset](#) as if the dataset were a p -dimensional array, where p is the number of axes. Cells are addressed in row-major order. The following illustration shows the formula for calculating the ordinal number of a cell.

If axis k has U_k members, the ordinal number of a cell whose tuple ordinals are $(S_0, S_1, S_2, \dots, S_{p-1})$ is

$$\sum_{i=0}^{p-1} S_i \times E_i \text{ where } E_0 = 1 \text{ and } E_i = \prod_{k=0}^{i-1} U_k$$

Σ represents the sum of the terms in the series and Π the product.

We will apply the above formula to the result set shown in the following table. The query asked for four [measures](#) on columns and a crossjoin of two states with four quarters on rows. In following the dataset result, the **CellOrdinal** property for the part of the dataset result shown in the box is the set {9, 10, 11, 13, 14, 15, 17, 18, 19}. This is because the cells are numbered in row major order, starting with a **CellOrdinal** of zero for the upper left cell.

Next, we apply the above formula to the cell that is {CA, Q3, Store Cost}. Axis $k=0$ has $U_k=4$ members and axis $k=1$ has $U_k=8$ [tuples](#). P is the total number of axes in the query, here equal to 2. S_0 , the initial summation is $i=0$ to 1. For $i=0$, the tuple ordinal on axis 0 of {Store Cost} is 1. For $i=1$, the tuple ordinal of {CA, Q3} is 2.

For $i=0$, $E_i = 1$, so for $i=0$ the sum is $1 * 1 = 1$ and for $i=1$, the sum is 2 (tuple ordinal) * 4 (the value of E_i , computed as $1 * 4$), or 8 , and so the sum is equal to $1 + 8 = 9$, the cell ordinal for that cell.

		Unit Sales	Store Cost	Store Sales	Sales Count
CA	Q1	16,890.00	14,431.09	\$36,175.20	5498
	Q2	18,052.00	15,332.02	\$38,396.75	5915
	Q3	18,370.00	15,672.83	\$39,394.05	6014
	Q4	21,436.00	18,094.50	\$45,201.84	7015
OR	Q1	19,287.00	16,081.07	\$40,170.29	6184
	Q2	15,079.00	12,678.96	\$31,772.88	4799
	Q3	16,940.00	14,273.78	\$35,880.46	5432
	Q4	16,353.00	13,738.68	\$34,453.44	5196

2.2.4.1.4.1.2 Cell Value errors

Whenever an error occurs trying to return the value of a cell the server sends an Error element, inside the Value element of the cell, describing the error. For more information about the corresponding grammar, see [xm1a-ds:CellData complex type](#).

The following sample shows a security error when the user tried to get results from a cell to which he or she has restricted access.

```
<CellData>
...
  <Cell CellOrdinal="10">
    <Value>
      <Error>
        <ErrorCode>2148497527</ErrorCode>
        <Description>Security Error.</Description>
      </Error>
    </Value>
  </Cell>
...
</CellData>
</root>
```

2.2.4.1.5 Exception sub-element

The Exception sub-element signals that an exception has occurred at the server during the processing of the client request, and that no further data is available to the client.

The Exception sub-element MUST be an empty element. All available information about the causes of the exception SHOULD come in the Messages sub-element following this element. The Exception sub-element MUST precede the Messages sub-element if it exists.

2.2.4.1.6 Messages sub-element

The Messages sub-element contains information about error messages and warnings that occurred during the processing of the client request.

2.2.4.2 xm1a-e:root complex type

The xm1a-e:root complex type describes a response to a valid empty response or an exception that occurs at the server during the processing of a client request, and that no further data is available to the client. The server response MAY contain information about the causes of the exception. An empty response signals a successful execution of the client request with no further information on the response needed to the client.

```
<xsd:element name="root">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="Exception" type="xm1a-x:Exception" minOccurs="0" />
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```

```

        <xsd:element name="Messages" type="xmla-x:Messages" minOccurs="0" />
    </xsd:sequence>
</xsd:complexType>
</xsd:element>

```

2.2.4.2.1 Exception sub-element

The Exception sub-element signals that an exception has occurred at the server during the processing of the client request, and no further data is available to the client.

The Exception sub-element **MUST** be an empty element. All available information about the causes of the exception **SHOULD** come in the Messages sub-element following this element. The Exception sub-element **MUST** precede the Messages sub-element if it exists.

2.2.4.2.2 Messages sub-element

The Messages sub-element contains information about error messages and warnings that occurred during the processing of the client request. For more information, see [xmla-e:root complex type](#).

2.2.4.3 xmla-rs:root complex type

The xmla-rs:root complex type describes data from a cube in a tabular representation. This means that the data is flattened into an extended denormalized table. The xmla-rs:root consists of these main sections:

- Schema: Contains the complete schema definition for this message.
- Row: Describes the structure of the result set.

```

<xsd:element name="root">
  <xsd:complexType>
    <xsd:sequence minOccurs="0" maxOccurs="unbounded">
      <xsd:element name="schema" namespace="http://www.w3.org/2001/XMLSchema" target
namespace="http://www.w3.org/2001/XMLSchema" processContents="strict" minOccurs="0" />
      <xsd:element name="Row" type="xmla-rs:row" minOccurs="0" />
      <xsd:element name="Exception" type="xmla-x:Exception" minOccurs="0" />
      <xsd:element name="Messages" type="xmla-x:Messages" minOccurs="0" />
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>

```

2.2.4.3.1 Schema sub-element

The Schema sub-element contains the complete description of the structure of the data being sent within this root element. By reading the Schema sub-element, the client can fully parse the response received. This element **MUST** come with any response that contains data for the client. The Schema element does not describe exceptions or messages in the response.

The following sample illustrates the schema definition received for a Discover command over a DISCOVER_KEYWORDS request type.

```
<root xmlns="urn:schemas-microsoft-com:xml-analysis:rowset"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:schema targetNamespace="urn:schemas-microsoft-com:xml-analysis:rowset"
xmlns:sql="urn:schemas-microsoft-com:xml-sql" elementFormDefault="qualified">
    <xsd:element name="root">
      <xsd:complexType>
        <xsd:sequence minOccurs="0" maxOccurs="unbounded">
          <xsd:element name="row" type="row" />
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
    <xsd:complexType name="row">
      <xsd:sequence>
        <xsd:element sql:field="Keyword" name="Keyword" type="xsd:string" />
      </xsd:sequence>
    </xsd:complexType>
  </xsd:schema>
  <row>
    <Keyword>ACTIONPARAMETERSET</Keyword>
  </row>
  <row>
    <Keyword>ALLOWSIBLINGSWITHSAMENAME</Keyword>
  </row>
</root>
```

The schema syntax received allows the client to correctly parse the Row elements received and assemble a meaningful response.

2.2.4.3.2 Row sub-element

The Row sub-element contains a row of data for the extended denormalized table described above. The general structure of a Row sub-element is a collection of elements with values assigned to each. Each sub-element name is the name of a column in the extended table, and contains the value for the column at that particular row.

From the sample above, the schema definition in `<xsd:element name="row" type="row" />` states that the Row element is of type Row, and the type is defined as `<xsd:complexType name="row">` a few lines later. The schema says the Row element is composed of one element, with name 'Keyword' and of type 'string'. Also, the schema says that the element Keyword comes from a column with the name 'Keyword'.

With this information the client can assemble all of the Row elements received into a single column table similar to the following:

Keyword
ACTIONPARAMETERSET
ALLOWSIBLINGSWITHSAMENAME

2.2.4.3.2.1 xmla-rs:row complex type

The xmla-rs:row complex type describes a row structure from a tabular data representation. The xmla-rs:row data type contains the specific schema definition for the row of data being sent in the current response. Each element in the row type represents a column in the tabular representation of the row. The following syntax definition describes the grammar of the xmla-rs:row data type.

```
<xsd:complexType name="row">
  <xsd:sequence>
    <xsd:element name="row" minOccurs="0" maxOccurs="unbounded">
      <xsd:complexType>
        <xsd:element name="##targetName" minOccurs="0" maxOccurs="unbounded">
          <xsd:attribute name="sql:field" type="xsd:string" />
          <xsd:attribute name="name" type="xsd:string" minOccurs="1"/>
          <xsd:attribute name="type" type="xs:##targetType" minOccurs="1"/>
        </xsd:element>
      </xsd:complexType>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

Given the following Execute command:

```
<Execute xmlns="urn:schemas-microsoft-com:xml-analysis">
  <Command>
    <Statement>
      Select nonemptycrossjoin([Date].[Calendar Quarter of Year].members,
[Customer].[Gender].members) on 1,
      { [Measures].[Internet Sales Amount], [Measures].[Internet Order Quantity]} on 0
      from [Adventure Works]
    </Statement>
  </Command>
  <Properties>
    <PropertyList>
      <Catalog>Adventure Works DW</Catalog>
      <Format>Tabular</Format>
    </PropertyList>
  </Properties>
</Execute>
```

The following sample, taken from the response to the previous execute command, shows the relationship between the schema definition of the row complex type and the row elements in the response.

```
<ExecuteResponse xmlns="urn:schemas-microsoft-com:xml-analysis">
  <return>
    <root xmlns="urn:schemas-microsoft-com:xml-analysis:rowset"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns:xsd="http://www.w3.org/2001/XMLSchema">
      <xsd:schema targetNamespace="urn:schemas-microsoft-com:xml-analysis:rowset"
        xmlns:sql="urn:schemas-microsoft-com:xml-sql" elementFormDefault="qualified">
        <xsd:element name="root">
          <xsd:complexType>
            <xsd:sequence minOccurs="0" maxOccurs="unbounded">
              <xsd:element name="row" type="row" />
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
        ...
        <xsd:complexType name="row">
          <xsd:sequence>
            <xsd:element sql:field="[Date].[Calendar Quarter of Year].[Calendar Quarter
of Year].[MEMBER_CAPTION]"
              name="_x005B_Date_x005D_. _x005B_Calendar_x0020_Quarter_x0020_of_x0020_Year_x005D_. _x005B_
Calendar_x0020_Quarter_x0020_of_x0020_Year_x005D_. _x005B_MEMBER_CAPTION_x005D_"
              type="xsd:string" minOccurs="0" />
            <xsd:element sql:field="[Customer].[Gender].[Gender].[MEMBER_CAPTION]"
              name="_x005B_Customer_x005D_. _x005B_Gender_x005D_. _x005B_Gender_x005D_. _x005B_MEMBER_CAPT
ION_x005D_" type="xsd:string" minOccurs="0" />
            <xsd:element sql:field="[Measures].[Internet Sales Amount]"
              name="_x005B_Measures_x005D_. _x005B_Internet_x0020_Sales_x0020_Amount_x005D_"
              minOccurs="0" />
            <xsd:element sql:field="[Measures].[Internet Order Quantity]"
              name="_x005B_Measures_x005D_. _x005B_Internet_x0020_Order_x0020_Quantity_x005D_"
              minOccurs="0" />
          </xsd:sequence>
        </xsd:complexType>
      </xsd:schema>
      <row>
        <_x005B_Measures_x005D_. _x005B_Internet_x0020_Sales_x0020_Amount_x005D_
          xsi:type="xsd:decimal">29358677.2207</_x005B_Measures_x005D_. _x005B_Internet_x0020_Sales_
          x0020_Amount_x005D_>
        <_x005B_Measures_x005D_. _x005B_Internet_x0020_Order_x0020_Quantity_x005D_
          xsi:type="xsd:int">60398</_x005B_Measures_x005D_. _x005B_Internet_x0020_Order_x0020_Quanti
          ty_x005D_>
      </row>
      <row>

<_x005B_Customer_x005D_. _x005B_Gender_x005D_. _x005B_Gender_x005D_. _x005B_MEMBER_CAPTION_x
```

```

005D>Female</_x005B_Customer_x005D_. _x005B_Gender_x005D_. _x005B_Gender_x005D_. _x005B_MEMBER_CAPTION_x005D_>
    <_x005B_Measures_x005D_. _x005B_Internet_x0020_Sales_x0020_Amount_x005D_
xsi:type="xsd:decimal">14813618.6752</_x005B_Measures_x005D_. _x005B_Internet_x0020_Sales_
x0020_Amount_x005D_>
    <_x005B_Measures_x005D_. _x005B_Internet_x0020_Order_x0020_Quantity_x005D_
xsi:type="xsd:int">30017</_x005B_Measures_x005D_. _x005B_Internet_x0020_Order_x0020_Quanti
ty_x005D_>
    </row>
<row>

<_x005B_Customer_x005D_. _x005B_Gender_x005D_. _x005B_Gender_x005D_. _x005B_MEMBER_CAPTION_x
005D_>Male</_x005B_Customer_x005D_. _x005B_Gender_x005D_. _x005B_Gender_x005D_. _x005B_MEMBER
R_CAPTION_x005D_>
    <_x005B_Measures_x005D_. _x005B_Internet_x0020_Sales_x0020_Amount_x005D_
xsi:type="xsd:decimal">14545058.5455</_x005B_Measures_x005D_. _x005B_Internet_x0020_Sales_
x0020_Amount_x005D_>
    <_x005B_Measures_x005D_. _x005B_Internet_x0020_Order_x0020_Quantity_x005D_
xsi:type="xsd:int">30381</_x005B_Measures_x005D_. _x005B_Internet_x0020_Order_x0020_Quanti
ty_x005D_>
    </row>
<row>

<_x005B_Date_x005D_. _x005B_Calendar_x0020_Quarter_x0020_of_x0020_Year_x005D_. _x005B_Calen
dar_x0020_Quarter_x0020_of_x0020_Year_x005D_. _x005B_MEMBER_CAPTION_x005D_>CY
Q1</_x005B_Date_x005D_. _x005B_Calendar_x0020_Quarter_x0020_of_x0020_Year_x005D_. _x005B_Ca
lendar_x0020_Quarter_x0020_of_x0020_Year_x005D_. _x005B_MEMBER_CAPTION_x005D_>
    <_x005B_Measures_x005D_. _x005B_Internet_x0020_Sales_x0020_Amount_x005D_
xsi:type="xsd:decimal">7488858.7127</_x005B_Measures_x005D_. _x005B_Internet_x0020_Sales_x
0020_Amount_x005D_>
    <_x005B_Measures_x005D_. _x005B_Internet_x0020_Order_x0020_Quantity_x005D_
xsi:type="xsd:int">15254</_x005B_Measures_x005D_. _x005B_Internet_x0020_Order_x0020_Quanti
ty_x005D_>
    </row>
<row>

<_x005B_Date_x005D_. _x005B_Calendar_x0020_Quarter_x0020_of_x0020_Year_x005D_. _x005B_Calen
dar_x0020_Quarter_x0020_of_x0020_Year_x005D_. _x005B_MEMBER_CAPTION_x005D_>CY
Q1</_x005B_Date_x005D_. _x005B_Calendar_x0020_Quarter_x0020_of_x0020_Year_x005D_. _x005B_Ca
lendar_x0020_Quarter_x0020_of_x0020_Year_x005D_. _x005B_MEMBER_CAPTION_x005D_>

<_x005B_Customer_x005D_. _x005B_Gender_x005D_. _x005B_Gender_x005D_. _x005B_MEMBER_CAPTION_x
005D_>Female</_x005B_Customer_x005D_. _x005B_Gender_x005D_. _x005B_Gender_x005D_. _x005B_MEMBER
CAPTION_x005D_>
    <_x005B_Measures_x005D_. _x005B_Internet_x0020_Sales_x0020_Amount_x005D_
xsi:type="xsd:decimal">3706159.9987</_x005B_Measures_x005D_. _x005B_Internet_x0020_Sales_x
0020_Amount_x005D_>
    <_x005B_Measures_x005D_. _x005B_Internet_x0020_Order_x0020_Quantity_x005D_
xsi:type="xsd:int">7460</_x005B_Measures_x005D_. _x005B_Internet_x0020_Order_x0020_Quantit
y_x005D_>

```

```

        </row>
    ...
</root>
</return>
</ExecuteResponse>

```

In the sample, at the definition of the row data type, the first element is defined as:

sql:field	"[Date].[Calendar Quarter of Year].[Calendar Quarter of Year].[MEMBER_CAPTION]"
name	"_x005B_Date_x005D_. _x005B_Calendar_x0020_Quarter_x0020_of_x0020_Year_x005D_. _x005B_Calendar_x0020_Quarter_x0020_of_x0020_Year_x005D_. _x005B_MEMBER_CAPTION_x005D_"
type	"xsd:string"
minOccurs	"0"

For the same definition at the row elements you can find that there is an element with the same name and of type: string.

```

<_x005B_Date_x005D_. _x005B_Calendar_x0020_Quarter_x0020_of_x0020_Year_x005D_. _x005B_Calendar_x0020_Quarter_x0020_of_x0020_Year_x005D_. _x005B_MEMBER_CAPTION_x005D_>CY
Q1</_x005B_Date_x005D_. _x005B_Calendar_x0020_Quarter_x0020_of_x0020_Year_x005D_. _x005B_Calendar_x0020_Quarter_x0020_of_x0020_Year_x005D_. _x005B_MEMBER_CAPTION_x005D_>

```

Along with this element you can also find the rest of the elements defined in the schema definition.

If an element from the row schema is missing, then the data returned for that row belongs to the 'All' member of the missing hierarchy member. In the sample above, the first row returned does not contain the elements

_x005B_Date_x005D_. _x005B_Calendar_x0020_Quarter_x0020_of_x0020_Year_x005D_. _x005B_Calendar_x0020_Quarter_x0020_of_x0020_Year_x005D_. _x005B_MEMBER_CAPTION_x005D_ and

_x005B_Customer_x005D_. _x005B_Gender_x005D_. _x005B_Gender_x005D_. _x005B_MEMBER_CAPTION_x005D_, meaning that the returned row contains information for [Date].[Calendar Quarter of Year].[All] and [Customer].[Gender].[All].

2.2.4.3.2.1.1 Nested Rowsets elements

A Nested Rowset element is an element of a complex type that MUST be described explicitly within the row schema definition.

For example, the DISCOVER_SCHEMA_ROWSETS request returns the list of possible restrictions for each schema rowset: a collection of values for a single field. The following sample shows the syntax in the schema definition:

```
<xsd:complexType name="row">
  <xsd:sequence>
    <xsd:element sql:field="SchemaName" name="SchemaName" type="xsd:string" />
    <xsd:element sql:field="SchemaGuid" name="SchemaGuid" type="uuid" minOccurs="0" />
    <xsd:element sql:field="Restrictions" name="Restrictions" minOccurs="0"
maxOccurs="unbounded">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element sql:field="Name" name="Name" type="xsd:string" minOccurs="0" />
          <xsd:element sql:field="Type" name="Type" type="xsd:string" minOccurs="0" />
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
    <xsd:element sql:field="Description" name="Description" type="xsd:string"
minOccurs="0" />
    <xsd:element sql:field="RestrictionsMask" name="RestrictionsMask"
type="xsd:unsignedLong" minOccurs="0" />
  </xsd:sequence>
</xsd:complexType>
...
<row>
  <SchemaName>DBSCHEMA_TABLES</SchemaName>
  <SchemaGuid>C8B52229-5CF3-11CE-ADE5-00AA0044773D</SchemaGuid>
  <Restrictions>
    <Name>TABLE_CATALOG</Name>
    <Type>xsd:string</Type>
  </Restrictions>
  <Restrictions>
    <Name>TABLE_SCHEMA</Name>
    <Type>xsd:string</Type>
  </Restrictions>
  <Restrictions>
    <Name>TABLE_NAME</Name>
    <Type>xsd:string</Type>
  </Restrictions>
  <Restrictions>
    <Name>TABLE_TYPE</Name>
    <Type>xsd:string</Type>
  </Restrictions>
  <Restrictions>
    <Name>TABLE_OLAP_TYPE</Name>
    <Type>xsd:string</Type>
  </Restrictions>
  <RestrictionsMask>31</RestrictionsMask>
</row>
```

2.2.4.3.3 *Exception sub-element*

The Exception sub-element signals that an exception has occurred at the server during the processing of the client request, and that no further data is available to the client.

The Exception sub-element MUST be an empty element. All available information about the causes of the exception SHOULD^{<4>} come in the Messages sub-element following this element. The Exception sub-element MUST precede the Messages sub-element if it exists.

2.2.4.3.4 *Messages sub-element*

The Messages sub-element contains information about error messages and warnings that occurred during the processing of the client request. For more information about this sub-element, see [xm1a-e:root complex type](#).

2.2.4.4 **xm1a-x:Messages complex type**

The xm1a-x:Messages complex type describes the structure of the error and warning messages sent by the server to the client. The message type is a collection of Error elements or Warning elements. Each element is described through its attributes and MUST be an empty element.

```
<xsd:complexType name="Messages">
  <xsd:sequence>
    <xsd:element name="Error" minOccurs="0" maxOccurs="unbounded" >
      <xsd:attribute name="ErrorCode" type="xsd:unsignedInt" use="required" />
      <xsd:attribute name="Description" type="xsd:string" use="required" />
      <xsd:attribute name="HelpFile" type="xsd:string" use="optional" />
      <xsd:attribute name="Source" type="xsd:string" use="optional" />
    </xsd:element>
    <xsd:element name="Warning" minOccurs="0" maxOccurs="unbounded" >
      <xsd:attribute name="ErrorCode" type="xsd:unsignedInt" use="required" />
      <xsd:attribute name="Description" type="xsd:string" use="required" />
      <xsd:attribute name="HelpFile" type="xsd:string" use="optional" />
      <xsd:attribute name="Source" type="xsd:string" use="optional" />
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

2.2.5 Simple Types

This specification does not define any common XML Schema [simple type](#) definitions.

2.2.6 Attributes

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

2.2.7 Groups

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

2.2.8 Attribute Groups

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

2.3 Transport Specific Message Details

Unless otherwise specified, this protocol uses network byte order (big-endian) for all data.

2.3.1 TCP

When using TCP as the transport, the client and server **MUST** compose messages using DIME [\[DIME\]](#). A DIME message consists of one or more DIME [records](#). Each DIME message **MAY** optionally be broken into smaller records. [<5>](#)

The following is the layout of a DIME record:

Field	Description
VERSION (5 bits)	Specifies the version of the DIME message.
MB (1 bit)	Specifies that this record is the first record of the message.
ME (1 bit)	Specifies that this record is the last record of the message.
CF (1 bit)	Specifies that the contents of the message have been broken into smaller records.
TYPE_T (4 bits)	Specifies the structure and format of the TYPE field.
RESERVED (4 bits)	The behavior of this field is undefined (must be set to 0). <6>
OPTIONS_LENGTH (16 bits)	Specifies the length (in bytes) of the OPTIONS field, excluding any necessary padding (up to 3 bytes).
ID_LENGTH (16 bits)	Specifies the length (in bytes) of the ID field, excluding any necessary padding (up to 3 bytes).

Field	Description
TYPE_LENGTH (16 bits)	Specifies the length (in bytes) of the TYPE field, excluding any necessary padding (up to 3 bytes).
DATA_LENGTH (32 bits)	Specifies the length (in bytes) of the DATA field, excluding any necessary padding (up to 3 bytes).
OPTIONS	Contains any optional information used by a DIME parser.
ID	Contains a uniform resource identifier (URI) for uniquely identifying a DIME payload with any additional padding; the length of this field is specified by ID_LENGTH. For more information, see [RFC2396] .
TYPE	Specifies the encoding for the record based on a type reference URI or a MIME media-type; reference type is specified by TYPE_T, and the length of this field is specified by TYPE_LENGTH. For more information, see [RFC2396] .
DATA	Contains the actual data payload for the record; format of the data depends on the TYPE specified for the record; length of this field is specified by DATA_LENGTH.

The VERSION field (5 bits) is used to identify the internal version of DIME parser used by both parties. This value MUST be set to 1.

The MB field (1 bit) MUST be set to 1 for every DIME record that is beginning a new DIME message and MUST be set to 0 for all consecutive DIME records.

The ME field (1 bit) MUST be set to 1 for every DIME record that is a last record of every DIME message and MUST be set to 0 for all other DIME records.

The CF field (1 bit) MUST be set to 1 for every chunked DIME record except for the last record. Every chunked sequence is required to be encapsulated entirely within one DIME

message and cannot span across multiple DIME messages. Therefore, a first or a middle record MUST NOT have the ME field value set to 1.

The TYPE_T field (4 bits) MUST be set to 1 for every DIME record that is beginning a new DIME message and MUST be set to 0 for all consecutive DIME records.

This protocol allows the optional use of binary xml [\[MS-BINXML\]](#) and compression that the client or server MAY apply on the SOAP request or response to reduce network latency. The following content types are supported:<7>

TYPE_LENGTH	TYPE	Description
8	text/xml	Data content is clear text xml
14	application/sx	Data content is binary xml
22	application/xml+xpress	Data content is compressed xml
21	application/sx+xpress	Data content is compressed binary xml

Because the support for binary [XML](#) and compression is optional, the client and server MUST negotiate the content type of the messages for the duration of the connection. This is done using [flags](#) in the OPTIONS field. The OPTIONS field consists of 4 bytes of which only the first byte is used. The last three bytes are reserved and MUST be set to zero. The following table describes the bits in the first byte in order from the least significant bit to the most significant bit.

Bit	Description
NEGO	Specifies whether message content type negotiation is complete.
REQ_SX	Specifies whether request from the client should or will be binary xml.
REQ_XPRESS	Specifies whether request from the client should or will be compressed.

41 of 178

Bit	Description
RESP_SX	Specifies whether response from the server should or will be binary xml.
RESP_XPRESS	Specifies whether response from the server should or will be compressed.
RESERVED	Behavior is undefined (MUST be set to 0).
RESERVED	Behavior is undefined (MUST be set to 0).
RESERVED	Behavior is undefined (MUST be set to 0).

2.3.2 HTTP/HTTPS

When using HTTP/HTTPS as the transport, the client and server MUST set the following HTTP headers:

Field	Description
SOAPAction	Specifies SOAP Action type: "urn:schemas-microsoft-com:xml-analysis:Discover" for Discover requests or "urn:schemas-microsoft-com:xml-analysis:Execute" for Execute requests.
X-Transport-Caps-Negotiation-Flags	Used for content type negotiation. The value is a comma-separated list of five values corresponding to NEGO, REQ_SX, REQ_XPRESS, RESP_SX and RESP_XPRESS.

Field	Description
Content-Type	<p>Specifies the content type of the payload. The value is one of the following:</p> <ul style="list-style-type: none"> - "text/xml" - "application/sx" - "application/xml+xpress" - "application/sx+xpress"

2.3.3 Encryption

When using TCP as the transport, the client and server MAY choose to encrypt or hash messages using [\[RFC2743\]](#). This is negotiated at the time of authentication after which the client and server can use GSS-API to determine if encryption or hashing is turned on for the connection.

If encryption or hashing is being used, the message can be composed of one or more encryption data blocks. Each encryption data block has the following layout:

Field	Description
DATA_SIZE (16 bit)	Specifies the size of the encrypted data. This field uses little-endian byte order.
TOKEN_SIZE (16 bit)	Specifies the size of the encryption token. This field uses little-endian byte order.
DATA	Encrypted data.
TOKEN	Encryption token.

Note that an encryption data block can span multiple DIME records.

2.3.4 Compression

When using TCP or HTTP/HTTPS as the transport, the client and server MAY choose to compress messages. This is first negotiated when the connection is established.

If compression is being used, the message can be composed of one or more compression data blocks. Each compression data block has the following layout:

Field	Description
ORIGINAL_SIZE (32 bit)	Specifies the original size of the data. This field uses little-endian byte order.
COMPRESSED_SIZE (32 bit)	Specifies the size of the data after compression. This field uses little-endian byte order.
DATA	Compressed data.

Note that a compression data block can span multiple DIME records or encryption data blocks.

2.3.5 Binary XML

When using TCP or HTTP/HTTPS as the transport, the client and server MAY choose to encode messages in binary XML [\[MS-BINXML\]](#). This is first negotiated when the connection is established.

The following table lists the binary XML data types supported in this protocol. The client and server MUST NOT use data types outside this list.

XSD type	Binary XML type
xsd:boolean	XSD-BOOLEAN
xsd:byte	SQL-TINYINT
xsd:short	SQL-SMALLINT

XSD type	Binary XML type
xsd:int	SQL-INT
xsd:long	SQL-BIGINT
xsd:unsignedByte	XSD-UNSIGNEDBYTE
xsd: unsignedShort	XSD-UNSIGNEDSHORT
xsd: unsignedInt	XSD-UNSIGNEDINT
xsd: unsignedLong	XSD-UNSIGNEDLONG
xsd:double	SQL-REAL
xsd:float	SQL-FLOAT
xsd:decimal	SQL-MONEY
xsd:dateTime	SQL-DATETIME
uuid	SQL-UUID
xsd:string	SQL-NTEXT SQL-NCHAR SQL-NVARCHAR
xsd:base64Binary	SQL-BINARY

3 Protocol Details

3.1 Server Details

3.1.1 Abstract Data Model

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

The server SHOULD organize its data as per the OLAP data model:

- The server contains a collection of databases.
- Each database contains a collection of cubes.
- Each cube contains a collection of dimensions, [measure groups](#), measures, sets, [key performance indicators \(KPI\)](#) and actions.
- Each dimension contains a collection of [hierarchies](#).
- Each hierarchy contains a collection of levels.
- Each level contains a collection of members.
- Each member contains a collection of properties.
- Each measure group contains a collection of measure group dimensions and measures.

For more information on the OLAP data model, see [\[MSDN-SSAS\]](#).

3.1.2 Timers

None. All protocol requests are initiated by the client.

3.1.3 Initialization

The server MUST start and begin listening for requests.

For stateless connections, no further initialization is required. For stateful connections, the following example shows how [sessions](#) are supported:

To begin the session, the client adds a BeginSession [SOAP Header](#) to the request.

```
<?xml version="1.0" encoding="utf-8"?>
<Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
  <Header>
    <BeginSession xmlns="urn:schemas-microsoft-com:xml-analysis" mustUnderstand="1"/>
  </Header>
  <Body>
```

```

    ...<!-- Discover or Execute element goes here.-->
  </Body>
</Envelope>

```

The SOAP response message from the server includes the session ID in the SOAP header.

```

<Header>
  <Session
    xmlns="urn:schemas-microsoft-com:xml-analysis"
    SessionId="537C61C6-827C-4305-83A6-C8CE4A91001B" />
</Header>

```

For each subsequent request, the client must include the Session ID provided by the server.

```

<Header>
  <Session
    xmlns="urn:schemas-microsoft-com:xml-analysis"
    mustUnderstand="1"
    SessionId="537C61C6-827C-4305-83A6-C8CE4A91001B" />
</Header>

```

To end the session, the client MUST send the <EndSession> header containing the related session ID value to the server.

```

<Header>
  <EndSession
    xmlns="urn:schemas-microsoft-com:xml-analysis"
    mustUnderstand="1"
    SessionId="537C61C6-827C-4305-83A6-C8CE4A91001B" />
</Header>

```

Support for sessions on the server is mandatory. [<8>](#) Multiple commands MAY be executed in the context of a single session. The server MAY choose to time out an active session after a period of inactivity. [<9>](#)

The following table lists the SOAP header elements and attributes that this protocol defines for initiating, maintaining, and closing a session.

SOAP Header	Description
BeginSession	Requests that the server create a new session. The server should respond by constructing a new session and returning the session ID in the Session header in the SOAP response.

SOAP Header	Description
Session	For every method call that is to occur in the session, this header must be used, and the session ID must be included in the header.
EndSession	To end the session, use this header. The session ID must be included in the header.

If the session ID specified in the Session or EndSession SOAP header is not valid or has timed out, then the server **MUST** return a [SOAP Fault](#).

3.1.4 Message Processing Events and Sequencing Rules

3.1.4.1 Authentication

This operation **MAY** be used by the client and server to exchange UTF-8 encoded security token data blocks as a part of the authentication process. The client **MUST** send an Authenticate request message and the server **MUST** respond with an AuthenticateResponse message.

The following WSDL describes the Authenticate operation.

```
<wsdl:operation name="Authenticate">
  <wsdl:input message="AuthenticateSoapIn" />
  <wsdl:output message="AuthenticateSoapOut" />
</wsdl:operation>
```

3.1.4.1.1 Messages

3.1.4.1.1.1 AuthenticateSoapIn

This message is the request message for the Authenticate operation.

The SOAP Action value of the message is defined as:

```
"http://schemas.microsoft.com/analysisisservices/2003/ext "
```

The [SOAP Body](#) **MUST** contain an Authenticate element.

```
<message name="AuthenticateSoapIn">
  <part name="parameters" element="xla:Authenticate" />
</message>
```

3.1.4.1.1.2 AuthenticateSoapOut

This message is the response message for the Authenticate operation.

The SOAP Action value of the message is defined as:

```
"http://schemas.microsoft.com/analysisservices/2003/ext"
```

The SOAP Body MUST contain a AuthenticateResponse element.

```
<message name=" AuthenticateSoapOut">
  <part name="parameters" element="xmla: AuthenticateResponse" />
</message>
```

3.1.4.1.2 Elements

The following XML Schema element definitions are specific to this operation.

3.1.4.1.2.1 Authenticate

The Authenticate element has the following definition:

```
<xs:element name="Authenticate">
  <xs:complexType>
    <xs:element name="SspiHandshake" minOccurs="1" maxOccurs="1" nillable="false"
type="s:base64Binary" />
  </xs:complexType>
</xs:element>
```

The following is an example of Authenticate request:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <Authenticate xmlns="http://schemas.microsoft.com/analysisservices/2003/ext">
      <SspiHandshake>[Base64 encoded security token data block here]</SspiHandshake>
    </Authenticate>
  </soap:Body>
</soap:Envelope>
```

3.1.4.1.2.2 AuthenticateResponse

The AuthenticateResponse element has the following definition:

```
<xs:element name=" AuthenticateResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="return" minOccurs="1" maxOccurs="1">
        <xs:complexType>
          <xs:element name="SspiHandshake" minOccurs="1" maxOccurs="1" nillable="false"
type="s:base64Binary" />
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

```

        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>

```

The following is an example of AuthenticateResponse response:

```

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <AuthenticateResponse xmlns="http://schemas.microsoft.com/analysisservices/2003/ext">
      <return>
        <SspiHandshake>[Base64 encoded security token data block here]</SspiHandshake>
      </return>
    </AuthenticateResponse>
  </soap:Body>
</soap:Envelope>

```

3.1.4.2 Discover

This operation is used to find information about an OLAP server.

The following WSDL describes the Discover operation.

```

<wsdl:operation name="Discover">
  <wsdl:input message="DiscoverSoapIn" />
  <wsdl:output message="DiscoverSoapOut" />
</wsdl:operation>

```

The protocol client **MUST** send a DiscoverSoapIn request message and the protocol server **MUST** respond with a DiscoverSoapOut response message.

3.1.4.2.1 Messages

3.1.4.2.1.1 DiscoverSoapIn

This message is the request message for the Discover operation.

The SOAP Action value of the message is defined as:

```
"urn:schemas-microsoft-com:xml-analysis:Discover"
```

The SOAP Body **MUST** contain a Discover element.

```

<message name="DiscoverSoapIn">
  <part name="parameters" element="xmla:Discover" />
</message>

```

3.1.4.2.1.2 DiscoverSoapOut

This message is the response message for the Discover operation.

The SOAP Action value of the message is defined as:

```
"urn:schemas-microsoft-com:xml-analysis:Discover"
```

The SOAP Body MUST contain a DiscoverResponse element.

```
<message name="DiscoverSoapOut">
  <part name="parameters" element="xsla:DiscoverResponse" />
</message>
```

3.1.4.2.2 Elements

The following XML Schema element definitions are specific to this operation.

3.1.4.2.2.1 Discover

The Discover element has the following definition:

```
<xsd:element name="Discover">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="RequestType" minOccurs="1" maxOccurs="1" type="s:string" />
      <xsd:element name="Restrictions" minOccurs="1" maxOccurs="1">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name="RestrictionList" minOccurs="0" maxOccurs="1"
nillable="true">
              <xsd:complexType>
                <xsd:any />
              </xsd:complexType>
            </xsd:element>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
      <xsd:element name="Properties" minOccurs="1" maxOccurs="1" >
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name="PropertyList" minOccurs="0" maxOccurs="1" nillable="true">
              <xsd:complexType>
                <xsd:any />
              </xsd:complexType>
            </xsd:element>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```

The Discover element describes the structure of the Discover request. The Discover request is used to obtain information about the metadata objects of the server.

The Discover request consists of these sub-elements:

- RequestType
- Restrictions
- Properties

3.1.4.2.2.1.1 RequestType sub-element

The RequestType sub-element is a string that specifies the name of the metadata object to be discovered. Each RequestType sub-element MUST correspond to a return rowset. The Discover message MUST contain one and only one RequestType sub-element. For rowset definitions, see 0. This enumeration MAY be extended to support server-specific enumeration strings. <[10](#)>

3.1.4.2.2.1.2 Restrictions sub-element

The Restrictions sub-element contains a set of properties and values that refine the results for the metadata object being discovered. It enables the restriction of data in the rowset returned in the DiscoverResponse message. For more information about a list of rowset columns that MAY be restricted, see [Discover Request Types](#). The format of the restrictions sub-element and the resulting XML result set is also dependent on the value specified in the RequestType sub-element. To obtain the restriction information for server-specific schema rowsets, the client MUST use the DISCOVER_SCHEMA_ROWSETS request type. <[11](#)>

The Restrictions sub-element MAY contain one and only one element RestrictionList. The RestrictionList sub-element MAY contain zero or more elements with a simple data type value. The name of the element represents a property of the object being discovered and the value represents the filtering value for the results to come. The RestrictionList sub-element MAY be an empty element.

The following example shows how to obtain, from the MDSCHEMA_DIMENSIONS, object, the information pertinent to CATALOG_NAME="Adventure Works DW" and CUBE_NAME="Adventure Works" properties.

```
<Discover xmlns="urn:schemas-microsoft-com:xml-analysis">
  <RequestType>MDSCHEMA_DIMENSIONS</RequestType>
  <Restrictions>
    <RestrictionList>
      <CATALOG_NAME>Adventure Works DW</CATALOG_NAME>
      <CUBE_NAME>Adventure Works</CUBE_NAME>
    </RestrictionList>
  </Restrictions>
```

52 of 178

```

<Properties>
  <PropertyList>
    <Catalog>Adventure Works DW</Catalog>
    <LocaleIdentifier>1036</LocaleIdentifier>
  </PropertyList>
</Properties>
</Discover>

```

3.1.4.2.2.1.3 **Properties** sub-element

This sub-element contains a collection of request properties. For the list of properties and their data types and descriptions, see 3.1.4.4.

3.1.4.2.2.2 **DiscoverResponse**

The DiscoverResponse element has the following definition:

```

<xsd:element name="DiscoverResponse">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="return" minOccurs="0" maxOccurs="1">
        <xsd:complexType>
          <xsd:element name="root" minOccurs="0" maxOccurs="1">
            </xsd:element>
          </xsd:complexType>
        </xsd:element>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>

```

The DiscoverResponse element describes the structure of response received from the Discover command.

The DiscoverResponse element consists of one Return sub-element which contains a root element.

3.1.4.2.2.2.1 Return sub-element

This sub-element, of the rowset type, contains the result set returned by the server. For more information about the rowset type, see 0.

3.1.4.2.3 **Discover Request Types**

The following sections list all Discover request types that an implementation of this protocol MUST support. The title of each section is the RequestType. Each section describes the request, specifies the restrictions, and specifies the columns in the rowset returned by the server.

All restrictions are optional; using an empty element or missing an element is the same as having used the default value for the restriction, if any defined. Unless otherwise noted in the

53 of 178

restrictions section a restriction with no default value means all possible values are to be returned if no value was specified for the restriction, the restriction was an empty element or a missing element.

All columns described in the columns section are mandatory in the returned row set. Values for any column are optional; but they MUST adhere to the specific type defined or the element MUST be the empty element or MUST be a missing element. When values are defined as a list of possible values then all returned values MUST be in the specified list or the element MUST be the empty element or MUST be a missing element.

3.1.4.2.3.1 DBSCHEMA_CATALOGS

This Discover element requests the catalogs accessible on the server.

3.1.4.2.3.1.1 Restrictions

The DBSCHEMA_CATALOGS rowset MAY be restricted on the columns listed in the following table.

Name	Type	Description
CATALOG_NAME	xsd:string	The catalog name.

3.1.4.2.3.1.2 Columns

The DBSCHEMA_CATALOGS rowset MUST contain the following columns.

Name	Type	Description
CATALOG_NAME	xsd:string	The catalog name.
DESCRIPTION	xsd:string	A human-readable description of the table.
ROLES	xsd:string	The list of roles to which the current user belongs. MUST be a comma delimited list. < 12 >
DATE_MODIFIED	xsd:dateTime	The date that the catalog was last modified.

The rowset is sorted on CATALOG_NAME.

3.1.4.2.3.2 MDSHEMA_CUBES

This Discover element describes the structure of cubes within a database.

3.1.4.2.3.2.1 Restrictions

The MDSHEMA_CUBES rowset MAY be restricted on the columns listed in the following table.

Name	Type	Description
CATALOG_NAME	xsd:string	The name of the database. MAY be null or an empty string.
SCHEMA_NAME	xsd:string	The name of the schema. MAY be null or an empty string.
CUBE_NAME	xsd:string	The name of the cube or dimension. Dimension names MUST be prefaced by a dollar sign (\$) symbol.
CUBE_SOURCE	xsd:unsignedShort	A bitmask with one of these valid values: Value Description 0x01 Cube 0x02 Dimension Default restriction is a value of 1.
BASE CUBE_NAME	xsd:string	The name of the source cube if this cube is a perspective cube.

3.1.4.2.3.2.2 Columns

Name	Type	Description
CATALOG_NAME	xsd:string	The catalog name. MUST NOT be null or an empty string.

Name	Type	Description
SCHEMA_NAME	xsd:string	The name of the schema. MAY be null or an empty string.< 13 >
CUBE_NAME	xsd:string	The name of the cube or dimension. Dimension names MUST be prefaced by a dollar sign (\$) symbol.
CUBE_TYPE	xsd:string	The type of the cube. Valid values are: <ul style="list-style-type: none"> • CUBE • DIMENSION.
CUBE_GUID	uuid	The globally unique identifier (GUID) of the cube.
CREATED_ON	xsd:dateTime	The date and time the cube was created.
LAST_SCHEMA_UPDATE	xsd:dateTime	The time that the cube was last processed.
SCHEMA_UPDATED_BY	xsd:string	The person whom last updated the cube.
LAST_DATA_UPDATE	xsd:dateTime	The time that the cube was last processed.
DATA_UPDATED_BY	xsd:string	The person whom last updated the data of the cube.
LAST_UPDATED_BY	xsd:string	The person whom last updated the cube.
DESCRIPTION	xsd:string	A user-friendly description of the cube.

Name	Type	Description						
IS_DRILLTHROUGH_ENABLED	xsd:boolean	A Boolean that indicates whether the cube can be drillthrough.< 14 >						
IS_LINKABLE	xsd:boolean	A Boolean that indicates whether a cube can be used in a linked cube.						
IS_WRITE_ENABLED	xsd:boolean	A Boolean that indicates whether a cube is write-enabled.						
IS_SQL_ENABLED	xsd:boolean	A Boolean that indicates whether SQL can be used on the cube.						
CUBE_CAPTION	xsd:string	The caption of the cube.						
BASE_CUBE_NAME	xsd:string	The name of the source cube if this cube is a perspective cube.						
CUBE_SOURCE	xsd:unsignedShort	A bitmask with one of these valid values: <table><tr><th>Value</th><th>Description</th></tr><tr><td>0x01</td><td>Cube</td></tr><tr><td>0x02</td><td>Dimension</td></tr></table>	Value	Description	0x01	Cube	0x02	Dimension
Value	Description							
0x01	Cube							
0x02	Dimension							

The rowset is sorted on CATALOG_NAME and CUBE_NAME.

3.1.4.2.3.3 MDSCHEMA_DIMENSIONS

This Discover element describes the shared and private dimensions within a database.

3.1.4.2.3.3.1 Restrictions

The MDSCHEMA_DIMENSIONS rowset MAY be restricted on the columns listed in the following table.

Name	Type	Description						
CATALOG_NAME	xsd:string	The name of the database.						
SCHEMA_NAME	xsd:string	The name of the schema. < 15 >						
CUBE_NAME	xsd:string	The name of the cube.						
DIMENSION_NAME	xsd:string	The name of the dimension.						
DIMENSION_UNIQUE_NAME	xsd:string	The unique name of the dimension.						
CUBE_SOURCE	xsd:unsignedShort	A bitmask with one of these valid values: <table><tr><th>Value</th><th>Description</th></tr><tr><td>0x01</td><td>Cube</td></tr><tr><td>0x02</td><td>Dimension</td></tr></table> Default restriction is a value of 1.	Value	Description	0x01	Cube	0x02	Dimension
Value	Description							
0x01	Cube							
0x02	Dimension							
DIMENSION_VISIBILITY	xsd:unsignedShort	A bitmask with one of these valid values: <table><tr><th>Value</th><th>Description</th></tr><tr><td>0x01</td><td>Visible</td></tr><tr><td>0x02</td><td>Not Visible</td></tr></table> Default restriction is a value of 1.	Value	Description	0x01	Visible	0x02	Not Visible
Value	Description							
0x01	Visible							
0x02	Not Visible							

3.1.4.2.3.3.2 Columns

Name	Type	Description
CATALOG_NAME	xsd:string	The name of the database.
SCHEMA_NAME	xsd:string	The name of the schema. < 16 >
CUBE_NAME	xsd:string	The name of the cube.

Name	Type	Description
DIMENSION_NAME	xsd:string	The name of the dimension.
DIMENSION_UNIQUE_NAME	xsd:string	The unique name of the dimension.
DIMENSION_GUID	uuid	The GUID of the dimension.
DIMENSION_CAPTION	xsd:string	The caption of the dimension. Use this when displaying the name of the dimension to the user, such as in the user interface or reports.
DIMENSION_ORDINAL	xsd:unsignedInt	The position of the dimension within the cube.

Name	Type	Description																																				
DIMENSION_TYPE	xsd:short	<div>The type of the dimension As described in the following table:</div> <table><tr><th>Value</th><th>Description</th></tr><tr><td>0</td><td>UNKNOWN</td></tr><tr><td>1</td><td>TIME</td></tr><tr><td>2</td><td>MEASURE</td></tr><tr><td>3</td><td>OTHER</td></tr><tr><td>5</td><td>QUANTITATIVE</td></tr><tr><td>6</td><td>ACCOUNTS</td></tr><tr><td>7</td><td>CUSTOMERS</td></tr><tr><td>8</td><td>PRODUCTS</td></tr><tr><td>9</td><td>SCENARIO</td></tr><tr><td>10</td><td>UTILIY</td></tr><tr><td>11</td><td>CURRENCY</td></tr><tr><td>12</td><td>RATES</td></tr><tr><td>13</td><td>CHANNEL</td></tr><tr><td>14</td><td>PROMOTION</td></tr><tr><td>15</td><td>ORGANIZATION</td></tr><tr><td>16</td><td>BILL OF MATERIALS</td></tr><tr><td>17</td><td>GEOGRAPHY</td></tr></table>	Value	Description	0	UNKNOWN	1	TIME	2	MEASURE	3	OTHER	5	QUANTITATIVE	6	ACCOUNTS	7	CUSTOMERS	8	PRODUCTS	9	SCENARIO	10	UTILIY	11	CURRENCY	12	RATES	13	CHANNEL	14	PROMOTION	15	ORGANIZATION	16	BILL OF MATERIALS	17	GEOGRAPHY
Value	Description																																					
0	UNKNOWN																																					
1	TIME																																					
2	MEASURE																																					
3	OTHER																																					
5	QUANTITATIVE																																					
6	ACCOUNTS																																					
7	CUSTOMERS																																					
8	PRODUCTS																																					
9	SCENARIO																																					
10	UTILIY																																					
11	CURRENCY																																					
12	RATES																																					
13	CHANNEL																																					
14	PROMOTION																																					
15	ORGANIZATION																																					
16	BILL OF MATERIALS																																					
17	GEOGRAPHY																																					
DIMENSION_CARDINALITY	xsd:unsignedInt	The number of members in the key attribute.																																				
DEFAULT_HIERARCHY	xsd:string	A hierarchy from the dimension.																																				
DESCRIPTION	xsd:string	A user-friendly description of the dimension.																																				

Name	Type	Description						
IS_VIRTUAL	xsd:boolean	A Boolean that indicates whether the dimension is virtual.< 17 >						
IS_READWRITE	xsd:boolean	A Boolean that indicates whether the dimension is write-enabled.						
DIMENSION_UNIQUE_SETTINGS	xsd:int	<div>A list of values that specifies which columns contain unique values:</div> <table><tr><th>Value</th><th>Description</th></tr><tr><td>0x00000001</td><td>Member key columns establish uniqueness</td></tr><tr><td>0x00000002</td><td>Member name columns establish uniqueness</td></tr></table>	Value	Description	0x00000001	Member key columns establish uniqueness	0x00000002	Member name columns establish uniqueness
Value	Description							
0x00000001	Member key columns establish uniqueness							
0x00000002	Member name columns establish uniqueness							
DIMENSION_MASTER_NAME	xsd:string	The name of the dimension.						
DIMENSION_IS_VISIBLE	xsd:boolean	A Boolean that indicates whether the dimension is visible.< 18 >						

The rowset is sorted on CATALOG_NAME, CUBE_NAME, and DIMENSION_NAME.

3.1.4.2.3.4 MDSHEMA_HIERARCHIES

This Discover element describes each hierarchy within a particular dimension.

3.1.4.2.3.4.1 Restrictions

The MDSHEMA_HIERARCHIES rowset MAY be restricted on the columns listed in the following table.

Name	Type	Description
CATALOG_NAME	xsd:string	The name of the database.
SCHEMA_NAME	xsd:string	The name of the schema.< 19 >

Name	Type	Description										
CUBE_NAME	xsd:string	The name of the cube.										
DIMENSION_UNIQUE_NAME	xsd:string	The unique name of the dimension.										
HIERARCHY_NAME	xsd:string	The name of the hierarchy.										
HIERARCHY_UNIQUE_NAME	xsd:string	The unique name of the hierarchy.										
HIERARCHY_ORIGIN	xsd:unsignedShort	<div>A bit mask that determines the source of the hierarchy.</div> <table><tr><th>Value</th><th>Description</th></tr><tr><td>0x01</td><td>Identifies user-defined hierarchies.</td></tr><tr><td>0x02</td><td>Identifies attribute hierarchies.</td></tr><tr><td>0x04</td><td>Identifies key attribute hierarchies.</td></tr><tr><td>0x08</td><td>Identifies attributes with no attribute hierarchies.</td></tr></table>	Value	Description	0x01	Identifies user-defined hierarchies.	0x02	Identifies attribute hierarchies.	0x04	Identifies key attribute hierarchies.	0x08	Identifies attributes with no attribute hierarchies.
Value	Description											
0x01	Identifies user-defined hierarchies.											
0x02	Identifies attribute hierarchies.											
0x04	Identifies key attribute hierarchies.											
0x08	Identifies attributes with no attribute hierarchies.											
CUBE_SOURCE	xsd:unsignedShort	<div>A bitmask with one of these valid values:</div> <table><tr><th>Value</th><th>Description</th></tr><tr><td>0x01</td><td>Cube</td></tr><tr><td>0x02</td><td>Dimension</td></tr></table> <div>Default restriction is a value of 1.</div>	Value	Description	0x01	Cube	0x02	Dimension				
Value	Description											
0x01	Cube											
0x02	Dimension											

Name	Type	Description						
HIERARCHY_VISIBILITY	xsd:unsignedShort	<p>A bitmask with one of the following valid values:</p> <table><tr><th>Value</th><th>Description</th></tr><tr><td>0x01</td><td>Visible</td></tr><tr><td>0x02</td><td>Not Visible</td></tr></table> <p>Default restriction is a value of 1.</p>	Value	Description	0x01	Visible	0x02	Not Visible
Value	Description							
0x01	Visible							
0x02	Not Visible							

3.1.4.2.3.4.2 Columns

The MDSHEMA_HIERARCHIES rowset contains the following columns.

Name	Type	Description
CATALOG_NAME	xsd:string	The name of the database.
SCHEMA_NAME	xsd:string	The name of the schema.
CUBE_NAME	xsd:string	The name of the cube to which this hierarchy belongs.
DIMENSION_UNIQUE_NAME	xsd:string	The unique name of the dimension to which this hierarchy belongs.< 20 >
HIERARCHY_NAME	xsd:string	The name of the hierarchy. This column MAY be blank if there is only a single hierarchy in the dimension.< 21 >
HIERARCHY_UNIQUE_NAME	xsd:string	The unique name of the hierarchy.
HIERARCHY_GUID	uuid	The GUID of the hierarchy.

Name	Type	Description																																				
HIERARCHY_CAPTION	xsd:string	<p>The caption of the hierarchy. Use this when displaying the name of the hierarchy to the user, such as in the user interface or reports.</p> <p>If a caption does not exist, HIERARCHY_NAME is returned. If the dimension either does not contain a hierarchy or has just one hierarchy, this column will contain the name of the dimension.</p>																																				
DIMENSION_TYPE	xsd:short	<p>The type of the dimension.</p> <table><tr><th>Value</th><th>Description</th></tr><tr><td>0</td><td>UNKNOWN</td></tr><tr><td>1</td><td>TIME</td></tr><tr><td>2</td><td>MEASURE</td></tr><tr><td>3</td><td>OTHER</td></tr><tr><td>5</td><td>QUANTITATIVE</td></tr><tr><td>6</td><td>ACCOUNTS</td></tr><tr><td>7</td><td>CUSTOMERS</td></tr><tr><td>8</td><td>PRODUCTS</td></tr><tr><td>9</td><td>SCENARIO</td></tr><tr><td>10</td><td>UTILITY</td></tr><tr><td>11</td><td>CURRENCY</td></tr><tr><td>12</td><td>RATES</td></tr><tr><td>13</td><td>CHANNEL</td></tr><tr><td>14</td><td>PROMOTION</td></tr><tr><td>15</td><td>ORGANIZATION</td></tr><tr><td>16</td><td>BILL_OF_MATERIALS</td></tr><tr><td>17</td><td>GEOGRAPHY</td></tr></table>	Value	Description	0	UNKNOWN	1	TIME	2	MEASURE	3	OTHER	5	QUANTITATIVE	6	ACCOUNTS	7	CUSTOMERS	8	PRODUCTS	9	SCENARIO	10	UTILITY	11	CURRENCY	12	RATES	13	CHANNEL	14	PROMOTION	15	ORGANIZATION	16	BILL_OF_MATERIALS	17	GEOGRAPHY
Value	Description																																					
0	UNKNOWN																																					
1	TIME																																					
2	MEASURE																																					
3	OTHER																																					
5	QUANTITATIVE																																					
6	ACCOUNTS																																					
7	CUSTOMERS																																					
8	PRODUCTS																																					
9	SCENARIO																																					
10	UTILITY																																					
11	CURRENCY																																					
12	RATES																																					
13	CHANNEL																																					
14	PROMOTION																																					
15	ORGANIZATION																																					
16	BILL_OF_MATERIALS																																					
17	GEOGRAPHY																																					

Name	Type	Description										
HIERARCHY_CARDINALITY	xsd:unsignedInt	The number of members in the hierarchy.										
DEFAULT_MEMBER	xsd:string	The default member for this hierarchy. This is a unique name. Every hierarchy MUST have a default member.										
ALL_MEMBER	xsd:string	The member name at the highest level of the hierarchy.										
DESCRIPTION	xsd:string	A human-readable description of the hierarchy.										
STRUCTURE	xsd:short	<p>The structure of the hierarchy.</p> <p>Valid values are defined in the following table.</p> <table><tr><th>Value</th><th>Description</th></tr><tr><td>0</td><td>Hierarchy is a fully balanced :</td></tr><tr><td>1</td><td>Hierarchy is a ragged balance</td></tr><tr><td>2</td><td>Hierarchy is an unbalanced st</td></tr><tr><td>3</td><td>Hierarchy is a network structu</td></tr></table>	Value	Description	0	Hierarchy is a fully balanced :	1	Hierarchy is a ragged balance	2	Hierarchy is an unbalanced st	3	Hierarchy is a network structu
Value	Description											
0	Hierarchy is a fully balanced :											
1	Hierarchy is a ragged balance											
2	Hierarchy is an unbalanced st											
3	Hierarchy is a network structu											
IS_VIRTUAL	xsd:boolean	A Boolean that indicates whether the hierarchy is a virtual hierarchy.<22>										
IS_READWRITE	xsd:boolean	A Boolean that indicates whether the Write Back to dimension column is enabled.<23>										

Name	Type	Description						
DIMENSION_UNIQUE_SETTINGS	xsd:int	<div>A list of values that specifies which columns contain unique values.<24></div> <table><thead><tr><th>Value</th><th>Description</th></tr></thead><tbody><tr><td>0x00000001</td><td>Member key columns establish uniqueness</td></tr><tr><td>0x00000002</td><td>Member name columns establish uniqueness</td></tr></tbody></table>	Value	Description	0x00000001	Member key columns establish uniqueness	0x00000002	Member name columns establish uniqueness
Value	Description							
0x00000001	Member key columns establish uniqueness							
0x00000002	Member name columns establish uniqueness							
DIMENSION_MASTER_UNIQUE_NAME	xsd:string	The name of the dimension.< 25 >						
DIMENSION_IS_VISIBLE	xsd:boolean	A Boolean that indicates whether the underlying dimension is visible.< 26 >						
HIERARCHY_ORDINAL	xsd:unsignedInt	The ordinal number of the hierarchy across all hierarchies of the cube.						
DIMENSION_IS_SHARED	xsd:boolean	A Boolean that indicates whether the underlying dimension is visible.< 27 >						
HIERARCHY_IS_VISIBLE	xsd:boolean	A Boolean that indicates whether the hierarchy is visible.						

Name	Type	Description										
HIERARCHY_ORIGINAL	xsd:unsignedShort	<p>A bit mask that determines the source of the hierarchy.</p> <table><thead><tr><th>Value</th><th>Description</th></tr></thead><tbody><tr><td>0x0001</td><td>Identifies user defined hierarchies.</td></tr><tr><td>0x0002</td><td>Identifies attribute hierarchies.</td></tr><tr><td>0x0004</td><td>Identifies key attribute hierarchies.</td></tr><tr><td>0x0008</td><td>Identifies attributes with no attribute hierarchies.</td></tr></tbody></table> <p>The default restriction value is 0x0003</p>	Value	Description	0x0001	Identifies user defined hierarchies.	0x0002	Identifies attribute hierarchies.	0x0004	Identifies key attribute hierarchies.	0x0008	Identifies attributes with no attribute hierarchies.
Value	Description											
0x0001	Identifies user defined hierarchies.											
0x0002	Identifies attribute hierarchies.											
0x0004	Identifies key attribute hierarchies.											
0x0008	Identifies attributes with no attribute hierarchies.											
HIERARCHY_DISPLAY_FOLDER	xsd:string	A user-defined path to be used when displaying the hierarchy in the user interface. Folder names are separated by a semicolon (;). Nested folders are indicated by a backslash (\).										

Name	Type	Description												
INSTANCE_SELECTION	xsd:unsignedShort	<p>A list of values to provide a hint to the client application on how to display the hierarchy values. Valid values include the following values:</p> <table><tr><th>Value</th><th>Description</th></tr><tr><td>0</td><td>NONE hint is suggested</td></tr><tr><td>1</td><td>DROPDOWN type of display is suggested</td></tr><tr><td>2</td><td>LIST type of display is suggested</td></tr><tr><td>3</td><td>FILTERED LIST type of display is suggested</td></tr><tr><td>4</td><td>MANDATORY FILTER type of display is suggested</td></tr></table>	Value	Description	0	NONE hint is suggested	1	DROPDOWN type of display is suggested	2	LIST type of display is suggested	3	FILTERED LIST type of display is suggested	4	MANDATORY FILTER type of display is suggested
Value	Description													
0	NONE hint is suggested													
1	DROPDOWN type of display is suggested													
2	LIST type of display is suggested													
3	FILTERED LIST type of display is suggested													
4	MANDATORY FILTER type of display is suggested													

The rowset is sorted on CATALOG_NAME, CUBE_NAME, DIMENSION_UNIQUE_NAME, and HIERARCHY_NAME.

3.1.4.2.3.5 MDSHEMA_LEVELS

This Discover element describes each level within a particular hierarchy.

3.1.4.2.3.5.1 Restrictions

The MDSHEMA_LEVELS rowset MAY be restricted on the columns listed in the following table.

Name	Type	Description
CATALOG_NAME	xsd:string	The name of the catalog.

Name	Type	Description										
SCHEMA_NAME	xsd:string	The name of the schema.										
CUBE_NAME	xsd:string	The name of the cube.										
DIMENSION_UNIQUE_NAME	xsd:string	The unique name of the dimension.										
HIERARCHY_UNIQUE_NAME	xsd:string	The unique name of the hierarchy.										
LEVEL_NAME	xsd:string	The name of the level.										
LEVEL_UNIQUE_NAME	xsd:string	The unique name of the level.										
LEVEL_ORIGIN	xsd:unsignedShort	<p>A bit mask that determines the source of the hierarchy.</p> <table><thead><tr><th>Value</th><th>Description</th></tr></thead><tbody><tr><td>0x0001</td><td>Identifies user defined hierarchies.</td></tr><tr><td>0x0002</td><td>Identifies attribute hierarchies.</td></tr><tr><td>0x0004</td><td>Identifies key attribute hierarchies.</td></tr><tr><td>0x0008</td><td>Identifies attributes with no attribute hierarchies.</td></tr></tbody></table> <p>The default restriction value is 0x00000003.</p>	Value	Description	0x0001	Identifies user defined hierarchies.	0x0002	Identifies attribute hierarchies.	0x0004	Identifies key attribute hierarchies.	0x0008	Identifies attributes with no attribute hierarchies.
Value	Description											
0x0001	Identifies user defined hierarchies.											
0x0002	Identifies attribute hierarchies.											
0x0004	Identifies key attribute hierarchies.											
0x0008	Identifies attributes with no attribute hierarchies.											

Name	Type	Description						
CUBE_SOURCE	xsd:unsignedShort	<p>A bitmask with one of these valid values:</p> <table><thead><tr><th>Value</th><th>Description</th></tr></thead><tbody><tr><td>0x01</td><td>Cube</td></tr><tr><td>0x02</td><td>Dimension</td></tr></tbody></table> <p>Default restriction is a value of 1.</p>	Value	Description	0x01	Cube	0x02	Dimension
Value	Description							
0x01	Cube							
0x02	Dimension							
LEVEL_VISIBILITY	xsd:unsignedShort	<p>A bitmask with one of the following valid values:</p> <table><thead><tr><th>Value</th><th>Description</th></tr></thead><tbody><tr><td>0x01</td><td>Visible</td></tr><tr><td>0x02</td><td>Not Visible</td></tr></tbody></table> <p>Default restriction is a value of 1.</p>	Value	Description	0x01	Visible	0x02	Not Visible
Value	Description							
0x01	Visible							
0x02	Not Visible							

3.1.4.2.3.5.2 Columns

The MDSCHEMA_LEVELS rowset contains the following columns.

Name	Type	Description
CATALOG_NAME	xsd:string	The name of the database to which this level belongs.
SCHEMA_NAME	xsd:string	The name of the schema to which this level belongs.
CUBE_NAME	xsd:string	The name of the cube to which this level belongs.
DIMENSION_UNIQUE_NAME	xsd:string	The unique name of the dimension to which this level belongs. < 28 >
HIERARCHY_UNIQUE_NAME	xsd:string	The unique name of the hierarchy.

Name	Type	Description
LEVEL_NAME	xsd:string	The name of the level.
LEVEL_UNIQUE_NAME	xsd:string	The unique name of the level.
LEVEL_GUID	uuid	The GUID of the level.
LEVEL_CAPTION	xsd:string	The caption of the hierarchy. Use this when displaying the name of the hierarchy to the user, such as in the user interface or reports. <29>
LEVEL_NUMBER	xsd:unsignedInt	The distance of the level from the root of the hierarchy. Root level is zero (0).
LEVEL_CARDINALITY	xsd:unsignedInt	The number of members in the level.

Name	Type	Description
LEVEL_TYPE	xsd:int	<p>The type of the level from a list of possible values.</p> <p>0x2001 Geography continent</p> <p>0x2002 Geography region</p> <p>0x2003 Geography country</p> <p>0x2004 Geography state or province</p> <p>0x2005 Geography county</p> <p>0x2006 Geography city</p> <p>0x2007 Geography postalcode</p> <p>0x2008 Geography point</p> <p>0x1011 Organization unit</p> <p>0x1012 Bill of materials resource</p> <p>0x1013 Quantitative</p> <p>0x1014 Account</p> <p>0x1021 Customer</p> <p>0x1022 Customer group</p> <p>0x1023 Customer household</p> <p>0x1031 Product</p> <p>0x1032 Product group</p> <p>0x1015 Scenario</p> <p>0x1016 Utility</p>

Name	Type	Description
		0x1041 Person 0x1042 Company 0x1051 Currency source 0x1052 Currency destination 0x1061 Channel 0x1062 Representative 0x1071 Promotion
DESCRIPTION	xsd:string	A human-readable description of the level.

Name	Type	Description												
CUSTOM_ROLLUP_SETTINGS	xsd:int	<div>A bitmask that specifies the custom rollup options. MUST be any combination of the following values:</div> <table><tr><th>Value</th><th>Description</th></tr><tr><td>0x01</td><td>Indicates an expression exists for this level.</td></tr><tr><td>0x02</td><td>Indicates that there is a custom rollup column for this level.</td></tr><tr><td>0x04</td><td>Indicates that there is a skipped level associated with members of this level.</td></tr><tr><td>0x08</td><td>Indicates that members of the level have custom member properties.</td></tr><tr><td>0x10</td><td>Indicates that members on the level have unary operators.</td></tr></table>	Value	Description	0x01	Indicates an expression exists for this level.	0x02	Indicates that there is a custom rollup column for this level.	0x04	Indicates that there is a skipped level associated with members of this level.	0x08	Indicates that members of the level have custom member properties.	0x10	Indicates that members on the level have unary operators.
Value	Description													
0x01	Indicates an expression exists for this level.													
0x02	Indicates that there is a custom rollup column for this level.													
0x04	Indicates that there is a skipped level associated with members of this level.													
0x08	Indicates that members of the level have custom member properties.													
0x10	Indicates that members on the level have unary operators.													
LEVEL_UNIQUE_SETTINGS	xsd:int	<div>A bitmask that specifies which columns contain unique values, if the level only has members with unique names or keys:</div> <table><tr><th>Value</th><th>Description</th></tr><tr><td>0x00000001</td><td>Member key columns establish uniqueness</td></tr><tr><td>0x00000002</td><td>Member name columns establish uniqueness</td></tr></table>	Value	Description	0x00000001	Member key columns establish uniqueness	0x00000002	Member name columns establish uniqueness						
Value	Description													
0x00000001	Member key columns establish uniqueness													
0x00000002	Member name columns establish uniqueness													
LEVEL_IS_VISIBLE	xsd:boolean	A Boolean that indicates whether the level is visible.												

Name	Type	Description
LEVEL_ORDERING_PROPERTY	xsd:string	The ID of the attribute that the level is sorted on.

Name	Type	Description
LEVEL_DBTYPE	xsd:int	<p>The type of the member key column that is used for the level attribute. MUST be null if concatenated keys are used as the member key column.</p> <p>Type values are described in the following table:</p> <p>0 Specifies no value was specified.</p> <p>2 Indicates a two-byte signed integer.</p> <p>3 Indicates a four-byte signed integer.</p> <p>4 Indicates a single-precision floating-point value.</p> <p>5 Indicates a double-precision floating-point value.</p> <p>6 Indicates a currency value. Currency is a fixed-point number with four digits to the right of the decimal point and is stored in an eight-byte signed integer scaled by 10,000.</p> <p>7 Indicates a date value. Date values are stored as Double, the whole part of which is the number of days since December 30, 1899, and the fractional part of which is the fraction of a day.</p> <p>8 Indicates a null-terminated character string (Unicode).</p>

Name	Type	Description
		<p>9 Indicates a pointer to an IDispatch interface on an OLE object. This data type is not currently supported by ADO, and usage of this data type may cause unpredictable results.</p> <p>10 Indicates a 32-bit error code.</p> <p>11 Indicates a Boolean value.</p> <p>12 Indicates an Automation Variant. This data type is not currently supported by ADO, and usage of this data type may cause unpredictable results.</p> <p>13 Indicates a pointer to an IUnknown interface on an OLE object. This data type is not currently supported by ADO, and usage of this data type may cause unpredictable results.</p> <p>14 Indicates an exact numeric value with a fixed precision and scale.</p> <p>16 Indicates a one-byte signed integer.</p> <p>17 Indicates a one-byte unsigned integer.</p> <p>18 Integer indicates a two-byte unsigned integer.</p> <p>19 Indicates a four-byte unsigned integer.</p> <p>20 Indicates an eight-byte signed integer.</p>

Name	Type	Description
		<p>21 Indicates an eight-byte unsigned integer.</p> <p>64 Indicates a 64-bit value representing the number of 100-nanosecond intervals since January 1, 1601.</p> <p>72 Indicates a globally unique identifier (GUID).</p> <p>128 Indicates a binary value.</p> <p>129 Indicates a String value.</p> <p>130 Indicates a null-terminated Unicode character string.</p> <p>131 Indicates an exact numeric value with a fixed precision and scale.</p> <p>131 Indicates an Automation PROPVARIANT.</p> <p>132 Indicates a user-defined variable.</p> <p>133 Indicates a date value (yyyymmdd).</p> <p>134 Indicates a time value (hhmmss).</p> <p>135 Indicates a date-time stamp (yyyymmddhhmmss plus a fraction in billionths).</p>

Name	Type	Description
		136 Indicates a four-byte chapter value used to identify rows in a child rowset.
LEVEL_MASTER_UNIQUE_NAME	xsd:string	The unique name of the level.
LEVEL_NAME_SQL_COLUMN_NAME	xsd:string	The data source representation of the level member names.
LEVEL_KEY_SQL_COLUMN_NAME	xsd:string	The data source representation of the level member key values.
LEVEL_UNIQUE_NAME_SQL_COLUMN_NAME	xsd:string	The data source representation of the member unique names.
LEVEL_ATTRIBUTE_HIERARCHY_NAME	xsd:string	The name of the attribute hierarchy providing the source of the level.
LEVEL_KEY_CARDINALITY	xsd:unsignedShort	The number of columns in the level key.

Name	Type	Description
LEVEL_ORIGIN	xsd:unsignedShort	<p>A bitmask that defines how the level was sourced:</p> <ul style="list-style-type: none"> • 0x00000001 Identifies levels in a user defined hierarchy. • 0x00000002 Identifies levels in an attribute hierarchy. • 0x00000004 Identifies levels in a key attribute hierarchy. • 0x00000008 Identifies levels in attribute hierarchies that are not enabled.

The rowset is sorted on CATALOG_NAME, CUBE_NAME, DIMENSION_UNIQUE_NAME, and HIERARCHY_NAME.

3.1.4.2.3.6 MDSCHEMA_MEASURES

This Discover element describes each measure within a cube.

3.1.4.2.3.6.1 Restrictions

The MDSCHEMA_MEASURES rowset MAY be restricted on the columns listed in the following table.

Name	Type	Description
CATALOG_NAME	xsd:string	The name of the catalog.
SCHEMA_NAME	xsd:string	The name of the schema.
CUBE_NAME	xsd:string	The name of the cube.

Name	Type	Description
MEASURE_NAME	xsd:string	The name of the measure.
MEASURE_UNIQUE_NAME	xsd:string	The unique name of the measure.
MEASUREGROUP_NAME	xsd:string	The name of the measure group.
CUBE_SOURCE	xsd:unsignedShort	A bitmask with one of the following valid values: 1 CUBE 2 DIMENSION Default restriction is a value of 1.
MEASURE_VISIBILITY	xsd:unsignedShort	A bitmask with one of the following valid values: 1 Visible 2 Not Visible Default restriction is a value of 1.

3.1.4.2.3.6.2 Columns

The MDSCHEMA_MEASURES rowset contains the following columns.

Name	Type	Description
CATALOG_NAME	xsd:string	The name of the catalog to which this measure belongs.
SCHEMA_NAME	xsd:string	The name of the schema to which this measure belongs.

Name	Type	Description
CUBE_NAME	xsd:string	The name of the cube to which this measure belongs.
MEASURE_NAME	xsd:string	The name of the measure.
MEASURE_UNIQUE_NAME	xsd:string	The unique name of the measure. For providers that generate unique names by qualification, each component of this name is delimited.
MEASURE_CAPTION	xsd:string	A label or caption associated with the measure. Used primarily for display purposes. If a caption does not exist, MEASURE_NAME is returned.
MEASURE_GUID	uuid	The GUID of the measure.

Name	Type	Description																		
MEASURE_AGGREGATOR	xsd:int	<p>An enumeration that identifies how a measure was derived. Can be one of the following values:</p> <table><tr><th>Value</th><th>Description</th></tr><tr><td>1</td><td>MEASURE aggregates from SUM.</td></tr><tr><td>2</td><td>MEASURE aggregates from COUNT.</td></tr><tr><td>3</td><td>MEASURE aggregates from MIN.</td></tr><tr><td>4</td><td>MEASURE aggregates from MAX.</td></tr><tr><td>5</td><td>MEASURE aggregates from AVG.</td></tr><tr><td>6</td><td>MEASURE aggregates from VAR.</td></tr><tr><td>7</td><td>Identifies that the measure was derived from an aggregation function that was SUM, COUNT, MIN, MAX, AVG, VAR, or STDEV, respectively.</td></tr><tr><td>127</td><td>Identifies that the measure was derived from a formula that was not any single function above.</td></tr></table>	Value	Description	1	MEASURE aggregates from SUM.	2	MEASURE aggregates from COUNT.	3	MEASURE aggregates from MIN.	4	MEASURE aggregates from MAX.	5	MEASURE aggregates from AVG.	6	MEASURE aggregates from VAR.	7	Identifies that the measure was derived from an aggregation function that was SUM, COUNT, MIN, MAX, AVG, VAR, or STDEV, respectively.	127	Identifies that the measure was derived from a formula that was not any single function above.
Value	Description																			
1	MEASURE aggregates from SUM.																			
2	MEASURE aggregates from COUNT.																			
3	MEASURE aggregates from MIN.																			
4	MEASURE aggregates from MAX.																			
5	MEASURE aggregates from AVG.																			
6	MEASURE aggregates from VAR.																			
7	Identifies that the measure was derived from an aggregation function that was SUM, COUNT, MIN, MAX, AVG, VAR, or STDEV, respectively.																			
127	Identifies that the measure was derived from a formula that was not any single function above.																			

Name	Type	Description
		0 Identifies that the measure was derived from an unknown aggregation function or formula.

Name	Type	Description
DATA_TYPE	xsd:int	<p>The type of the member key column that is used for the level attribute. MUST be null if concatenated keys are used as the member key column.</p> <p>Type values are described in the following table:</p> <p>0 Specifies no value was specified.</p> <p>2 Indicates a two-byte signed integer.</p> <p>3 Indicates a four-byte signed integer.</p> <p>4 Indicates a single-precision floating-point value.</p> <p>5 Indicates a double-precision floating-point value.</p> <p>6 Indicates a currency value. Currency is a fixed-point number with four digits to the right of the decimal point and is stored in an eight-byte signed integer scaled by 10,000.</p> <p>7 Indicates a date value. Date values are stored as Double, the whole part of which is the number of days since December 30, 1899, and the fractional part of which is the fraction of a day.</p> <p>8 Indicates a null-terminated character string (Unicode).</p>

Name	Type	Description
		<p>9 Indicates a pointer to an IDispatch interface on an OLE object. This data type is not currently supported by ADO, and usage of this data type may cause unpredictable results.</p> <p>10 Indicates a 32-bit error code.</p> <p>11 Indicates a Boolean value.</p> <p>12 Indicates an Automation Variant. This data type is not currently supported by ADO, and usage of this data type may cause unpredictable results.</p> <p>13 Indicates a pointer to an IUnknown interface on an OLE object. This data type is not currently supported by ADO, and usage of this data type may cause unpredictable results.</p> <p>14 Indicates an exact numeric value with a fixed precision and scale.</p> <p>16 Indicates a one-byte signed integer.</p> <p>17 Indicates a one-byte unsigned integer.</p> <p>18 Integer indicates a two-byte unsigned integer.</p> <p>19 Indicates a four-byte unsigned integer.</p> <p>20 Indicates an eight-byte signed integer.</p>

Name	Type	Description
		<p>21 Indicates an eight-byte unsigned integer.</p> <p>64 Indicates a 64-bit value representing the number of 100-nanosecond intervals since January 1, 1601.</p> <p>72 Indicates a globally unique identifier (GUID).</p> <p>128 Indicates a binary value.</p> <p>129 Indicates a String value.</p> <p>130 Indicates a null-terminated Unicode character string.</p> <p>131 Indicates an exact numeric value with a fixed precision and scale.</p> <p>131 Indicates an Automation PROPVARIANT.</p> <p>132 Indicates a user-defined variable.</p> <p>133 Indicates a date value (yyyymmdd).</p> <p>134 Indicates a time value (hhmmss).</p> <p>135 Indicates a date-time stamp (yyyymmddhhmmss plus a fraction in billionths).</p>

Name	Type	Description
		136 Indicates a four-byte chapter value used to identify rows in a child rowset.
NUMERIC_PRECISION	xsd:unsignedShort	The maximum precision of the property if the measure object's data type is numeric, decimal or datetime. NULL for all other property types.
NUMERIC_SCALE	xsd:short	The number of digits to the right of the decimal point if the measure object's type indicator is numeric, decimal or datetime. Otherwise, this value is NULL.
MEASURE_UNITS	xsd:string	Not supported.
DESCRIPTION	xsd:string	A human-readable description of the measure. NULL if no description exists.
EXPRESSION	xsd:string	An expression for the member.
MEASURE_IS_VISIBLE	xsd:boolean	A Boolean that always returns TRUE. If the measure is not visible, it will not be included in the schema rowset.
LEVELS_LIST	xsd:string	A string that always returns NULL.
MEASURE_NAME_SQL_COLUMN_NAME	xsd:string	The name of the column in the SQL query that corresponds to the measure's name.
MEASURE_UNQUALIFIED_CAPTION	xsd:string	The name of the measure, not qualified with the measure group name.

Name	Type	Description
MEASUREGROUP_NAME	xsd:string	The name of the measure group to which the measure belongs.
MEASURE_DISPLAY_FOLDER	xsd:string	The path to be used when displaying the measure in the user interface. <30>
DEFAULT_FORMAT_STRING	xsd:string	The default format string for the measure.

The rowset is sorted on CATALOG_NAME, CUBE_NAME, and DIMENSION_NAME.

3.1.4.2.3.7 MDSHEMA_PROPERTIES

This Discover element describes the properties of members within a database.

3.1.4.2.3.7.1 Restrictions

The MDSHEMA_PROPERTIES rowset MAY be restricted on the columns listed in the following table.

Name	Type	Description
CATALOG_NAME	xsd:string	The name of the catalog.
SCHEMA_NAME	xsd:string	The name of the schema.
CUBE_NAME	xsd:string	The name of the cube.
DIMENSION_UNIQUE_NAME	xsd:string	The unique name of the dimension.
HIERARCHY_UNIQUE_NAME	xsd:string	The unique name of the hierarchy.
LEVEL_UNIQUE_NAME	xsd:string	The unique name of the level.

Name	Type	Description
MEMBER_UNIQUE_NAME	xsd:string	The unique name of the member.
PROPERTY_NAME	xsd:string	The name of the property.
PROPERTY_TYPE	xsd:short	The type of the property.
PROPERTY_CONTENT_TYPE	xsd:short	A default restriction is in place with value of 0x03 that stands for Member or Cell properties.
PROPERTY_ORIGIN	xsd:unsignedShort	A default restriction is in place with value of 0x03 that stands for user defined or system enabled origin.
CUBE_SOURCE	xsd:unsignedShort	A bitmask with one of the following valid values: 1 CUBE 2 DIMENSION Default restriction is a value of 1.
PROPERTY_VISIBILITY	xsd:unsignedShort	A bitmask with one of the following valid values: 1 Visible 2 Not visible Default restriction is a value of 1.

3.1.4.2.3.7.2 Columns

The MDSCHEMA_PROPERTIES rowset contains the following columns.

90 of 178

Name	Type	Description
CATALOG_NAME	xsd:string	The name of the database.
SCHEMA_NAME	xsd:string	The name of the schema to which this property belongs. NULL if the provider does not support schemas.
CUBE_NAME	xsd:string	The name of the cube.
DIMENSION_UNIQUE_NAME	xsd:string	The unique name of the dimension. For providers that generate unique names by qualification, each component of this name is delimited.
HIERARCHY_UNIQUE_NAME	xsd:string	The unique name of the hierarchy. For providers that generate unique names by qualification, each component of this name is delimited.
LEVEL_UNIQUE_NAME	xsd:string	The unique name of the level to which this property belongs. If the provider does not support named levels, it returns the DIMENSION_UNIQUE_NAME value for this field. For providers that generate unique names by qualification, each component of this name is delimited.
MEMBER_UNIQUE_NAME	xsd:string	The unique name of the member to which the property belongs. Used for data stores that do not support named levels or have properties on a member - by- member basis. If the property applies to all members in a level, this column is NULL. For providers that generate unique names by qualification, each component of this name is delimited.

Name	Type	Description
PROPERTY_TYPE	xsd:short	<p>A bitmask that specifies the type of the property:</p> <p>1 Identifies a property of a member.</p> <p>2 Identifies a property of a cell.</p> <p>4 Identifies an internal property.</p> <p>8 Identifies a property which contains a binary large object (BLOB).</p>
PROPERTY_NAME	xsd:string	The name of the property. <31>
PROPERTY_CAPTION	xsd:string	A label or caption associated with the property, used primarily for display purposes. <32>

Name	Type	Description
DATA_TYPE	xsd:int	<p>The type of the member key column that is used for the level attribute. MUST be null if concatenated keys are used as the member key column.</p> <p>Type values are described in the following table:</p> <p>0 Specifies no value was specified.</p> <p>2 Indicates a two-byte signed integer.</p> <p>3 Indicates a four-byte signed integer.</p> <p>4 Indicates a single-precision floating-point value.</p> <p>5 Indicates a double-precision floating-point value.</p> <p>6 Indicates a currency value. Currency is a fixed-point number with four digits to the right of the decimal point and is stored in an eight-byte signed integer scaled by 10,000.</p> <p>7 Indicates a date value. Date values are stored as Double, the whole part of which is the number of days since December 30, 1899, and the fractional part of which is the fraction of a day.</p> <p>8 Indicates a null-terminated character string (Unicode).</p>

Name	Type	Description
		<p>9 Indicates a pointer to an IDispatch interface on an OLE object. This data type is not currently supported by ADO, and usage of this data type may cause unpredictable results.</p> <p>10 Indicates a 32-bit error code.</p> <p>11 Indicates a Boolean value.</p> <p>12 Indicates an Automation Variant. This data type is not currently supported by ADO, and usage of this data type may cause unpredictable results.</p> <p>13 Indicates a pointer to an IUnknown interface on an OLE object. This data type is not currently supported by ADO, and usage of this data type may cause unpredictable results.</p> <p>14 Indicates an exact numeric value with a fixed precision and scale.</p> <p>16 Indicates a one-byte signed integer.</p> <p>17 Indicates a one-byte unsigned integer.</p> <p>18 Integer indicates a two-byte unsigned integer.</p> <p>19 Indicates a four-byte unsigned integer.</p> <p>20 Indicates an eight-byte signed integer.</p>

Name	Type	Description
		<p>21 Indicates an eight-byte unsigned integer.</p> <p>64 Indicates a 64-bit value representing the number of 100-nanosecond intervals since January 1, 1601.</p> <p>72 Indicates a globally unique identifier (GUID).</p> <p>128 Indicates a binary value.</p> <p>129 Indicates a String value.</p> <p>130 Indicates a null-terminated Unicode character string.</p> <p>131 Indicates an exact numeric value with a fixed precision and scale.</p> <p>131 Indicates an Automation PROPVARIANT.</p> <p>132 Indicates a user-defined variable.</p> <p>133 Indicates a date value (yyyymmdd).</p> <p>134 Indicates a time value (hhmmss).</p> <p>135 Indicates a date-time stamp (yyyymmddhhmmss plus a fraction in billionths).</p>

Name	Type	Description
		136 Indicates a four-byte chapter value used to identify rows in a child rowset.
CHARACTER_MAXIMUM_LENGTH	xsd:unsignedInt	<p>The maximum possible length of the property, if it is a character, binary, or bit type.</p> <p>Zero indicates there is no defined maximum length.</p> <p>Returns NULL for all other data types.</p>
CHARACTER_OCTET_LENGTH	xsd:unsignedInt	<p>The maximum possible length (in bytes) of the property, if it is a character or binary type.</p> <p>Zero indicates there is no defined maximum length.</p> <p>Returns NULL for all other data types.</p>
NUMERIC_PRECISION	xsd:unsignedShort	The maximum precision of the property if the measure object's data type is numeric, decimal or datetime. NULL for all other property types.
NUMERIC_SCALE	xsd:short	The number of digits to the right of the decimal point if the measure object's type indicator is numeric, decimal or datetime. Otherwise, this value is NULL.
DESCRIPTION	xsd:string	A human readable description of the property. NULL if no description exists.
PROPERTY_CONTAINER_TYPE	xsd:short	0xB4 Date canceled

Name	Type	Description
PROPERTY_CONTENT_TYPE	xsd:short	<p>The content type of the property.</p> <p>A user extensible enumeration. User extended values SHOULD NOT override default values as the server would not differentiate them.</p> <p>Default values listed here:</p> <p>0x00 Regular</p> <p>0x01 Id</p> <p>0x02 Relation to parent</p> <p>0x03 Rollup operator</p> <p>0x11 Organization title</p> <p>0x21 Caption</p> <p>0x22 Caption short</p> <p>0x23 Caption description</p> <p>0x24 Caption abbreviation</p> <p>0x31 Web url</p> <p>0x32 Web html</p> <p>0x33 Web xml or xsl</p> <p>0x34 Web mail alias</p> <p>0x41 Address</p> <p>0x42 Address street</p> <p>0x43 Address house</p> <p>0x44 Address city</p> <p>0x45 Address state or province</p>

97 of 178

Name	Type	Description
		0x46 Address zip
		0x47 Address quarter
		0x48 Address country
		0x49 Address building
		0x4A Address room
		0x4B Address floor
		0x4C Address fax
		0x4D Address phone
		0x61 Geography centroid x
		0x62 Geography centroid y
		0x63 Geography centroid z
		0x64 Geography boundary top
		0x65 Geography boundary left
		0x66 Geography boundary bottom
		0x67 Geography boundary right
		0x68 Geography boundary front
		0x69 Geography boundary rear
		0x6A Geography boundary polygon
		0x71 Physical size
		0x72 Physical color

Name	Type	Description
		0x73 Physical weight
		0x74 Physical height
		0x75 Physical width
		0x76 Physical depth
		0x77 Physical volume
		0x78 Physical density
		0x82 Person full name
		0x83 Person first name
		0x84 Person last name
		0x85 Person middle name
		0x86 Person demographic
		0x87 Person contact
		0x91 Quantity range low
		0x92 Quantity range high
		0xA1 Formatting color
		0xA2 Formatting order
		0xA3 Formatting font
		0xA4 Formatting font effects
		0xA5 Formatting font size
		0xA6 Formatting sub total

Name	Type	Description
		0xB1 Date 0xB2 Date start 0xB3 Date ended 0xB4 Date canceled 0xB5 Date modified 0xB6 Date duration 0xC1 Version
SQL_COLUMN_NAME	xsd:string	The name of the property used in SQL queries from the cube dimension or database dimension.
LANGUAGE	xsd:unsignedShort	The translation expressed as an LCID. Only valid for property translations.
PROPERTY_ORIGIN	xsd:unsignedShort	A bitmask that specifies the type of hierarchy that the property applies to: 1 Indicates the property is on a user defined hierarchy 2 Indicates the property is on an attribute hierarchy 4 Indicates the property is on a key attribute hierarchy 8 Indicates the property is on an attribute hierarchy that is not enabled.
PROPERTY_ATTRIBUTE_HIERARCHY_NAME	xsd:string	The name of the attribute hierarchy sourcing this property.

Name	Type	Description
PROPERTY_CARDINALITY	xsd:string	The cardinality of the property. Possible values include the following strings: ONE MANY
MIME_TYPE	xsd:string	The mime type for binary large objects (BLOBs).
PROPERTY_IS_VISIBLE	xsd:boolean	A Boolean that indicates whether the property is visible.

This schema rowset is not sorted.

3.1.4.2.3.8 MDSHEMA_MEMBERS

This Discover element describes the members within a database.

3.1.4.2.3.8.1 Restrictions

The MDSHEMA_MEMBERS rowset MAY be restricted on the columns listed in the following table.

Name	Type	Description
CATALOG_NAME	xsd:string	The name of the catalog.
SCHEMA_NAME	xsd:string	The name of the schema.
CUBE_NAME	xsd:string	The name of the cube.
DIMENSION_UNIQUE_NAME	xsd:string	The unique name of the dimension.
HIERARCHY_UNIQUE_NAME	xsd:string	The unique name of the hierarchy.

Name	Type	Description
LEVEL_UNIQUE_NAME	xsd:string	The unique name of the level.
LEVEL_NUMBER	xsd:unsignedInt	The number of the level.
MEMBER_NAME	xsd:string	The name of the member.
MEMBER_UNIQUE_NAME	xsd:string	The unique name of the member.
MEMBER_CAPTION	xsd:string	The caption of the member.
MEMBER_TYPE	xsd:int	The type of the member.
TREE_OP	xsd:int	<p>Only applies to a single member:</p> <p>0x20 Returns all of the ancestors.</p> <p>0x01 Returns only the immediate children.</p> <p>0x02 Returns members on the same level.</p> <p>0x04 Returns only the immediate parent.</p> <p>0x08 Returns itself in the list of returned rows.</p> <p>0x10 Returns all of the descendants.</p>

Name	Type	Description
CUBE_SOURCE	xsd:unsignedShort	<p>A bitmask with one of the following valid values:</p> <p>1 CUBE</p> <p>2 DIMENSION</p> <p>Default restriction is a value of 1.</p>

3.1.4.2.3.8.2 Columns

The MDSCHEMA_MEMBERS rowset contains the following columns.

Name	Type	Description
CATALOG_NAME	xsd:string	The name of the database to which this member belongs.
SCHEMA_NAME	xsd:string	The name of the schema to which this member belongs.
CUBE_NAME	xsd:string	The name of the cube to which this member belongs.
DIMENSION_UNIQUE_NAME	xsd:string	The unique name of the dimension to which this member belongs.
HIERARCHY_UNIQUE_NAME	xsd:string	The unique name of the hierarchy to which this member belongs.
LEVEL_UNIQUE_NAME	xsd:string	The unique name of the level to which this member belongs.
LEVEL_NUMBER	xsd:unsignedInt	The distance of the member from the root of the hierarchy. The root level is zero (0).

Name	Type	Description												
MEMBER_ORDINAL	xsd:unsignedInt	The ordinal of the member in its level.												
MEMBER_NAME	xsd:string	The name of the member.												
MEMBER_UNIQUE_NAME	xsd:string	The unique name of the member.												
MEMBER_TYPE	xsd:int	<div>The type of the member:<33></div> <table><tr><th>Value</th><th>Description</th></tr><tr><td>1</td><td>Is a regular member</td></tr><tr><td>2</td><td>Is the All member</td></tr><tr><td>3</td><td>is a measure</td></tr><tr><td>4</td><td>Is a formula</td></tr><tr><td>0</td><td>Is unknown type</td></tr></table>	Value	Description	1	Is a regular member	2	Is the All member	3	is a measure	4	Is a formula	0	Is unknown type
Value	Description													
1	Is a regular member													
2	Is the All member													
3	is a measure													
4	Is a formula													
0	Is unknown type													
MEMBER_GUID	uuid	The GUID of the member. NULL if no GUID exists.												
MEMBER_CAPTION	xsd:string	A label or caption associated with the member. Used primarily for display purposes. If a caption does not exist, MEMBER_NAME is returned.												
CHILDREN_CARDINALITY	xsd:unsignedInt	The number of children that the member has. This can be an estimate. Providers return the best estimate possible.												

Name	Type	Description
PARENT_LEVEL	xsd:unsignedInt	The distance of the member 's parent from the root level of the hierarchy. The root level is zero (0).
PARENT_UNIQUE_NAME	xsd:string	The unique name of the member 's parent. NULL is returned for any members at the root level.
PARENT_COUNT	xsd:unsignedInt	The number of parents that this member has.
DESCRIPTION	xsd:string	This column always returns a NULL value. This column exists for backwards compatibility.
EXPRESSION	xsd:string	The expression for calculations, if the member is of type 4 (formula).
MEMBER_KEY	xsd:string	The value of the member 's key column. Returns NULL if the member has a composite key.
IS_PLACEHOLDER_MEMBER	xsd:boolean	A Boolean that indicates whether a member is a placeholder member for an empty position in a dimension hierarchy. It is valid only if the MDX Compatibility property has been set to 1.
IS_DATAMEMBER	xsd:boolean	A Boolean that indicates whether the member is a data member. Returns TRUE if the member is a data member.

Name	Type	Description						
SCOPE	xsd:int	<p>The scope of the member. The member can be a session -calculated member or global-calculated member. The column returns NULL for non-calculated members.</p> <p>This column can have one of the following values:</p> <table><tr><th>Value</th><th>Description</th></tr><tr><td>1</td><td>Global</td></tr><tr><td>2</td><td>Session</td></tr></table>	Value	Description	1	Global	2	Session
Value	Description							
1	Global							
2	Session							
Zero or more additional columns	xsd:unsignedShort	<p>No properties are returned if the members could be returned from multiple levels. For example, if the Tree operator is PARENT and SELF for a non-parent child hierarchy, no member properties are returned.</p> <p>This applies to ragged hierarchies where tree operators could return members from different levels (for example, if the prior level contains holes and parent on members is requested).</p>						

The rowset is sorted on CATALOG_NAME, CUBE_NAME, DIMENSION_UNIQUE_NAME, HIERARCHY_UNIQUE_NAME, LEVEL_UNIQUE_NAME, LEVEL_NUMBER, MEMBER_ORDINAL.

3.1.4.2.3.9 MDSCHEMA_Actions

This Discover element describes the actions that may be available to the client application.

3.1.4.2.3.9.1 Restrictions

The MDSCHEMA_ACTIONS rowset can be restricted on the columns listed in the following table.

Name	Type	Description
CATALOG_NAME	xsd:string	The name of the catalog.
SCHEMA_NAME	xsd:string	The name of the schema.
CUBE_NAME	xsd:string	Mandatory.
ACTION_NAME	xsd:string	The name of the action.
ACTION_TYPE	xsd:int	The type of the action.
COORDINATE	xsd:string	Mandatory.
COORDINATE_TYPE	xsd:int	Mandatory.
INVOCATION	xsd:int	<p>The INVOCATION restriction column defaults to the value of 1.</p> <p>To retrieve the actions for all invocation types that are defined for the specified coordinate, coordinate_type and cube restrictions, use a value of zero (0) in the INVOCATION restriction column.</p>
CUBE_SOURCE	xsd:unsignedShort	<p>A bitmask with one of the following valid values:</p> <p>1 CUBE</p> <p>2 DIMENSION</p> <p>Default restriction is a value of 1.</p>

Client applications can define more than one ACTION_TYPE by using the by a bitwise OR operation over the different ACTION_TYPE values and passing the resulting value in the restriction list..

3.1.4.2.3.9.2 Columns

The MDSCHEMA_ACTIONS rowset contains the following columns.

Name	Type	Description
CATALOG_NAME	xsd:string	The name of the database.
SCHEMA_NAME	xsd:string	The name of the schema to which this action belongs.
CUBE_NAME	xsd:string	The name of the cube to which this action belongs.
ACTION_NAME	xsd:string	The name of this action.

Name	Type	Description																				
ACTION_TYPE	xsd:int	<p>A bitmask that is used to specify the action's triggering method. The Msmd.h file defines the following bit value constants for this bitmask:</p> <table><tr><th>Value</th><th>Description</th></tr><tr><td>0x01</td><td>Action type is URL</td></tr><tr><td>0x02</td><td>Action type is HTML</td></tr><tr><td>0x04</td><td>Action type is statement</td></tr><tr><td>0x08</td><td>Action type is dataset</td></tr><tr><td>0x10</td><td>Action type is rowset</td></tr><tr><td>0x20</td><td>Action type is commandline</td></tr><tr><td>0x40</td><td>Action type is proprietary</td></tr><tr><td>0x80</td><td>Action type is report</td></tr><tr><td>0x100</td><td>Action type is drillthrough</td></tr></table>	Value	Description	0x01	Action type is URL	0x02	Action type is HTML	0x04	Action type is statement	0x08	Action type is dataset	0x10	Action type is rowset	0x20	Action type is commandline	0x40	Action type is proprietary	0x80	Action type is report	0x100	Action type is drillthrough
Value	Description																					
0x01	Action type is URL																					
0x02	Action type is HTML																					
0x04	Action type is statement																					
0x08	Action type is dataset																					
0x10	Action type is rowset																					
0x20	Action type is commandline																					
0x40	Action type is proprietary																					
0x80	Action type is report																					
0x100	Action type is drillthrough																					
COORDINATE	xsd:string	<p>A MDX expression that specifies an object or a coordinate in the multidimensional space in which the action is performed. It is the responsibility of the client application to provide the value of this restriction column.</p> <p>The COORDINATE must resolve to the object specified in COORDINATE_TYPE.</p>																				

Name	Type	Description														
COORDINATE_TYPE	xsd:int	<p>An enumeration that specifies how the COORDINATE restriction column is interpreted. The possible values are listed below:</p> <table><tr><th>Value</th><th>Description</th></tr><tr><td>1</td><td>Action coordinate refers to the cube</td></tr><tr><td>2</td><td>Action coordinate refers to a dimension</td></tr><tr><td>3</td><td>Action coordinate refers to a level</td></tr><tr><td>4</td><td>Action coordinate refers to a member</td></tr><tr><td>5</td><td>Action coordinate refers to a set</td></tr><tr><td>6</td><td>Action coordinate refers to a cell</td></tr></table>	Value	Description	1	Action coordinate refers to the cube	2	Action coordinate refers to a dimension	3	Action coordinate refers to a level	4	Action coordinate refers to a member	5	Action coordinate refers to a set	6	Action coordinate refers to a cell
Value	Description															
1	Action coordinate refers to the cube															
2	Action coordinate refers to a dimension															
3	Action coordinate refers to a level															
4	Action coordinate refers to a member															
5	Action coordinate refers to a set															
6	Action coordinate refers to a cell															

Name	Type	Description
ACTION_CAPTION	xsd:string	<p>The action name if no caption was specified and no translations were specified when the action was created or altered.</p> <p>At creation time, or when altering the action the user has the option of setting the property CaptionIsMDX and multiple translation properties for the action. These properties are not visible in MDSCHEMA_ACTIONS rowset but affect the outcome of it.</p> <p>If a caption or translations were specified, and CaptionIsMDX is FALSE, one of the following strings is returned:</p> <ul style="list-style-type: none"> • The translation for the appropriate language. • The specified caption if no translation was found for the specified language. • The action name if no translation was found and the caption was not specified in the DDL. <p>If a caption or translations were specified, and CaptionIsMDX is TRUE, the string resulting from finding the appropriate translation for the specified language or the specified translation in the DDL caption, and calculating the formula to create the string.</p> <p>If the action was specified in MDX Script, there are no translations and the caption is always treated as an MDX expression.</p>

Name	Type	Description
DESCRIPTION	xsd:string	A user-friendly description of the action.
CONTENT	xsd:string	The expression or content of the action that is to be run.
APPLICATION	xsd:string	The name of the application that is to be used to run the action.
INVOCATION	xsd:int	Information about how to invoke the action: <ul style="list-style-type: none"> • 1 Indicates a regular action used during normal operations. This is the default value for this column. • 2 Indicates to perform the action when the cube is first opened. • 4 Indicates that the action is performed as part of a batch operation.

The rowset is sorted on CATALOG_NAME, CUBE_NAME, ACTION_NAME. Actions of MDACTION_TYPE_PROPRIETARY type must provide a value for the APPLICATION column.

3.1.4.2.3.9.3 Remarks

The following table lists the valid **COORDINATE** and **COORDINATE_TYPE** combinations.

COORDINATE object type	COORDINATE_TYPE
Cube	MDACTION_COORDINATE_CUBE

COORDINATE object type	COORDINATE_TYPE
Dimension	MDACTION_COORDINATE_DIMENSION MDACTION_COORDINATE_LEVEL MDACTION_COORDINATE_MEMBER MDACTION_COORDINATE_SET MDACTION_COORDINATE_CELL
Hierarchy	MDACTION_COORDINATE_DIMENSION
Level	MDACTION_COORDINATE_LEVEL
Member	MDACTION_COORDINATE_MEMBER
Set	MDACTION_COORDINATE_SET
cell	MDACTION_COORDINATE_CELL

3.1.4.2.3.10 MDSHEMA_SETS

This Discover element describes any sets that are currently defined in a database, including session-scoped sets.

3.1.4.2.3.10.1 Restrictions

The MDSHEMA_SETS rowset can be restricted on the columns listed in the following table.

Name	Type	Description
CATALOG_NAME	xsd:string	The name of the catalog.
SCHEMA_NAME	xsd:string	The name of the schema.

Name	Type	Description						
CUBE_NAME	xsd:string	The name of the cube.						
SET_NAME	xsd:string	The name of the set.						
SCOPE	xsd:int	<p>The scope of the member. The member can be a session calculated member or global calculated member. The column returns NULL for non-calculated members.</p> <p>This column can have one of the following values:</p> <table><tr><th>Value</th><th>Description</th></tr><tr><td>1</td><td>Global</td></tr><tr><td>2</td><td>Session</td></tr></table>	Value	Description	1	Global	2	Session
Value	Description							
1	Global							
2	Session							
HIERARCHY_UNIQUE_NAME	xsd:string	The unique name of the hierarchy.						
CUBE_SOURCE	xsd:unsignedShort	<p>Optional.</p> <p>Note that only one hierarchy can be included, and only those named sets whose hierarchies exactly match the restriction are returned.</p>						

3.1.4.2.3.10.2 Columns

The MDSCHEMA_SETS rowset contains the following columns.

Name	Type	Description
CATALOG_NAME	xsd:string	The name of the database.

Name	Type	Description						
SCHEMA_NAME	xsd:string	The name of the schema.						
CUBE_NAME	xsd:string	The name of the cube.						
SET_NAME	xsd:string	The name of the set, as specified in the CREATE SET statement.						
SCOPE	xsd:int	<p>The scope of the set:</p> <p>The scope of the set. The set can be a session defined set or global defined set.</p> <p>This column can have one of the following values:</p> <table><tr><td>Value</td><td>Description</td></tr><tr><td>1</td><td>Global</td></tr><tr><td>2</td><td>Session</td></tr></table>	Value	Description	1	Global	2	Session
Value	Description							
1	Global							
2	Session							
DESCRIPTION	xsd:string	A description of the set.						
EXPRESSION	xsd:string	The expression for the set.						
DIMENSIONS	xsd:string	A comma delimited list of hierarchies included in the set.						
SET_CAPTION	xsd:string	A label or caption associated with the set. The label or caption is used primarily for display purposes.						

Name	Type	Description
SET_DISPLAY_FOLDER	xsd:string	A string that identifies the path of the display folder that the client application uses to show the set. The folder level separator is defined by the client application. <34>
SET_EVALUATION_CONTEXT	xsd:int	<p>The context for the set. The set can be static or dynamic.</p> <p>This column can have one of the following values:</p> <ul style="list-style-type: none"> • 1 STATIC • 2 DYNAMIC

The rowset is sorted on CATALOG_NAME and CUBE_NAME.

3.1.4.2.3.11 DISCOVER_INSTANCES

This Discover element describes the instances on the server.

3.1.4.2.3.11.1 Restrictions

The DISCOVER_INSTANCES rowset can be restricted on the columns listed in the following table.

Name	Type	Description
INSTANCE_NAME	xsd:string	The name of the instance.

3.1.4.2.3.11.2 Columns

The DISCOVER_INSTANCES rowset contains the following columns.

Name	Type	Description
INSTANCE_NAME	xsd:string	The name of the instance.

Name	Type	Description
INSTANCE_PORT_NUMBER	xsd:int	The port number the instance listens on.
INSTANCE_STATE	xsd:int	The state of the server instance: <ul style="list-style-type: none"> Started Stopped Starting Stopping Paused

This schema rowset is not sorted.

3.1.4.2.3.12 MDSCHEMA_KPIS

This Discover element describes the KPIs within a database.

3.1.4.2.3.12.1 Restrictions

The MDSCHEMA_KPIS rowset can be restricted on the columns listed in the following table.

Name	Type	Restriction
CATALOG_NAME	xsd:string	The name of the catalog.
SCHEMA_NAME	xsd:string	The name of the schema.
CUBE_NAME	xsd:string	The name of the cube.
KPI_NAME	xsd:string	The name of the KPI.

Name	Type	Restriction
CUBE_SOURCE	xsd:unsignedShort	<p>A bitmask with one of the following valid values:</p> <p>1 CUBE</p> <p>2 DIMENSION</p> <p>Default restriction is a value of 1.</p>

3.1.4.2.3.12.2 Columns

The MDSCHEMA_KPIS rowset contains the following columns.

Name	Type	Description
CATALOG_NAME	xsd:string	The source database.
SCHEMA_NAME	xsd:string	The name of the schema.
CUBE_NAME	xsd:string	The parent cube for the KPI.
MEASUREGROUP_NAME	xsd:string	The associated measure group for the KPI.
KPI_NAME	xsd:string	The name of the KPI.
KPI_CAPTION	xsd:string	A label or caption associated with the KPI. Used primarily for display purposes. If a caption does not exist, KPI_NAME is returned.
KPI_DESCRIPTION	xsd:string	A human-readable description of the KPI.

Name	Type	Description
KPI_DISPLAY_FOLDER	xsd:string	A string that identifies the path of the display folder that the client application uses to show the member. The folder level separator is defined by the client application. For the tools and clients supplied by SQL Server Analysis Services, the backslash (\) is the level separator. To provide multiple display folders, use a semicolon (;) to separate the folders.
KPI_VALUE	xsd:string	The unique name of the member in the measures dimension for the KPI Value.
KPI_GOAL	xsd:string	The unique name of the member in the measures dimension for the KPI Goal. Returns NULL if there is no Goal defined.
KPI_STATUS	xsd:string	The unique name of the member in the measures dimension for the KPI Status. Returns NULL if there is no Status defined.
KPI_TREND	xsd:string	The unique name of the member in the measures dimension for the KPI Trend. Returns NULL if there is no Trend defined.
KPI_STATUS_GRAPHIC	xsd:string	The default graphical representation of the KPI.
KPI_TREND_GRAPHIC	xsd:string	The default graphical representation of the KPI.

Name	Type	Description
KPI_WEIGHT	xsd:string	The unique name of the member in the measures dimension for the KPI Weight.
KPI_CURRENT_TIME_MEMBER	xsd:string	The unique name of the member in the time dimension that defines the temporal context of the KPI. Returns NULL if there is no Time member defined.
KPI_PARENT_KPI_NAME	xsd:string	The name of the parent KPI.
ANNOTATIONS	xsd:string	(Optional) A set of notes, in XML format.

This schema rowset is not sorted.

3.1.4.2.3.13 MDSCHEMA_MEASUREGROUPS

This Discover element describes the measure groups within a database.

3.1.4.2.3.13.1 Restrictions

The MDSCHEMA_MEASUREGROUPS rowset MAY be restricted on the columns listed in the following table.

Name	Type	Description
CATALOG_NAME	xsd:string	The name of the catalog.
SCHEMA_NAME	xsd:string	The name of the schema.
CUBE_NAME	xsd:string	The name of the cube.
MEASUREGROUP_NAME	xsd:string	The name of the measure group.

3.1.4.2.3.13.2 Columns

The MDSCHEMA_MEASUREGROUPS rowset contains the following columns.

Name	Type	Description
CATALOG_NAME	xsd:string	The name of the catalog to which this measure group belongs. NULL if the server does not support catalogs.
SCHEMA_NAME	xsd:string	The name of the schema.
CUBE_NAME	xsd:string	The name of the cube to which this measure group belongs.
MEASUREGROUP_NAME	xsd:string	The name of the measure group.
DESCRIPTION	xsd:string	A human-readable description of the member.
IS_WRITE_ENABLED	xsd:boolean	A Boolean that indicates whether the measure group is write-enabled. Returns TRUE if the measure group is write enabled.
MEASUREGROUP_CAPTION	xsd:string	The display caption for the measure group.

This schema rowset is not sorted.

3.1.4.2.3.14 MDSCHEMA_MEASUREGROUP_DIMENSIONS

This Discover element enumerates the dimensions of measure groups.

3.1.4.2.3.14.1 Restrictions

The MDSCHEMA_MEASUREGROUP_DIMENSIONS rowset MAY be restricted on the columns listed in the following table.

Name	Type	Description
CATALOG_NAME	xsd:string	The name of the catalog.
SCHEMA_NAME	xsd:string	The name of the schema.
CUBE_NAME	xsd:string	The name of the cube.
MEASUREGROUP_NAME	xsd:string	The name of the measure group.
DIMENSION_UNIQUE_NAME	xsd:string	The unique name of the dimension.
DIMENSION_VISIBILITY	xsd:unsignedShort	<p>A bitmask with one of the following valid values:</p> <ul style="list-style-type: none"> • 1 Visible • 2 Not visible <p>Default restriction is a value of 1.</p>

3.1.4.2.3.14.2 Columns

The MDSHEMA_MEASUREGROUP_DIMENSIONS rowset contains the following columns.

Name	Type	Description
CATALOG_NAME	xsd:string	The name of the catalog to which this measure group belongs. NULL if the server does not support catalogs.
SCHEMA_NAME	xsd:string	The name of the schema.
CUBE_NAME	xsd:string	The name of the cube to which this measure group belongs.

Name	Type	Description
MEASUREGROUP_NAME	xsd:string	The name of the measure group.
MEASUREGROUP_CARDINALITY	xsd:string	<p>The number of instances a measure in the measure group can have for a single dimension member. Possible values include:</p> <ul style="list-style-type: none"> • ONE • MANY
DIMENSION_UNIQUE_NAME	xsd:string	The unique name for the dimension.
DIMENSION_CARDINALITY	xsd:string	<p>The number of instances a dimension member can have for a single instance of a measure group measure.</p> <p>Possible values include:</p> <ul style="list-style-type: none"> • ONE • MANY
DIMENSION_IS_VISIBLE	xsd:boolean	<p>A Boolean that indicates whether hierarchies in the dimension are visible.</p> <p>Returns TRUE if one or more hierarchies in the dimension is visible; otherwise, FALSE.</p>
DIMENSION_IS_FACT_DIMENSION	xsd:boolean	<p>A Boolean that indicates whether the dimension is a fact dimension.</p> <p>Returns TRUE if the dimension is a fact dimension ; otherwise, FALSE.</p>

Name	Type	Description
DIMENSION_PATH	nested rowset	A list of dimensions for the reference dimension.
DIMENSION_GRANULARITY	xsd:string	The unique name of the attribute hierarchy that represents the granularity of the dimension.

The rowset supports sorting on CATALOG_NAME, CUBE_NAME, MEASUREGROUP_NAME, DIMENSION_UNIQUE_NAME.

3.1.4.2.3.15 DISCOVER_PROPERTIES

This Discover element returns a list of information and values about the standard and server-specific properties that are supported by the XMLA server for the specified data source. Unsupported properties are not listed in the returned result set.

If you call the Discover method with the DISCOVER_PROPERTIES enumeration value in the RequestType element, the Discover method returns the DISCOVER_PROPERTIES rowset.

3.1.4.2.3.15.1 Restrictions

The DISCOVER_PROPERTIES rowset can be restricted on the columns listed in the following table.

Name	Type	Description
PropertyName	xsd:string	The name of the property.

3.1.4.2.3.15.2 Columns

The DISCOVER_PROPERTIES rowset contains the following columns.

Name	Type	Description
PropertyName	xsd:string	The name of the property.

Name	Type	Description
PropertyDescription	xsd:string	A localizable text description of the property, or NULL.
PropertyType	xsd:string	The XML data type of the property, or NULL.
PropertyAccessType	xsd:string	The access for the property. The value can be Read, Write, or ReadWrite.
IsRequired	xsd:boolean	A Boolean that indicates whether a property is required. TRUE if a property is required; FALSE if it is not required. Return can be NULL.
Value	xsd:string	The current value of the property, or NULL.

This schema rowset is not sorted.

3.1.4.2.3.16 DISCOVER_LITERALS

This Discover element returns information about literals, including data types and values, supported by the XMLA server.

3.1.4.2.3.16.1 Restrictions

The DISCOVER_LITERALS rowset can be restricted on the columns listed in the following table.

Name	Type	Description
LiteralName	xsd:string	The name of the literal.

3.1.4.2.3.16.2 Columns

125 of 178

The DISCOVER_LITERALS rowset contains the following columns.

Name	Type	Description
LiteralName	xsd:string	<p>The name of the literal described in the row.</p> <p>For example: DBLITERAL_LIKE_PERCENT</p>
LiteralValue	xsd:string	<p>An actual literal value.</p> <p>For example, if LiteralName is DBLITERAL_LIKE_PERCENT and the percent character (%) matches zero or more characters in a LIKE clause, the value of the LiteralValue column is "%".</p>
LiteralInvalidChars	xsd:string	<p>The characters that are not valid in the literal.</p> <p>For example, if table names can contain anything other than a numeric character, this string is "0123456789".</p>
LiteralInvalidStarting Chars	xsd:string	<p>The characters that are not valid as the first character of the literal. If the literal can start with any valid character, this is null.</p>
LiteralMaxLength	xsd:int	<p>The maximum number of characters in the literal. If there is no maximum or the maximum is unknown, the value is -1.</p>

Name	Type	Description
LiteralNameEnumValue	xsd:int	<p>One of the following:</p> <p>DBLITERAL_INVALID = 0</p> <p>DBLITERAL_BINARY_LITERAL = 1</p> <p>DBLITERAL_CATALOG_NAME = 2</p> <p>DBLITERAL_CATALOG_SEPARATOR = 3</p> <p>DBLITERAL_CHAR_LITERAL = 4</p> <p>DBLITERAL_COLUMN_ALIAS = 5</p> <p>DBLITERAL_COLUMN_NAME = 6</p> <p>DBLITERAL_CORRELATION_NAME = 7</p> <p>DBLITERAL_CURSOR_NAME = 8</p> <p>DBLITERAL_ESCAPE_PERCENT = 9</p> <p>DBLITERAL_ESCAPE_UNDERSCORE = 10</p> <p>DBLITERAL_INDEX_NAME = 11</p> <p>DBLITERAL_LIKE_PERCENT = 12</p> <p>DBLITERAL_LIKE_UNDERSCORE = 13</p> <p>DBLITERAL_PROCEDURE_NAME = 14</p>

Name	Type	Description
		DBLITERAL_QUOTE_PREFIX = 15 DBLITERAL_SCHEMA_NAME = 16 DBLITERAL_TABLE_NAME = 17 DBLITERAL_TEXT_COMMAND = 18 DBLITERAL_USER_NAME = 19 DBLITERAL_VIEW_NAME = 20 DBLITERAL_CUBE_NAME = 21 DBLITERAL_DIMENSION_NAME = 22 DBLITERAL_HIERARCHY_NAME = 23 DBLITERAL_LEVEL_NAME = 24 DBLITERAL_MEMBER_NAME = 25 DBLITERAL_PROPERTY_NAME = 26 DBLITERAL_SCHEMA_SEPARATOR = 27 DBLITERAL_QUOTE_SUFFIX = 28 DBLITERAL_ESCAPE_PERCENT_SUFFIX = 29 DBLITERAL_ESCAPE_UNDERSCORE_SUFFIX = 30

This schema rowset is not sorted.

3.1.4.2.3.17 DISCOVER_SCHEMA_ROWSETS

This Discover element returns the names, restrictions, description, and other information for all enumeration values and any additional server-specific enumeration values supported by the XMLA server.

If you call the Discover method with the DISCOVER_SCHEMA_ROWSETS enumeration value in the RequestType element, the Discover method returns the DISCOVER_SCHEMA_ROWSETS rowset.

3.1.4.2.3.17.1 Restrictions

The DISCOVER_SCHEMA_ROWSETS rowset can be restricted on the columns listed in the following table.

Name	Type	Description
SchemaName	xsd:string	The name of the schema.

3.1.4.2.3.17.2 Columns

The DISCOVER_SCHEMA_ROWSETS rowset contains the following columns.

Name	Type	Description
SchemaName	xsd:string	The name of the schema or request. This request returns the values in the RequestTypes enumeration.
SchemaGuid	uuid	The GUID of the schema.
Restrictions	nested rowset	An array of the restrictions supported by the server.
Description	xsd:string	A localizable description of the schema.
RestrictionsMask	xsd:unsignedLong	The lowest N bits set to 1, where N is the number of restrictions.

This schema rowset is not sorted.

3.1.4.2.3.18 DISCOVER_KEYWORDS

This Discover element returns information about keywords reserved by the XMLA server.

If you call the Discover method with the DISCOVER_KEYWORDS enumeration value in the RequestType element, the Discover method returns the DISCOVER_KEYWORDS rowset.

3.1.4.2.3.18.1 Restrictions

The DISCOVER_KEYWORDS rowset MAY be restricted on the columns listed in the following table.

Name	Type	Description
Keyword	xsd:string	A list of keywords that MAY be restricted for use by the server.

3.1.4.2.3.18.2 Columns

The DISCOVER_KEYWORDS rowset contains the following columns.

Name	Type	Description
Keyword	xsd:string	A list of all the keywords reserved by a server. Example: AND

This schema rowset is not sorted.

3.1.4.3 Execute

This operation is used for sending commands to the server. The command can retrieve data from the server or update data on the server.

```
<wsdl:operation name="Execute">  
  <wsdl:input message=" ExecuteSoapIn" />  
  <wsdl:output message=" ExecuteSoapOut" />  
</wsdl:operation>
```

The protocol client sends an ExecuteSoapIn request message and the protocol server responds with an ExecuteSoapOut response message.

130 of 178

3.1.4.3.1 Messages

The following [WSDL message](#) definitions are specific to this operation.

3.1.4.3.1.1 ExecuteSoapIn

This message is the request message for Execute.

The SOAP Action value of the message is defined as:

```
"urn:schemas-microsoft-com:xml-analysis:Execute"
```

The SOAP Body contains an Execute element.

```
<message name="ExecuteSoapIn">
  <part name="parameters" element="xmla:Execute" />
</message>
```

3.1.4.3.1.2 ExecuteSoapOut

This message is the response message for Execute.

The SOAP Action value of the message is defined as:

```
"urn:schemas-microsoft-com:xml-analysis:Execute"
```

The SOAP Body contains an ExecuteResponse element.

```
<message name="ExecuteSoapOut">
  <part name="parameters" element="xmla:ExecuteResponse" />
</message>
```

3.1.4.3.2 Elements

The following XML Schema element definitions are specific to this operation.

3.1.4.3.2.1 Execute

The Execute element has the following definition:

```
<xs:element name="Execute">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Command" minOccurs="1" maxOccurs="1" nillable="true">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="Statement" type="xs:string" minOccurs="1" maxOccurs="1"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="Properties" minOccurs="1" maxOccurs="1" nillable="true">
```

```

<xs:complexType>
  <xs:sequence>
    <xs:element name="PropertyList" minOccurs="0" maxOccurs="1" nillable="true">
      <xs:complexType>
        <xs:any />
      </xs:complexType>
    </xs:element>
  </xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>

```

The `xm1a:Execute` element describes the structure of the `Execute` command. The `Execute` command is used to execute MDX or other server-specific commands against an OLAP server.

The `xm1a:Execute` element contains the following sub-elements:

- **Command:** Contains a `Statement` sub-element with the MDX expression, or other server-specific command, to be executed.
- **Properties:** A set of properties and values that controls the execution of the `Command` sub-element, such as defining the information required to connect to the data source, specifying the return format of the result set, or specifying the locale in which the data should be formatted.

3.1.4.3.2.1.1 Command sub-element

The `Command` sub-element contains the `Statement` sub-element that contains the MDX expression, or other server-specific command, to be executed.

This sub-element **MUST** be included. There **MUST** be only one `Statement` sub-element per `Command` sub-element. The `Statement` sub-element **MAY** be an empty sub-element. The server **MUST** support the `Statement` sub-element being empty and respond with an [xm1a-root complex type](#). The value of the `Statement` sub-element is a provider-specific command text; typically an MDX statement or a SQL statement.

The following example shows how to send an MDX statement for execution at the server.

```

<Execute xmlns="urn:schemas-microsoft-com:xml-analysis">
  <Command>
    <Statement>
      with member [Measures].[Avg Sales Value per Item count]
      as '[Internet Sales Amount]/([Internet Order Quantity])'
      select [Measures].[Avg Sales Value per Item count] on 0,

```

```

        [Calendar].[Month].members on 1
    from [Adventure Works]
</Statement>
</Command>
<Properties>
    <PropertyList>
        <Catalog>Adventure Works DW</Catalog>
    </PropertyList>
</Properties>
</Execute>

```

3.1.4.3.2.1.2 Properties sub-element

This sub-element consists of a collection of request properties. Each property allows the user to control some aspect of the Execute method, such as defining the information required for the connection, specifying the return format of the result set, or specifying the locale in which the data should be formatted. The available properties and their values can be obtained by using the DISCOVER_PROPERTIES request type with the Discover method. There is no required order for the properties listed in the Properties sub-element. This sub-element **MUST** be included, but it **MAY** be empty.

The Properties sub-element **MUST** contain one and only one element PropertyList. The PropertyList sub-element is an open structure element that consists of zero or more elements with a simple data type value. The name of the element represents the name of the property being set and the value represents the value assigned to the property. The PropertyList sub-element **MAY** be an empty element.

The following example shows how to set the Catalog property to "Adventure Works DW".

```

<Execute xmlns="urn:schemas-microsoft-com:xml-analysis">
    <Command>
        <Statement>
            with member [Measures].[Avg Sales Value per Item count]
            as '[Internet Sales Amount]/([Internet Order Quantity)]'
            select [Measures].[Avg Sales Value per Item count] on 0,
                [Calendar].[Month].members on 1
            from [Adventure Works]
        </Statement>
    </Command>
    <Properties>
        <PropertyList>
            <Catalog>Adventure Works DW</Catalog>
        </PropertyList>
    </Properties>
</Execute>

```

3.1.4.3.2.2 ExecuteResponse

The ExecuteResponse element has the following definition:

133 of 178

```

<xs:element name="ExecuteResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="1" name="return">
        <xs:complexType>
          <xs:element name="root" minOccurs="1" maxOccurs="1">
            </xs:element>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

```

The `xmldb:ExecuteResponse` element describes the structure of response received from the Execute command.

The `ExecuteResponse` element **MUST** contain zero or one `Return` sub-elements.

3.1.4.3.2.2.1 Return sub-element

The `return` sub-element contains the result returned by the server. The `Command` sub-element and values in the `Properties` sub-element define the shape of the result set.

3.1.4.3.2.2.2 Root sub-element

The `Root` sub-element contains the response to an `Execute` command. The structure of the `Root` element is associated with the namespace defined for the element. The namespace for the `Root` sub-element **MUST** be any out of the following list:

- `urn:schemas-microsoft-com:xml-analysis:mddataset`
- `urn:schemas-microsoft-com:xml-analysis:rowset`
- `urn:schemas-microsoft-com:xml-analysis:empty`

For more information about the different types of `Root` responses, see sections [xmldb-ds.root complex type](#), [xmldb-e.root complex type](#), and [xmldb-rs.root complex type](#).

3.1.4.4 Request Properties

The following table lists all properties that the server **MUST** support. All of these properties **MUST** be included in the response to the `DISCOVER_PROPERTIES` Discover request. A client **MAY** specify one or more read-write properties in a Discover or Execute request.

Name	Type	Usage	Allowed values	Description
------	------	-------	----------------	-------------

Name	Type	Usage	Allowed values	Description
Catalog	xsd:string	R/W	Any valid catalog name.	<p>This property specifies the catalog for the request.</p> <p>The default value is an empty string and means the server can use any catalog for the request.</p>
DBMSVersion	xsd:string	R/O	A string representation of the server version, typically numbers separated by periods, e.g. "10.1.3.345"	This property specifies the version of the server.

Name	Type	Usage	Allowed values	Description
DbpropMsmdMDXCcompatibility	xsd:int	R/W	<p>0 - This value is equivalent to 1.</p> <p>1 – Placeholder members are exposed in Discover and Execute requests. Hierarchies in role-playing dimensions receive a caption that includes the dimension name and the hierarchy name: "dimension.hierarchy"</p> <p>2 - Placeholder members are not exposed in Discover and Execute requests. Hierarchies in role-playing dimensions receive a caption that includes the dimension name and the hierarchy name: "dimension.hierarchy"</p> <p>3 - Placeholder members are not exposed. Hierarchies in role-playing dimensions receive a caption that excludes the dimension name.</p>	<p>This property specifies how placeholder members in ragged hierarchies are exposed in Discover and Execute requests. It also specifies how captions for role-playing dimensions are exposed.</p> <p>The default value is 3.</p>

Name	Type	Usage	Allowed values	Description
LocaleIdentifier	xsd:int	R/W	For the complete hexadecimal list of language identifiers, see [MS-LCID] .	This property specifies the locale identifier for the request. There is no default value – it is specific to the server implementation. < 35 >
MdpropFlatteningSupport	xsd:int	R/O	1 - Full support for flattening 2 - Support for flattening using dummy column names 3 - Support for flattening by generating one column per dimension 4 – No support for flattening	This property specifies the server support for flattening multidimensional datasets into tabular rowsets. < 36 > For more information on flattening, see [MSDN-MDPROP]
MdpropMdxDrillFunctions	xsd:int	R/O	1 – Support for drilling on sets based on a single hierarchy 2 – Support for drilling on sets based on multiple hierarchies	This property specifies a bitmask indicating server support for MDX functions to drill up and down on members and tuples. < 37 >

Name	Type	Usage	Allowed values	Description
MdpropMdxFormulas	xsd:int	R/O	<p>1 – Support for creation of calculated members using the WITH clause</p> <p>2 - Support for creation of named sets using the WITH clause</p> <p>4 – Support for creation of calculated members using the CREATE statement</p> <p>8 – Support for creation of named sets using the CREATE statement</p> <p>16 – Support for the scope value of SESSION in the creation of calculated members and named sets</p> <p>32 - Support for the scope value of GLOBAL in the creation of calculated members and named sets</p>	This property specifies a bitmask indicating server support for calculated members and named sets. < 38 >

Name	Type	Usage	Allowed values	Description
MdpropMdxSetFunctions	xsd:int	R/O	1 – TopPercent 2 - BottomPercent 4 - TopSum 8 – BottomSum 16 - PeriodsToDate 32 - LastPeriods 64 - YTD 128 – QTD 256 – MTD 512 – WTD 1024 – DrillDownMember 2048 – DrillDownLevel 4096 - DrillDownMemberTop 8192 - DrillDownMemberBottom 16384 – DrillDownLevelTop 32768 – DrillDownLevelBottom 65536 – DrillUpMember 131072 – DrillUpLevel 262144 - ToggleDrillState	This property specifies a bitmask indicating server support for MDX set functions.< 39 >

Name	Type	Usage	Allowed values	Description
MdpropMdxSubqueries	xsd:int	R/O	0 – No support for subqueries 1 – Basic support for subqueries 3 – Full support for subqueries	This property specifies server support for subqueries in MDX.< 40 >
MdpropNamedLevels	xsd:int	R/O	1 – Support for named levels 2 – Support for numbered levels – using the Levels(n) MDX function	This property specifies a bitmask indicating server support for named levels in hierarchies.< 41 >
MdxMissingMemberMode	xsd:string	R/W	"Default" - The default behavior is specific to the server implementation. "Error" - Return an error when a missing member is encountered. "Ignore" - Ignore missing members.	This property specifies how the server should handle missing members in MDX statements. The default value for this property is "Default".
ProviderType	xsd:int	R/O	1 – Tabular data provider 2 – Multidimensional data provider 3 – Tabular and multidimensional data provider	This property specifies the type of the server.< 42 >

Name	Type	Usage	Allowed values	Description
ProviderVersion	xsd:string	R/O	A string representation of the server version, typically numbers separated by periods, for example, "03.50.6789".	This property specifies the version of the server.
SafetyOptions	xsd:int	R/W	<p>0 – This value is equivalent to 2.</p> <p>1 – Allow safe as well as unsafe operations (assuming high trust level for the client).</p> <p>2 – Allow only safe operations.</p> <p>3 – Disallow all vulnerable operations (safe and unsafe). It is up to the server implementation to decide which features are vulnerable.</p>	<p>This property specifies the level of safety that the server should observe for the request.</p> <p>The default value is 0.</p>
VisualMode	xsd:int	R/W	<p>0 – The default behavior is specific to the server implementation.</p> <p>1 – Return visual totals</p> <p>2 – Return full totals</p>	<p>This property specifies whether the server should return visual totals for the MDX statement.</p> <p>The default value is 0.</p>

3.1.5 Timer Events

None.

3.1.6 Other Local Events

3.1.6.1 Error Handling

Different kinds of errors are handled in different ways. The following errors can occur:

- Request failed
- Request succeeded, but with errors or warnings
- Request succeeded, but with cell errors

3.1.6.1.1 *Request failed*

Failure to execute a request is reported through a SOAP Fault message. When this occurs, Result is not returned.

The [SOAP Fault codes](#) relating to this specification begin with "XMLAnalysisError" followed by a period and the hexadecimal HR result code. For example, an error code of "0x80000005" would be formatted as "XMLAnalysisError.0x80000005".

The following is an example of a SOAP Fault for a failed method call:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <soap:Fault xmlns="http://schemas.xmlsoap.org/soap/envelope/">
      <faultcode>XMLAnalysisError.0xc10a0004</faultcode>
      <faultstring>The Adventure Works cube either does not exist or has not been
processed.</faultstring>
      <detail>
        <Error
          ErrorCode="3238658052"
          Description="The Adventure Works cube either does not exist or has not been
processed."
          Source="Server"
          HelpFile="" />
      </detail>
    </soap:Fault>
  </soap:Body>
</soap:Envelope>
```

3.1.6.1.2 *Request succeeded, but with errors or warnings*

When the method completes with errors or warnings, these are returned to the client with *Result*.

For more information, see [Exception sub-element 2.2.4.1.5](#), [xmla-e:root complex type](#), and [Exception sub-element 2.2.4.3.3](#).

If the server encounters an error after it begins serializing the output, it SHOULD output an `<Exception>` element at the point of the error, as follows:

```
<Exception xmlns=urn:schemas-microsoft-com:xml-analysis:exception>
```

The server should then close all open elements so that the XML document received by the client is valid. Finally, the server should output the Messages section containing the description of the error followed by the closing `</root>` element.

3.1.6.1.3 Request succeeded, but with cell errors

Errors specific to cells or data in *Result* are embedded within the result set.

3.2 Transport Specific Protocol Details

3.2.1 Connection

To connect to a server using TCP or HTTP/HTTPS, the client MUST know the IP address and port number of the server. The default HTTP port number is 80 and the default HTTPS port number is 443. However the server MAY be configured to listen on other port numbers. [<43>](#)

3.2.2 Authentication and Encryption

To use an authenticated or encrypted connection using TCP, both the client and server MUST use GSS-API [RFC4178]. This requires the exchange of [security tokens](#) between the client and server. The client sends its UTF-8 encoded security token using the Authenticate request ([AuthenticateSoapIn](#)) and the server responds with its UTF-8 encoded security token in the AuthenticateResponse message ([AuthenticateSoapOut](#)). This exchange of security tokens MUST continue back and forth until GSS-API reports completion or error. After it has completed, the client and server can call the GSS-API to determine if encryption or hashing is turned on for the connection.

When using HTTP/HTTPS, the server MAY or MAY NOT support authenticated connections. If the client requires messages to be encrypted, it SHOULD use the HTTPS protocol.

3.2.3 Content Type Negotiation

Because the support for binary XML [\[MS-BINXML\]](#) and compression is optional, the client and server MUST negotiate the content type of the messages for the duration of the connection. The negotiation is based on the preferences and capabilities of the client and server.

The first request sent by the client and the server response are both always text XML. Depending on whether the transport is TCP or HTTP/HTTPS, the DIME OPTIONS field or the HTTP X-Transport-Caps-Negotiation-Flags header is used by client and server to indicate the content type of the messages for the connection.

The process of negotiation is controlled by the client and the server does not need to remember the current state of negotiation between requests. The `NEGO` flag is used by the client to inform the server that negotiation is in progress. The client **SHOULD** set this flag to 1 in the first request and set it to 0 in all subsequent requests (since the negotiation is complete).

The `REQ_SX`, `REQ_XPRESS`, `RESP_SX` and `RESP_XPRESS` flags are used for the negotiation to represent binary XML in requests, compression in requests, binary XML in responses and compression in responses. The value 0 indicates that the capability is off or not supported. The value 1 indicates that the capability is on or supported.

The client uses the `RESP_SX` and `RESP_XPRESS` flags to inform the server whether it supports responses with binary XML and compression.

The server uses the `RESP_SX` and `RESP_XPRESS` flags to inform the client whether it will use binary XML and compression in its responses. This decision is based on what the client supports as well as the server capabilities.

Because the server does not remember the current state of negotiation, the client and server **MUST** send these flags in every request and response.

3.2.4 Generating and Parsing Messages

After the connection has been established and authentication and content type negotiation has been completed, the client and server know whether encryption, compression and binary XML are enabled for requests and responses.

To generate a message, the following steps **MUST** be followed:

TCP:

1. Generate the SOAP envelope.
2. If binary XML is enabled, then encode the SOAP message as described in 2.3.5 and [\[MS-BINXML\]](#). Otherwise encode the SOAP message in text XML.
3. If compression is enabled, then compress the message as described in [Compression](#). The message **MAY** optionally be divided into multiple compression data blocks.
4. If encryption is enabled, then encrypt the message using GSS-API. The message **MAY** optionally be divided into multiple encryption data blocks.
5. Compose the message into DIME records and send it via TCP. The message **MAY** optionally be divided into multiple DIME records.

HTTP/HTTPS:

1. Generate the SOAP envelope.
2. If binary XML is enabled, then encode the SOAP message as described in 2.3.5 and [\[MS-BINXML\]](#). Otherwise encode the SOAP message in text XML.
3. If compression is enabled, then compress the message as described in [Compression](#). The message MAY optionally be divided into multiple compression data blocks.
4. Send the message via HTTP/HTTPS along with the appropriate HTTP headers.

To parse a message, the following steps MUST be followed:

TCP:

1. Combine all the DIME records into a single data block.
2. If encryption is enabled, decrypt all the encryption data blocks and combine them into a single decrypted data block.
3. If compression is enabled, decompress all the compression data blocks as described in [Compression](#) and combine them into a single decompressed data block.
4. If binary XML is enabled, decode the data block as described in 2.3.5 and [\[MS-BINXML\]](#). Otherwise decode the data block as text XML.
5. Parse the SOAP envelope.

HTTP/HTTPS:

1. If compression is enabled, decompress all the compression data blocks as described in [Compression](#) and combine them into a single decompressed data block.
2. If binary XML is enabled, decode the data block as described in 2.3.5 and [\[MS-BINXML\]](#). Otherwise decode the data block as text XML.
3. Parse the SOAP envelope.

3.2.5 Compression

The client or server can choose any compression algorithm as long as the compressed data blocks can be decompressed using the following decompression algorithm.

The decompression algorithm takes a buffer of compressed data in the form of a byte array (InputBuffer), an output buffer in the form of a byte array (OutputBuffer), and the size of the output buffer (OutputBufferSize).

```

SET KindBit to 0
SET HaveNibble to FALSE
SET OutputBufferIndex to 0
SET InputBufferIndex to 0
WHILE OutputBufferIndex < OutputBufferSize
    IF KindBit is 0 THEN
        SET Kind to InputBuffer[InputBufferIndex] +
            (InputBuffer[InputBufferIndex+1] << 8) +
            (InputBuffer[InputBufferIndex+2] << 16) +
            (InputBuffer[InputBufferIndex+3] << 24)
        INCREMENT InputBufferIndex by 4
        SET KindBit to 32
    ENDIF
    DECREMENT KindBit
    IF (Kind & (1 << KindBit)) is 0 THEN
        SET OutputBuffer[OutputBufferIndex] to InputBuffer[InputBufferIndex]
        INCREMENT InputBufferIndex
        INCREMENT OutputBufferIndex
    ELSE
        SET Length to InputBuffer[InputBufferIndex] +
            (InputBuffer[InputBufferIndex+1] << 8)
        INCREMENT InputBufferIndex by 2
        SET Offset to (Length >> 3)
        SET Length to (Length & 7)
        IF Length is 7 THEN
            IF HaveNibble is FALSE THEN
                SET HaveNibble to TRUE
                Set NibbleValue to InputBuffer[InputBufferIndex]
                SET Length to (InputBuffer[InputBufferIndex] & 15)
                INCREMENT InputBufferIndex
            ELSE
                SET Length to (NibbleValue >> 4)
                SET HaveNibble to FALSE
            ENDIF
        ENDIF
        IF Length is 15 THEN
            SET Length to InputBuffer[InputBufferIndex]
            INCREMENT InputBufferIndex
            IF Length is 255 THEN
                SET Length to InputBuffer[InputBufferIndex] +
                    (InputBuffer[InputBufferIndex+1] << 8)
                INCREMENT InputBufferIndex by 2
                DECREMENT Length by 22
            ENDIF
            INCREMENT Length by 15
        ENDIF
        INCREMENT Length by 7
    ENDIF
    INCREMENT Length by 3
    WHILE Length is not 0
        SET OutputBuffer[OutputBufferIndex] to

```

```

        OutputBuffer[OutputBufferIndex-Offset-1]
    INCREMENT OutputBufferIndex
    DECREMENT Length
ENDWHILE
ENDIF
ENDWHILE

```

4 Protocol Examples

4.1 Client obtains a list of databases from the server over TCP

In this example, the client connects to the server using TCP and obtains server information by sending a DBSCHEMA_CATALOGS request.

4.1.1 Connection

The server listens on a TCP port for incoming requests from clients. The client creates a new TCP connection to the server.

4.1.2 Authentication

Once the connection is established, the client sends an authentication request to the server:

0E 10 00 04 00 00 00 08 00 00 01 3C 00 00 00 00<....
74 65 78 74 2F 78 6D 6C EF BB BF 3C 45 6E 76 65	text/xml???<Enve
6C 6F 70 65 20 78 6D 6C 6E 73 3D 22 68 74 74 70	lope xmlns="http
3A 2F 2F 73 63 68 65 6D 61 73 2E 78 6D 6C 73 6F	://schemas.xmlso
61 70 2E 6F 72 67 2F 73 6F 61 70 2F 65 6E 76 65	ap.org/soap/enve
6C 6F 70 65 2F 22 3E 0D 0A 20 20 3C 42 6F 64 79	lope/">.. <Body
3E 0D 0A 20 20 20 20 3C 41 75 74 68 65 6E 74 69	>.. <Authenti
63 61 74 65 20 78 6D 6C 6E 73 3D 22 68 74 74 70	cate xmlns="http
3A 2F 2F 73 63 68 65 6D 61 73 2E 6D 69 63 72 6F	://schemas.micro
73 6F 66 74 2E 63 6F 6D 2F 61 6E 61 6C 79 73 69	soft.com/analysi
73 73 65 72 76 69 63 65 73 2F 32 30 30 33 2F 65	sservices/2003/e
78 74 22 3E 0D 0A 20 20 20 20 20 20 3C 53 73 70	xt">.. <Ssp
69 48 61 6E 64 73 68 61 6B 65 3E 54 6C 52 4D 54	iHandshake>TlRMT
56 4E 54 55 41 41 42 41 41 41 41 42 37 49 49 6F	VNTUAABAAAAB7IIo

147 of 178

67	63	41	42	77	41	78	41	41	41	41	43	51	41	4A	41	gcABwAxAAAAACQAJA
43	67	41	41	41	41	46	41	73	34	4F	41	41	41	41	44	CgAAAAFAs4OAAAD
31	5A	42	54	45	56	53	53	55	73	77	4D	31	4A	46	52	1ZBTEVSSUswM1JFR
45	31	50	54	6B	51	3D	3C	2F	53	73	70	69	48	61	6E	E1PTkQ=</SspiHan
64	73	68	61	6B	65	3E	0D	0A	20	20	20	20	3C	2F	41	dshake>.. </A
75	74	68	65	6E	74	69	63	61	74	65	3E	0D	0A	20	20	uthenticate>..
3C	2F	42	6F	64	79	3E	0D	0A	3C	2F	45	6E	76	65	6C	</Body>..</Envel
6F	70	65	3E	ope>

Highlighted is the DIME header:

- *VERSION: 1*
- *MB: 1*
- *ME: 1*
- *CF: 0*
- *TYPE_T: 1*
- *RESERVED: 0*
- *OPTIONS_LENGTH: 4*
- *ID_LENGTH: 0*
- *TYPE_LENGTH: 8*
- *DATA_LENGTH: 316*
- *OPTIONS:*
- *NEGO: 0*
- *REQ_SX: 0*
- *REQ_XPRESS: 0*

- *RESP_SX*: 0
- *RESP_XPRESS*: 0
- *RESERVED*: 0
- *TYPE*: *text/xml*

The server responds with an authentication [handshake](#) response:

0E 10 00 04 00 00 00 08 00 00 02 95 00 00 00 00?....
74 65 78 74 2F 78 6D 6C 3C 73 6F 61 70 3A 45 6E	text/xml<soap:En
76 65 6C 6F 70 65 20 78 6D 6C 6E 73 3A 73 6F 61	velope xmlns:soa
70 3D 22 68 74 74 70 3A 2F 2F 73 63 68 65 6D 61	p="http://schema
73 2E 78 6D 6C 73 6F 61 70 2E 6F 72 67 2F 73 6F	s.xmlsoap.org/so
61 70 2F 65 6E 76 65 6C 6F 70 65 2F 22 3E 3C 73	ap/envelope/"><s
6F 61 70 3A 42 6F 64 79 3E 3C 41 75 74 68 65 6E	oap:Body><Authen
74 69 63 61 74 65 52 65 73 70 6F 6E 73 65 20 78	ticateResponse x
6D 6C 6E 73 3D 22 68 74 74 70 3A 2F 2F 73 63 68	mlns="http://sch
65 6D 61 73 2E 6D 69 63 72 6F 73 6F 66 74 2E 63	emas.microsoft.c
6F 6D 2F 61 6E 61 6C 79 73 69 73 73 65 72 76 69	om/analysis servi
63 65 73 2F 32 30 30 33 2F 65 78 74 22 3E 3C 72	ces/2003/ext"><r
65 74 75 72 6E 3E 3C 53 73 70 69 48 61 6E 64 73	eturn><SspiHands
68 61 6B 65 3E 54 6C 52 4D 54 56 4E 54 55 41 41	hake>TlRMTVNTUAA
43 41 41 41 41 44 67 41 4F 41 44 67 41 41 41 41	CAAAADgAOADgAAAA
46 77 6F 6D 69 30 34 54 6F 55 79 59 54 71 38 70	Fwomi04ToUyYTq8p
67 71 4A 6F 41 41 41 41 41 41 4E 67 41 32 41 42	gqJoAAAAAANGA2AB
47 41 41 41 41 42 51 4C 4F 44 67 41 41 41 41 39	GAAAABQLODgAAAA9
53 20 0A 41 45 55 41 52 41 42 4E 41 45 38 41 54	S .AEUARABNAE8AT
67 42 45 41 41 49 41 44 67 42 53 41 45 55 41 52	gBEAAIADgBSAEUAR

41 42 4E 41	45 38 41 54	67 42 45 41	41 45 41 45	ABNAE8ATgBEAAEAE
67 42 57 41	45 45 41 54	41 42 46 41	46 49 41 53	gBWAEETABFAFIAS
51 42 4C 41	44 41 41 4D	77 41 45 41	44 51 41 20	QBLADAAMwAEADQA
0A 63 67 42	6C 41 47 51	41 62 51 42	76 41 47 34	.cgBlAGQAbQBvAG4
41 5A 41 41	75 41 47 4D	41 62 77 42	79 41 48 41	AZAAuAGMAbwByAHA
41 4C 67 42	74 41 47 6B	41 59 77 42	79 41 47 38	ALgBtAGkAYwByAG8
41 63 77 42	76 41 47 59	41 64 41 41	75 41 47 4D	AcwBvAGYAdAAuAGM
41 62 77 42	74 41 41 4D	41 53 41 42	32 20 0A 41	AbwBtAAMASAB2 .A
47 45 41 62	41 42 6C 41	48 49 41 61	51 42 72 41	GEAbABlAHIAaQBrA
44 41 41 4D	77 41 75 41	48 49 41 5A	51 42 6B 41	DAAMwAuAHIAZQBkA
47 30 41 62	77 42 75 41	47 51 41 4C	67 42 6A 41	G0AbwBuAGQALgBjA
47 38 41 63	67 42 77 41	43 34 41 62	51 42 70 41	G8AcgBwAC4AbQBpA
47 4D 41 63	67 42 76 41	48 4D 41 20	0A 62 77 42	GMACgBvAHMA .bwB
6D 41 48 51	41 4C 67 42	6A 41 47 38	41 62 51 41	mAHQALgBjAG8AbQA
46 41 43 51	41 59 77 42	76 41 48 49	41 63 41 41	FACQAYwBvAHIAcAA
75 41 47 30	41 61 51 42	6A 41 48 49	41 62 77 42	uAG0AaQBjAHIAbwB
7A 41 47 38	41 5A 67 42	30 41 43 34	41 59 77 42	zAG8AZgB0AC4AYwB
76 41 47 30	41 41 41 41	41 20 0A 41	41 3D 3D 20	vAG0AAAAA .AA==
0A 3C 2F 53	73 70 69 48	61 6E 64 73	68 61 6B 65	.</SspiHandshake
3E 3C 2F 72	65 74 75 72	6E 3E 3C 2F	41 75 74 68	></return></Auth
65 6E 74 69	63 61 74 65	52 65 73 70	6F 6E 73 65	enticateResponse
3E 3C 2F 73	6F 61 70 3A	42 6F 64 79	3E 3C 2F 73	></soap:Body></s
6F 61 70 3A	45 6E 76 65	6C 6F 70 65	3E CC CC CC	oap:Envelope>???

Highlighted is the DIME header and the padding required for the 4 byte alignment:

- *VERSION: 1*

- *MB*: 1
- *ME*: 1
- *CF*: 0
- *TYPE_T*: 1
- *RESERVED*: 0
- *OPTIONS_LENGTH*: 4
- *ID_LENGTH*: 0
- *TYPE_LENGTH*: 8
- *DATA_LENGTH*: 661
- *OPTIONS*:
 - *NEGO*: 0
 - *REQ_SX*: 0
 - *REQ_XPRESS*: 0
 - *RESP_SX*: 0
 - *RESP_XPRESS*: 0
 - *RESERVED*: 0
- *TYPE*: *text/xml*

The client continues the authentication handshake:

0E 10 00 04 00 00 00 08 00 00 01 50 01 00 00 00P....
74 65 78 74 2F 78 6D 6C EF BB BF 3C 45 6E 76 65	text/xml???<Enve
6C 6F 70 65 20 78 6D 6C 6E 73 3D 22 68 74 74 70	lope xmlns="http
3A 2F 2F 73 63 68 65 6D 61 73 2E 78 6D 6C 73 6F	://schemas.xmlso
61 70 2E 6F 72 67 2F 73 6F 61 70 2F 65 6E 76 65	ap.org/soap/enve
6C 6F 70 65 2F 22 3E 0D 0A 20 20 3C 42 6F 64 79	lope/">.. <Body

151 of 178

3E 0D 0A 20 20 20 20 3C 41 75 74 68 65 6E 74 69	>.. <Authenti
63 61 74 65 20 78 6D 6C 6E 73 3D 22 68 74 74 70	cate xmlns="http
3A 2F 2F 73 63 68 65 6D 61 73 2E 6D 69 63 72 6F	://schemas.micro
73 6F 66 74 2E 63 6F 6D 2F 61 6E 61 6C 79 73 69	soft.com/analysi
73 73 65 72 76 69 63 65 73 2F 32 30 30 33 2F 65	sservices/2003/e
78 74 22 3E 0D 0A 20 20 20 20 20 20 3C 53 73 70	xt">.. <Ssp
69 48 61 6E 64 73 68 61 6B 65 3E 54 6C 52 4D 54	iHandshake>TlRMT
56 4E 54 55 41 41 44 41 41 41 41 41 41 41 41	VNTUAADAAAAAAAAA
45 67 41 41 41 41 41 41 41 41 53 41 41 41 41	EgAAAAAAAAASAAAA
41 41 41 41 41 42 49 41 41 41 41 41 41 41 41	AAAAABIAAAAAAAAAA
45 67 41 41 41 41 41 41 41 41 53 41 41 41 41	EgAAAAAAAAASAAAA
41 41 41 41 41 42 49 41 41 41 41 42 63 4B 49 6F	AAAAABIAAAABcKIo
67 55 43 7A 67 34 41 41 41 41 50 3C 2F 53 73 70	gUCzg4AAAAAP</Ssp
69 48 61 6E 64 73 68 61 6B 65 3E 0D 0A 20 20 20	iHandshake>..
20 3C 2F 41 75 74 68 65 6E 74 69 63 61 74 65 3E	</Authenticate>
0D 0A 20 20 3C 2F 42 6F 64 79 3E 0D 0A 3C 2F 45	.. </Body>..</E
6E 76 65 6C 6F 70 65 3E	nvelope>

Highlighted is the DIME header:

- *VERSION: 1*
- *MB: 1*
- *ME: 1*
- *CF: 0*
- *TYPE_T: 1*
- *RESERVED: 0*

- *OPTIONS_LENGTH*: 4
- *ID_LENGTH*: 0
- *TYPE_LENGTH*: 8
- *DATA_LENGTH*: 336
- *OPTIONS*:
- *NEGO*: 1
- *REQ_SX*: 0
- *REQ_XPRESS*: 0
- *RESP_SX*: 0
- *RESP_XPRESS*: 0
- *RESERVED*: 0
- *TYPE*: *text/xml*

The server responds with the authentication handshake completion:

0E 10 00 04 00 00 00 08 00 00 00 FA 00 00 00 00?....
74 65 78 74 2F 78 6D 6C 3C 73 6F 61 70 3A 45 6E	text/xml<soap:En
76 65 6C 6F 70 65 20 78 6D 6C 6E 73 3A 73 6F 61	velope xmlns:soa
70 3D 22 68 74 74 70 3A 2F 2F 73 63 68 65 6D 61	p="http://schema
73 2E 78 6D 6C 73 6F 61 70 2E 6F 72 67 2F 73 6F	s.xmlsoap.org/so
61 70 2F 65 6E 76 65 6C 6F 70 65 2F 22 3E 3C 73	ap/envelope/"><s
6F 61 70 3A 42 6F 64 79 3E 3C 41 75 74 68 65 6E	oap:Body><Authen
74 69 63 61 74 65 52 65 73 70 6F 6E 73 65 20 78	ticateResponse x
6D 6C 6E 73 3D 22 68 74 74 70 3A 2F 2F 73 63 68	mlns="http://sch
65 6D 61 73 2E 6D 69 63 72 6F 73 6F 66 74 2E 63	emas.microsoft.c
6F 6D 2F 61 6E 61 6C 79 73 69 73 73 65 72 76 69	om/analysis servi

63	65	73	2F	32	30	30	33	2F	65	78	74	22	3E	3C	72	ces/2003/ext"><r
65	74	75	72	6E	3E	3C	53	73	70	69	48	61	6E	64	73	eturn><SspiHands
68	61	6B	65	2F	3E	3C	2F	72	65	74	75	72	6E	3E	3C	hake/></return><
2F	41	75	74	68	65	6E	74	69	63	61	74	65	52	65	73	/AuthenticateRes
70	6F	6E	73	65	3E	3C	2F	73	6F	61	70	3A	42	6F	64	ponse></soap:Body
79	3E	3C	2F	73	6F	61	70	3A	45	6E	76	65	6C	6F	70	y></soap:Envelope
65	3E	CC	CC	e>??

Highlighted is the DIME header and the padding required for the 4 byte alignment:

- *VERSION: 1*
- *MB: 1*
- *ME: 1*
- *CF: 0*
- *TYPE_T: 1*
- *RESERVED: 0*
- *OPTIONS_LENGTH: 4*
- *ID_LENGTH: 0*
- *TYPE_LENGTH: 8*
- *DATA_LENGTH: 250*
- *OPTIONS:*
 - *NEGO: 0*
 - *REQ_SX: 0*
 - *REQ_XPRESS: 0*
 - *RESP_SX: 0*
 - *RESP_XPRESS: 0*

- *RESERVED*: 0

- *TYPE*: *text/xml*

4.1.3 New Session Request

The client sends a request to start a new session:

0E 10 00 04 00 00 00 08 00 00 02 89 01 00 00 00?....
74 65 78 74 2F 78 6D 6C EF BB BF 3C 45 6E 76 65	text/xml???<Enve
6C 6F 70 65 20 78 6D 6C 6E 73 3D 22 68 74 74 70	lope xmlns="http
3A 2F 2F 73 63 68 65 6D 61 73 2E 78 6D 6C 73 6F	://schemas.xmlso
61 70 2E 6F 72 67 2F 73 6F 61 70 2F 65 6E 76 65	ap.org/soap/enve
6C 6F 70 65 2F 22 3E 0D 0A 20 20 3C 48 65 61 64	lope/">.. <Head
65 72 3E 0D 0A 20 20 20 20 3C 42 65 67 69 6E 53	er>.. <BeginS
65 73 73 69 6F 6E 20 73 6F 61 70 3A 6D 75 73 74	ession soap:must
55 6E 64 65 72 73 74 61 6E 64 3D 22 31 22 20 78	Understand="1" x
6D 6C 6E 73 3A 73 6F 61 70 3D 22 68 74 74 70 3A	mlns:soap="http:
2F 2F 73 63 68 65 6D 61 73 2E 78 6D 6C 73 6F 61	//schemas.xmlsoa
70 2E 6F 72 67 2F 73 6F 61 70 2F 65 6E 76 65 6C	p.org/soap/envel
6F 70 65 2F 22 20 78 6D 6C 6E 73 3D 22 75 72 6E	ope/" xmlns="urn
3A 73 63 68 65 6D 61 73 2D 6D 69 63 72 6F 73 6F	:schemas-microso
66 74 2D 63 6F 6D 3A 78 6D 6C 2D 61 6E 61 6C 79	ft-com:xml-analy
73 69 73 22 20 2F 3E 0D 0A 20 20 20 20 3C 56 65	sis" />.. <Ve
72 73 69 6F 6E 20 53 65 71 75 65 6E 63 65 3D 22	rsion Sequence="
31 30 30 22 20 78 6D 6C 6E 73 3D 22 68 74 74 70	100" xmlns="http
3A 2F 2F 73 63 68 65 6D 61 73 2E 6D 69 63 72 6F	://schemas.micro
73 6F 66 74 2E 63 6F 6D 2F 61 6E 61 6C 79 73 69	soft.com/analysi
73 73 65 72 76 69 63 65 73 2F 32 30 30 33 2F 65	sservices/2003/e

6E 67 69 6E 65 2F 32 22 20 2F 3E 0D 0A 20 20 3C	engine/2" />.. <
2F 48 65 61 64 65 72 3E 0D 0A 20 20 3C 42 6F 64	/Header>.. <Bod
79 3E 0D 0A 20 20 20 20 3C 45 78 65 63 75 74 65	y>.. <Execute
20 78 6D 6C 6E 73 3D 22 75 72 6E 3A 73 63 68 65	xmlns="urn:sche
6D 61 73 2D 6D 69 63 72 6F 73 6F 66 74 2D 63 6F	mas-microsoft-co
6D 3A 78 6D 6C 2D 61 6E 61 6C 79 73 69 73 22 3E	m:xml-analysis">
0D 0A 20 20 20 20 20 20 3C 43 6F 6D 6D 61 6E 64	.. <Command
3E 0D 0A 20 20 20 20 20 20 20 3C 53 74 61 74	>.. <Stat
65 6D 65 6E 74 20 2F 3E 0D 0A 20 20 20 20 20 20	ement />..
3C 2F 43 6F 6D 6D 61 6E 64 3E 0D 0A 20 20 20 20	</Command>..
20 20 3C 50 72 6F 70 65 72 74 69 65 73 3E 0D 0A	<Properties>..
20 20 20 20 20 20 20 20 3C 50 72 6F 70 65 72 74	<Propert
79 4C 69 73 74 3E 0D 0A 20 20 20 20 20 20 20 20	yList>..
20 20 3C 4C 6F 63 61 6C 65 49 64 65 6E 74 69 66	<LocaleIdentif
69 65 72 3E 31 30 33 33 3C 2F 4C 6F 63 61 6C 65	ier>1033</Locale
49 64 65 6E 74 69 66 69 65 72 3E 0D 0A 20 20 20	Identifier>..
20 20 20 20 20 3C 2F 50 72 6F 70 65 72 74 79 4C	</PropertyL
69 73 74 3E 0D 0A 20 20 20 20 20 20 3C 2F 50 72	ist>.. </Pr
6F 70 65 72 74 69 65 73 3E 0D 0A 20 20 20 20 3C	operties>.. <
2F 45 78 65 63 75 74 65 3E 0D 0A 20 20 3C 2F 42	/Execute>.. </B
6F 64 79 3E 0D 0A 3C 2F 45 6E 76 65 6C 6F 70 65	ody>..</Envelope
3E CC CC CC	>???

Highlighted is the DIME header and the padding required for the 4 byte alignment:

- *VERSION: 1*

- *MB: 1*

- *ME*: 1
- *CF*: 0
- *TYPE_T*: 1
- *RESERVED*: 0
- *OPTIONS_LENGTH*: 4
- *ID_LENGTH*: 0
- *TYPE_LENGTH*: 8
- *DATA_LENGTH*: 649
- *OPTIONS*:
 - *NEGO*: 1
 - *REQ_SX*: 0
 - *REQ_XPRESS*: 0
 - *RESP_SX*: 0
 - *RESP_XPRESS*: 0
 - *RESERVED*: 0
- *TYPE*: *text/xml*

The server responds with the ID of a newly-created session:

0E 10 00 04 00 00 00 08 00 00 01 91 00 00 00 00?....
74 65 78 74 2F 78 6D 6C 3C 73 6F 61 70 3A 45 6E	text/xml<soap:En
76 65 6C 6F 70 65 20 78 6D 6C 6E 73 3A 73 6F 61	velope xmlns:soa
70 3D 22 68 74 74 70 3A 2F 2F 73 63 68 65 6D 61	p="http://schema
73 2E 78 6D 6C 73 6F 61 70 2E 6F 72 67 2F 73 6F	s.xmlsoap.org/so
61 70 2F 65 6E 76 65 6C 6F 70 65 2F 22 3E 3C 73	ap/envelope/"><s
6F 61 70 3A 48 65 61 64 65 72 3E 3C 53 65 73 73	oap:Header><Sess

157 of 178

69	6F	6E	20	78	6D	6C	6E	73	3D	22	75	72	6E	3A	73	ion xmlns="urn:s
63	68	65	6D	61	73	2D	6D	69	63	72	6F	73	6F	66	74	chemas-microsoft
2D	63	6F	6D	3A	78	6D	6C	2D	61	6E	61	6C	79	73	69	-com:xml-analysi
73	22	20	53	65	73	73	69	6F	6E	49	64	3D	22	46	39	s" SessionId="F9
44	37	44	42	37	30	2D	32	42	45	32	2D	34	43	35	32	D7DB70-2BE2-4C52
2D	38	46	46	44	2D	31	31	33	44	39	44	31	46	39	44	-8FFD-113D9D1F9D
32	34	22	2F	3E	3C	2F	73	6F	61	70	3A	48	65	61	64	24"/></soap:Head
65	72	3E	3C	73	6F	61	70	3A	42	6F	64	79	3E	3C	45	er><soap:Body><E
78	65	63	75	74	65	52	65	73	70	6F	6E	73	65	20	78	xecuteResponse x
6D	6C	6E	73	3D	22	75	72	6E	3A	73	63	68	65	6D	61	mlns="urn:schema
73	2D	6D	69	63	72	6F	73	6F	66	74	2D	63	6F	6D	3A	s-microsoft-com:
78	6D	6C	2D	61	6E	61	6C	79	73	69	73	22	3E	3C	72	xml-analysis"><r
65	74	75	72	6E	3E	3C	72	6F	6F	74	20	78	6D	6C	6E	eturn><root xmln
73	3D	22	75	72	6E	3A	73	63	68	65	6D	61	73	2D	6D	s="urn:schemas-m
69	63	72	6F	73	6F	66	74	2D	63	6F	6D	3A	78	6D	6C	icrosoft-com:xml
2D	61	6E	61	6C	79	73	69	73	3A	65	6D	70	74	79	22	-analysis:empty"
2F	3E	3C	2F	72	65	74	75	72	6E	3E	3C	2F	45	78	65	/></return></Exe
63	75	74	65	52	65	73	70	6F	6E	73	65	3E	3C	2F	73	cuteResponse></s
6F	61	70	3A	42	6F	64	79	3E	3C	2F	73	6F	61	70	3A	oap:Body></soap:
45	6E	76	65	6C	6F	70	65	3E	CC	CC	CC	Envelope>???

Highlighted is the DIME header and the padding required for the 4 byte alignment:

- *VERSION: 1*

- *MB: 1*

- *ME: 1*

- *CF: 0*

- *TYPE_T*: 1
- *RESERVED*: 0
- *OPTIONS_LENGTH*: 4
- *ID_LENGTH*: 0
- *TYPE_LENGTH*: 8
- *DATA_LENGTH*: 401
- *OPTIONS*:
 - *NEGO*: 0
 - *REQ_SX*: 0
 - *REQ_XPRESS*: 0
 - *RESP_SX*: 0
 - *RESP_XPRESS*: 0
 - *RESERVED*: 0
- *TYPE*: *text/xml*

4.1.4 Request for List of Catalogs

The client sends a DBSCHEMA_CATALOGS request:

<pre> 0E 10 00 04 00 00 00 08 00 00 02 3E 01 00 00 00 74 65 78 74 2F 78 6D 6C EF BB BF 3C 45 6E 76 65 6C 6F 70 65 20 78 6D 6C 6E 73 3D 22 68 74 74 70 3A 2F 2F 73 63 68 65 6D 61 73 2E 78 6D 6C 73 6F 61 70 2E 6F 72 67 2F 73 6F 61 70 2F 65 6E 76 65 6C 6F 70 65 2F 22 3E 0D 0A 20 20 3C 48 65 61 64 65 72 3E 0D 0A 20 20 20 20 3C 58 41 3A 53 65 73 73 69 6F 6E 20 73 6F 61 70 3A 6D 75 73 74 55 6E </pre>	<pre>>.... text/xml???<Enve lope xmlns="http ://schemas.xmlso ap.org/soap/enve lope/">.. <Head er>.. <XA:Ses sion soap:mustUn </pre>
--	--

64 65 72 73	74 61 6E 64	3D 22 31 22	20 53 65 73	derstand="1" Ses
73 69 6F 6E	49 64 3D 22	46 39 44 37	44 42 37 30	sionId="F9D7DB70
2D 32 42 45	32 2D 34 43	35 32 2D 38	46 46 44 2D	-2BE2-4C52-8FFD-
31 31 33 44	39 44 31 46	39 44 32 34	22 20 78 6D	113D9D1F9D24" xm
6C 6E 73 3A	73 6F 61 70	3D 22 68 74	74 70 3A 2F	lns:soap="http:/
2F 73 63 68	65 6D 61 73	2E 78 6D 6C	73 6F 61 70	/schemas.xmlsoap
2E 6F 72 67	2F 73 6F 61	70 2F 65 6E	76 65 6C 6F	.org/soap/envelo
70 65 2F 22	20 78 6D 6C	6E 73 3A 58	41 3D 22 75	pe/" xmlns:XA="u
72 6E 3A 73	63 68 65 6D	61 73 2D 6D	69 63 72 6F	rn:schemas-micro
73 6F 66 74	2D 63 6F 6D	3A 78 6D 6C	2D 61 6E 61	soft-com:xml-ana
6C 79 73 69	73 22 20 2F	3E 0D 0A 20	20 3C 2F 48	lysis" />.. </H
65 61 64 65	72 3E 0D 0A	20 20 3C 42	6F 64 79 3E	earer>.. <Body>
0D 0A 20 20	20 20 3C 44	69 73 63 6F	76 65 72 20	.. <Discover
78 6D 6C 6E	73 3D 22 75	72 6E 3A 73	63 68 65 6D	xmlns="urn:schem
61 73 2D 6D	69 63 72 6F	73 6F 66 74	2D 63 6F 6D	as-microsoft-com
3A 78 6D 6C	2D 61 6E 61	6C 79 73 69	73 22 3E 0D	:xml-analysis">.
0A 20 20 20	20 20 20 3C	52 65 71 75	65 73 74 54	. <RequestT
79 70 65 3E	44 42 53 43	48 45 4D 41	5F 43 41 54	ype>DBSCHEMA_CAT
41 4C 4F 47	53 3C 2F 52	65 71 75 65	73 74 54 79	ALOGS</RequestTy
70 65 3E 0D	0A 20 20 20	20 20 20 3C	52 65 73 74	pe>.. <Rest
72 69 63 74	69 6F 6E 73	3E 3C 2F 52	65 73 74 72	rictions></Restr
69 63 74 69	6F 6E 73 3E	0D 0A 20 20	20 20 20 20	ictions>..
3C 50 72 6F	70 65 72 74	69 65 73 3E	3C 50 72 6F	<Properties><Pro
70 65 72 74	79 4C 69 73	74 3E 3C 43	6F 6E 74 65	pertyList><Conte
6E 74 3E 44	61 74 61 3C	2F 43 6F 6E	74 65 6E 74	nt>Data</Content

```

3E 3C 2F 50 72 6F 70 65 72 74 79 4C 69 73 74 3E    ></PropertyList>
3C 2F 50 72 6F 70 65 72 74 69 65 73 3E 0D 0A 20    </Properties>..
20 20 20 3C 2F 44 69 73 63 6F 76 65 72 3E 0D 0A    </Discover>..
20 20 3C 2F 42 6F 64 79 3E 0D 0A 3C 2F 45 6E 76    </Body>..</Env
65 6C 6F 70 65 3E CC CC .. .. .. .. .. .. .. ..   elope>??

```

Highlighted is the DIME header and the padding required for the 4 byte alignment:

- *VERSION: 1*
- *MB: 1*
- *ME: 1*
- *CF: 0*
- *TYPE_T: 1*
- *RESERVED: 0*
- *OPTIONS_LENGTH: 4*
- *ID_LENGTH: 0*
- *TYPE_LENGTH: 8*
- *DATA_LENGTH: 574*
- *OPTIONS:*
 - *NEGO: 1*
 - *REQ_SX: 0*
 - *REQ_XPRESS: 0*
 - *RESP_SX: 0*
 - *RESP_XPRESS: 0*
 - *RESERVED: 0*
- *TYPE: text/xml*

The server responds with the list of catalogs:

0E 10 00 04 00 00 00 08 00 00 03 5D 00 00 00 00]....
74 65 78 74 2F 78 6D 6C 3C 73 6F 61 70 3A 45 6E	text/xml<soap:En
76 65 6C 6F 70 65 20 78 6D 6C 6E 73 3A 73 6F 61	velope xmlns:soa
70 3D 22 68 74 74 70 3A 2F 2F 73 63 68 65 6D 61	p="http://schema
73 2E 78 6D 6C 73 6F 61 70 2E 6F 72 67 2F 73 6F	s.xmlsoap.org/so
61 70 2F 65 6E 76 65 6C 6F 70 65 2F 22 3E 3C 73	ap/envelope/"><s
6F 61 70 3A 42 6F 64 79 3E 3C 44 69 73 63 6F 76	oap:Body><Discov
65 72 52 65 73 70 6F 6E 73 65 20 78 6D 6C 6E 73	erResponse xmlns
3D 22 75 72 6E 3A 73 63 68 65 6D 61 73 2D 6D 69	="urn:schemas-mi
63 72 6F 73 6F 66 74 2D 63 6F 6D 3A 78 6D 6C 2D	crosoft-com:xml-
61 6E 61 6C 79 73 69 73 22 20 78 6D 6C 6E 73 3A	analysis" xmlns:
64 64 6C 32 3D 22 68 74 74 70 3A 2F 2F 73 63 68	ddl2="http://sch
65 6D 61 73 2E 6D 69 63 72 6F 73 6F 66 74 2E 63	emas.microsoft.c
6F 6D 2F 61 6E 61 6C 79 73 69 73 73 65 72 76 69	om/analysis servi
63 65 73 2F 32 30 30 33 2F 65 6E 67 69 6E 65 2F	ces/2003/engine/
32 22 20 78 6D 6C 6E 73 3A 64 64 6C 32 5F 32 3D	2" xmlns:ddl2_2=
22 68 74 74 70 3A 2F 2F 73 63 68 65 6D 61 73 2E	"http://schemas.
6D 69 63 72 6F 73 6F 66 74 2E 63 6F 6D 2F 61 6E	microsoft.com/an
61 6C 79 73 69 73 73 65 72 76 69 63 65 73 2F 32	alysis services/2
30 30 33 2F 65 6E 67 69 6E 65 2F 32 2F 32 22 20	003/engine/2/2"
78 6D 6C 6E 73 3A 64 64 6C 31 30 30 3D 22 68 74	xmlns:ddl100="ht
74 70 3A 2F 2F 73 63 68 65 6D 61 73 2E 6D 69 63	tp://schemas.mic
72 6F 73 6F 66 74 2E 63 6F 6D 2F 61 6E 61 6C 79	rosoft.com/analy
73 69 73 73 65 72 76 69 63 65 73 2F 32 30 30 38	sis services/2008

2F 65 6E 67	69 6E 65 2F	31 30 30 22	20 78 6D 6C	/engine/100" xml
6E 73 3A 64	64 6C 31 30	30 5F 31 30	30 3D 22 68	ns:ddl100_100="h
74 74 70 3A	2F 2F 73 63	68 65 6D 61	73 2E 6D 69	ttp://schemas.mi
63 72 6F 73	6F 66 74 2E	63 6F 6D 2F	61 6E 61 6C	rosoft.com/anal
79 73 69 73	73 65 72 76	69 63 65 73	2F 32 30 30	ysisservices/200
38 2F 65 6E	67 69 6E 65	2F 31 30 30	2F 31 30 30	8/engine/100/100
22 3E 3C 72	65 74 75 72	6E 3E 3C 72	6F 6F 74 20	"><return><root
78 6D 6C 6E	73 3D 22 75	72 6E 3A 73	63 68 65 6D	xmlns="urn:schem
61 73 2D 6D	69 63 72 6F	73 6F 66 74	2D 63 6F 6D	as-microsoft-com
3A 78 6D 6C	2D 61 6E 61	6C 79 73 69	73 3A 72 6F	:xml-analysis:ro
77 73 65 74	22 20 78 6D	6C 6E 73 3A	78 73 69 3D	wset" xmlns:xsi=
22 68 74 74	70 3A 2F 2F	77 77 77 2E	77 33 2E 6F	"http://www.w3.o
72 67 2F 32	30 30 31 2F	58 4D 4C 53	63 68 65 6D	rg/2001/XMLSchem
61 2D 69 6E	73 74 61 6E	63 65 22 20	78 6D 6C 6E	a-instance" xmln
73 3A 78 73	64 3D 22 68	74 74 70 3A	2F 2F 77 77	s:xsd="http://ww
77 2E 77 33	2E 6F 72 67	2F 32 30 30	31 2F 58 4D	w.w3.org/2001/XM
4C 53 63 68	65 6D 61 22	3E 3C 72 6F	77 3E 3C 43	LSchema"><row><C
41 54 41 4C	4F 47 5F 4E	41 4D 45 3E	44 44 4C 54	ATALOG_NAME>DDL
65 73 74 44	42 3C 2F 43	41 54 41 4C	4F 47 5F 4E	estDB</CATALOG_N
41 4D 45 3E	3C 44 45 53	43 52 49 50	54 49 4F 4E	AME><DESCRIPTION
3E 4D 69 63	72 6F 73 6F	66 74 20 53	61 6D 70 6C	>Microsoft Sampl
65 20 44 61	74 61 62 61	73 65 3C 2F	44 45 53 43	e Database</DESC
52 49 50 54	49 4F 4E 3E	3C 52 4F 4C	45 53 3E 2A	RIPTION><ROLES>*
2C 72 61 2C	72 62 3C 2F	52 4F 4C 45	53 3E 3C 44	,ra,rb</ROLES><D
41 54 45 5F	4D 4F 44 49	46 49 45 44	3E 32 30 30	ATE_MODIFIED>200

38 2D 30 32	2D 30 38 54	30 32 3A 34	37 3A 35 34	8-02-08T02:47:54
3C 2F 44 41	54 45 5F 4D	4F 44 49 46	49 45 44 3E	</DATE_MODIFIED>
3C 2F 72 6F	77 3E 3C 2F	72 6F 6F 74	3E 3C 2F 72	</row></root></r
65 74 75 72	6E 3E 3C 2F	44 69 73 63	6F 76 65 72	eturn></Discover
52 65 73 70	6F 6E 73 65	3E 3C 2F 73	6F 61 70 3A	Response></soap:
42 6F 64 79	3E 3C 2F 73	6F 61 70 3A	45 6E 76 65	Body></soap:Enve
6C 6F 70 65	3E CC CC CC	lope>???

Highlighted is the DIME header and the padding required for the 4 byte alignment:

- *VERSION: 1*
- *MB: 1*
- *ME: 1*
- *CF: 0*
- *TYPE_T: 1*
- *RESERVED: 0*
- *OPTIONS_LENGTH: 4*
- *ID_LENGTH: 0*
- *TYPE_LENGTH: 8*
- *DATA_LENGTH: 861*
- *OPTIONS:*
 - *NEGO: 0*
 - *REQ_SX: 0*
 - *REQ_XPRESS: 0*
 - *RESP_SX: 0*
 - *RESP_XPRESS: 0*

- *RESERVED*: 0

- *TYPE*: *text/xml*

4.1.5 End of Session

The client sends a request to end this session:

0E 10 00 04 00 00 00 08 00 00 02 54 01 00 00 00T....
74 65 78 74 2F 78 6D 6C EF BB BF 3C 45 6E 76 65	text/xml???<Envelope
6C 6F 70 65 20 78 6D 6C 6E 73 3D 22 68 74 74 70	xmlns="http
3A 2F 2F 73 63 68 65 6D 61 73 2E 78 6D 6C 73 6F	://schemas.xmlso
61 70 2E 6F 72 67 2F 73 6F 61 70 2F 65 6E 76 65	ap.org/soap/enve
6C 6F 70 65 2F 22 3E 0D 0A 20 20 3C 48 65 61 64	lope/>.. <Head
65 72 3E 0D 0A 20 20 20 20 3C 45 6E 64 53 65 73	er>.. <EndSes
73 69 6F 6E 20 73 6F 61 70 3A 6D 75 73 74 55 6E	sion soap:mustUn
64 65 72 73 74 61 6E 64 3D 22 31 22 20 53 65 73	derstand="1" Ses
73 69 6F 6E 49 64 3D 22 46 39 44 37 44 42 37 30	sionId="F9D7DB70
2D 32 42 45 32 2D 34 43 35 32 2D 38 46 46 44 2D	-2BE2-4C52-8FFD-
31 31 33 44 39 44 31 46 39 44 32 34 22 20 78 6D	113D9D1F9D24" xm
6C 6E 73 3A 73 6F 61 70 3D 22 68 74 74 70 3A 2F	lns:soap="http:/
2F 73 63 68 65 6D 61 73 2E 78 6D 6C 73 6F 61 70	/schemas.xmlsoap
2E 6F 72 67 2F 73 6F 61 70 2F 65 6E 76 65 6C 6F	.org/soap/envelo
70 65 2F 22 20 78 6D 6C 6E 73 3D 22 75 72 6E 3A	pe/" xmlns="urn:
73 63 68 65 6D 61 73 2D 6D 69 63 72 6F 73 6F 66	schemas-microsof
74 2D 63 6F 6D 3A 78 6D 6C 2D 61 6E 61 6C 79 73	t-com:xml-analys
69 73 22 20 2F 3E 0D 0A 20 20 3C 2F 48 65 61 64	is" />.. </Head
65 72 3E 0D 0A 20 20 3C 42 6F 64 79 3E 0D 0A 20	er>.. <Body>..
20 20 20 3C 45 78 65 63 75 74 65 20 78 6D 6C 6E	<Execute xmln

73 3D 22 75	72 6E 3A 73	63 68 65 6D	61 73 2D 6D	s="urn:schemas-m
69 63 72 6F	73 6F 66 74	2D 63 6F 6D	3A 78 6D 6C	icrosoft-com:xml
2D 61 6E 61	6C 79 73 69	73 22 3E 0D	0A 20 20 20	-analysis">..
20 20 20 3C	43 6F 6D 6D	61 6E 64 3E	0D 0A 20 20	<Command>..
20 20 20 20	20 20 3C 53	74 61 74 65	6D 65 6E 74	<Statement
20 2F 3E 0D	0A 20 20 20	20 20 20 3C	2F 43 6F 6D	/>.. </Com
6D 61 6E 64	3E 0D 0A 20	20 20 20 20	20 3C 50 72	mand>.. <Pr
6F 70 65 72	74 69 65 73	3E 0D 0A 20	20 20 20 20	operties>..
20 20 20 3C	50 72 6F 70	65 72 74 79	4C 69 73 74	<PropertyList
3E 0D 0A 20	20 20 20 20	20 20 20 20	20 3C 4C 6F	>.. <Lo
63 61 6C 65	49 64 65 6E	74 69 66 69	65 72 3E 31	caleIdentifier>1
30 33 33 3C	2F 4C 6F 63	61 6C 65 49	64 65 6E 74	033</LocaleIdent
69 66 69 65	72 3E 0D 0A	20 20 20 20	20 20 20 20	ifier>..
3C 2F 50 72	6F 70 65 72	74 79 4C 69	73 74 3E 0D	</PropertyList>.
0A 20 20 20	20 20 20 3C	2F 50 72 6F	70 65 72 74	. </Propert
69 65 73 3E	0D 0A 20 20	20 20 3C 2F	45 78 65 63	ies>.. </Exec
75 74 65 3E	0D 0A 20 20	3C 2F 42 6F	64 79 3E 0D	ute>.. </Body>.
0A 3C 2F 45	6E 76 65 6C	6F 70 65 3E</Envelope>

Highlighted is the DIME header:

- *VERSION: 1*

- *MB: 1*

- *ME: 1*

- *CF: 0*

- *TYPE_T: 1*

- *RESERVED*: 0
- *OPTIONS_LENGTH*: 4
- *ID_LENGTH*: 0
- *TYPE_LENGTH*: 8
- *DATA_LENGTH*: 596
- *OPTIONS*:
 - *NEGO*: 1
 - *REQ_SX*: 0
 - *REQ_XPRESS*: 0
 - *RESP_SX*: 0
 - *RESP_XPRESS*: 0
 - *RESERVED*: 0
- *TYPE*: *text/xml*

The server responds with a confirmation:

<pre> 0E 10 00 04 00 00 00 08 00 00 01 0C 00 00 00 00 74 65 78 74 2F 78 6D 6C 3C 73 6F 61 70 3A 45 6E 76 65 6C 6F 70 65 20 78 6D 6C 6E 73 3A 73 6F 61 70 3D 22 68 74 74 70 3A 2F 2F 73 63 68 65 6D 61 73 2E 78 6D 6C 73 6F 61 70 2E 6F 72 67 2F 73 6F 61 70 2F 65 6E 76 65 6C 6F 70 65 2F 22 3E 3C 73 6F 61 70 3A 42 6F 64 79 3E 3C 45 78 65 63 75 74 65 52 65 73 70 6F 6E 73 65 20 78 6D 6C 6E 73 3D 22 75 72 6E 3A 73 63 68 65 6D 61 73 2D 6D 69 63 72 6F 73 6F 66 74 2D 63 6F 6D 3A 78 6D 6C 2D 61 </pre>	<pre> text/xml<soap:En velope xmlns:soa p="http://schema s.xmlsoap.org/so ap/envelope/"><s oap:Body><Execut eResponse xmlns= "urn:schemas-mic rosoft-com:xml-a </pre>
--	--

6E 61 6C 79 73 69 73 22 3E 3C 72 65 74 75 72 6E	nalysis"><return
3E 3C 72 6F 6F 74 20 78 6D 6C 6E 73 3D 22 75 72	><root xmlns="ur
6E 3A 73 63 68 65 6D 61 73 2D 6D 69 63 72 6F 73	n:schemas-micros
6F 66 74 2D 63 6F 6D 3A 78 6D 6C 2D 61 6E 61 6C	oft-com:xml-anal
79 73 69 73 3A 65 6D 70 74 79 22 2F 3E 3C 2F 72	ysis:empty"/></r
65 74 75 72 6E 3E 3C 2F 45 78 65 63 75 74 65 52	eturn></ExecuteR
65 73 70 6F 6E 73 65 3E 3C 2F 73 6F 61 70 3A 42	esponse></soap:B
6F 64 79 3E 3C 2F 73 6F 61 70 3A 45 6E 76 65 6C	ody></soap:Envel
6F 70 65 3E	ope>

Highlighted is the DIME header:

- *VERSION: 1*
- *MB: 1*
- *ME: 1*
- *CF: 0*
- *TYPE_T: 1*
- *RESERVED: 0*
- *OPTIONS_LENGTH: 4*
- *ID_LENGTH: 0*
- *TYPE_LENGTH: 8*
- *DATA_LENGTH: 268*
- *OPTIONS:*
 - *NEGO: 0*
 - *REQ_SX: 0*
 - *REQ_XPRESS: 0*

- *RESP_SX*: 0
- *RESP_XPRESS*: 0
- *RESERVED*: 0
- *TYPE*: *text/xml*

The client disconnects from the server.

4.2 Client obtains a list of cubes from the server over HTTP

In this example, the client creates an unauthenticated connection to the server using HTTP and sends a MDSHEMA_CUBES request.

4.2.1 Connection

The server listens on a TCP port for incoming HTTP requests from clients. The client creates a new TCP connection to the server.

4.2.2 New Session Request

The client sends an HTTP header with the request to create new session:

```
POST /as/msmdpump.dll HTTP/1.1
User-Agent: XmlaClient
Content-Type: text/xml
SOAPAction: "urn:schemas-microsoft-com:xml-analysis:Execute"
X-Transport-Caps-Negotiation-Flags: 0,0,0,0,1
Host: valerik02:2390
Content-Length: 647
Expect: 100-continue
Connection: Keep-Alive
```

The server responds with a confirmation:

```
HTTP/1.1 100 Continue
```

The client sends the payload part of the request:

```
<Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
  <Header>
    <BeginSession soap:mustUnderstand="1"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" xmlns="urn:schemas-microsoft-
com:xml-analysis" />
  </Header>
  <Body>
    <Execute xmlns="urn:schemas-microsoft-com:xml-analysis">
      <Command>
```

```

        <Statement />
    </Command>
    <Properties>
        <PropertyList>
            <LocaleIdentifier>1033</LocaleIdentifier>
        </PropertyList>
    </Properties>
</Execute>
</Body>
</Envelope>

```

The server responds with the newly-created session ID:

```

HTTP/1.1 200 OK
Date: Sat, 16 Feb 2008 00:30:34 GMT
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
Transfer-Encoding: chunked
X-Transport-Caps-Negotiation-Flags: 0,0,0,0,0
Content-Type: text/xml

191
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Header><Session xmlns="urn:schemas-microsoft-com:xml-analysis"
SessionId="537C61C6-827C-4305-83A6-C8CE4A91001B"/>
  </soap:Header>
  <soap:Body>
    <ExecuteResponse xmlns="urn:schemas-microsoft-com:xml-analysis">
      <return>
        <root xmlns="urn:schemas-microsoft-com:xml-analysis:empty"/>
      </return>
    </ExecuteResponse>
  </soap:Body>
</soap:Envelope>
0

```

4.2.3 Request for List of Cubes

The client sends a header for a MDSHEMA_CUBES request:

```

POST /as/msmdpump.dll HTTP/1.1
User-Agent: XmlaClient
Content-Type: text/xml
SOAPAction: "urn:schemas-microsoft-com:xml-analysis:Discover"
X-Transport-Caps-Negotiation-Flags: 1,0,0,0,0
Host: valerik02:2390
Content-Length: 571
Expect: 100-continue

```

The server responds with a confirmation:

170 of 178

HTTP/1.1 100 Continue

The client sends the payload portion of the request:

```
<Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
  <Header>
    <XA:Session soap:mustUnderstand="1" SessionId="537C61C6-827C-4305-83A6-C8CE4A91001B"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" xmlns:XA="urn:schemas-microsoft-
com:xml-analysis" />
  </Header>
  <Body>
    <Discover xmlns="urn:schemas-microsoft-com:xml-analysis">
      <RequestType>MDSHEMA_CUBES</RequestType>
      <Restrictions></Restrictions>
      <Properties><PropertyList><Content>Data</Content></PropertyList></Properties>
    </Discover>
  </Body>
</Envelope>
```

The server responds with an empty list of cubes:

```
HTTP/1.1 200 OK
Date: Sat, 16 Feb 2008 00:30:34 GMT
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
Transfer-Encoding: chunked
X-Transport-Caps-Negotiation-Flags: 0,0,0,0,0
Content-Type: text/xml

208
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <DiscoverResponse xmlns="urn:schemas-microsoft-com:xml-analysis"
xmlns:ddl2="http://schemas.microsoft.com/analysisisservices/2003/engine/2"
xmlns:ddl2_2="http://schemas.microsoft.com/analysisisservices/2003/engine/2/2">
      <return>
        <root xmlns="urn:schemas-microsoft-com:xml-analysis:rowset"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"/>
      </return>
    </DiscoverResponse>
  </soap:Body>
</soap:Envelope>
0
```

4.2.4 End of Session

The client sends a header of the request to end this session:

POST /as/msmdpump.dll HTTP/1.1

171 of 178

[MS-SSAS9] - v1.02

SQL Server Analysis Services Protocol Specification

Copyright © 2008 Microsoft Corporation.

Release: December 12, 2008

User-Agent: XmlaClient
Content-Type: text/xml
SOAPAction: "urn:schemas-microsoft-com:xml-analysis:Execute"
X-Transport-Caps-Negotiation-Flags: 1,0,0,0,0
Host: valerik02:2390
Content-Length: 596
Expect: 100-continue

The server responds with a confirmation:

HTTP/1.1 100 Continue

The client sends the payload portion of the request:

```
<Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
  <Header>
    <EndSession soap:mustUnderstand="1" SessionId="537C61C6-827C-4305-83A6-C8CE4A91001B"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" xmlns="urn:schemas-microsoft-
com:xml-analysis" />
  </Header>
  <Body>
    <Execute xmlns="urn:schemas-microsoft-com:xml-analysis">
      <Command>
        <Statement />
      </Command>
      <Properties>
        <PropertyList>
          <LocaleIdentifier>1033</LocaleIdentifier>
        </PropertyList>
      </Properties>
    </Execute>
  </Body>
</Envelope>
```

The server responds with a confirmation:

HTTP/1.1 200 OK
Date: Sat, 16 Feb 2008 00:30:35 GMT
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
Transfer-Encoding: chunked
X-Transport-Caps-Negotiation-Flags: 0,0,0,0,0
Content-Type: text/xml

10c

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ExecuteResponse xmlns="urn:schemas-microsoft-com:xml-analysis">
      <return>
```

```

        <root xmlns="urn:schemas-microsoft-com:xml-analysis:empty"/>
    </return>
</ExecuteResponse>
</soap:Body>
</soap:Envelope>
0

```

The client closes the connection.

5 Security

The server could be returning potentially sensitive data in its responses. Hence it is strongly recommended that the server be configured to use GSS-API based encryption over TCP or Secure Sockets Layer (SSL) over HTTPS to ensure the integrity of the data and to prevent tampering and unauthorized access.

There are two strategies for reducing the impact of denial-of-service (DOS) attacks against the server:

- Turn on authentication and deny access to unauthenticated clients. This will allow one to quickly disable access to rogue client machines.
- Make sure no single request takes too much processing time on the server. That will ensure that any attacker must keep up a steady stream of requests to deny access to the server, and so a simple network trace will allow one to identify the offending machine and shut it down. This applies to requests sent by "spoof clients" (for example, a virus emulating a client, which might try to pass an unbounded request or a long running MDX query).

6 Appendix A: Product Behavior

The information in this specification is applicable to the following Microsoft products and technologies:

- 2007 Microsoft® Office system
- Microsoft® SQL Server® 2005

Exceptions, if any, are noted below. Unless otherwise specified, any statement of optional behavior in this specification prescribed using the terms SHOULD or SHOULD NOT implies the aforementioned Microsoft products' behavior is in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term MAY implies these Microsoft products do not follow the prescription.

-
- <1> [Section 2.1](#): SQL Server 2005 Analysis Services (SSAS2005) enforces the use of encryption, compression and binary XML in the protocol by default. It can however be configured to turn off one or more of these capabilities.
- <2> [Section 2.1](#): SSAS2005 implements authentication and encryption using [\[MSDN-SSPI\]](#) which is based on GSS-API.
- <3> [Section 2.2.4.1.3.1.1](#): SSAS2005 supports only the Tuples set type. It does not support Members, CrossProduct and Union set types.
- <4> [Section 2.2.4.3.3](#): SSAS2005 always returns the Messages sub-element following the Exception sub-element.
- <5> [Section 2.3.1](#): SSAS2005 does break down DIME messages into multiple records.
- <6> [Section 2.3.1](#): SSAS2005 sets the RESERVED field in the DIME OPTIONS field to 0.
- <7> [Section 2.3.1](#): SSAS2005 uses the content type, application/sx+xpress, by default. It can however be configured to use the other content types – text/xml, application/sx, application/xml+xpress.
- <8> [Section 3.1.3](#): SSAS2005 supports sessions.
- <9> [Section 3.1.3](#): SSAS2005 ends sessions automatically after a period of inactivity that can be configured.
- <10> [Section 3.1.4.2.2.1.1](#): SSAS2005 supports several implementation-specific Discover request types in addition to those described in section 3.1.4.2.3.
- <11> [Section 3.1.4.2.2.1.2](#): SSAS2005 supports all of the request types, restrictions and columns described in section 3.1.4.2.3.
- <12> [Section 3.1.4.2.3.1.2](#): SSAS2005 includes an asterisk (*) in the ROLES column if the current user is a server administrator or database administrator. SSAS2005 includes the system user name as a role if one of the roles uses dynamic security.
- <13> [Section 3.1.4.2.3.2.2](#): SSAS2005 does not support the concept of schemas and always returns NULL for SCHEMA_NAME in all Discover responses containing this column.
- <14> [Section 3.1.4.2.3.2.2](#): SSAS2005 always returns TRUE for the IS_DRILLTHROUGH_ENABLED column.

-
- <15> [Section 3.1.4.2.3.3.1](#): SSAS2005 does not support the concept of schemas and always returns NULL for SCHEMA_NAME in all Discover responses containing this column.
- <16> [Section 3.1.4.2.3.3.2](#): SSAS2005 does not support the concept of schemas and always returns NULL for SCHEMA_NAME in all Discover responses containing this column.
- <17> [Section 3.1.4.2.3.3.2](#): SSAS2005 always returns FALSE for the IS_VIRTUAL column.
- <18> [Section 3.1.4.2.3.3.2](#): SSAS2005 always returns TRUE for the DIMENSION_IS_VISIBLE column.
- <19> [Section 3.1.4.2.3.4.1](#): SSAS2005 does not support the concept of schemas and always returns NULL for SCHEMA_NAME in all Discover responses containing this column.
- <20> [Section 3.1.4.2.3.4.2](#): For SSAS2005 providers that generate unique names by qualification, each component of this unique dimension name is delimited.
- <21> [Section 3.1.4.2.3.4.2](#): SSAS2005 always returns a value for the HIERARCHY_NAME column.
- <22> [Section 3.1.4.2.3.4.2](#): SSAS2005 always returns FALSE for the IS_VIRTUAL column.
- <23> [Section 3.1.4.2.3.4.2](#): SSAS2005 returns TRUE if the WritebackToDimension column that represents this column is enabled.
- <24> [Section 3.1.4.2.3.4.2](#): SSAS2005 always returns 0x01 for the DIMENSION_UNIQUE_SETTINGS column.
- <25> [Section 3.1.4.2.3.4.2](#): SSAS2005 always misses the DIMENSION_MASTER_UNIQUE_NAME column.
- <26> [Section 3.1.4.2.3.4.2](#): SSAS2005 always returns TRUE for the DIMENSION_IS_VISIBLE column. If the dimension is not visible, it will not appear in the schema rowset.
- <27> [Section 3.1.4.2.3.4.2](#): SSAS2005 always returns TRUE for the DIMENSION_IS_SHARED column.

-
- <28> [Section 3.1.4.2.3.5.2](#): For SSAS2005 providers that generate unique names by qualification, each component of the unique dimension name is delimited.
- <29> [Section 3.1.4.2.3.5.2](#): In SSAS2005, if a caption does not exist, LEVEL_NAME is returned.
- <30> [Section 3.1.4.2.3.6.2](#): SSAS2005 folder names are separated by a semicolon. Nested folders are indicated by a backslash (\).
- <31> [Section 3.1.4.2.3.7.2](#): In SSAS2005, if the key for the property is the same as the name for the property, the PROPERTY_NAME column will be blank.
- <32> [Section 3.1.4.2.3.7.2](#): SSAS2005 returns the PROPERTY_NAME column if a caption does not exist.
- <33> [Section 3.1.4.2.3.8.2](#): In SSAS2005, the member type value 4 (formula) takes precedence over the member type value 3 (measure). For example, if there is a formula (calculated) member on the Measures dimension, it is listed as 4.
- <34> [Section 3.1.4.2.3.10.2](#): In SSAS2005, the backslash (\) is the level separator. To provide multiple display folders, use a semicolon (;) to separate the folders.
- <35> [Section 3.1.4.4](#): SSAS2005 uses 1033 (English - United States) as the default locale identifier. It can however be configured to use other locale identifiers by default.
- <36> [Section 3.1.4.4](#): SSAS2005 returns the value of the MdpropFlatteningSupport property as 1.
- <37> [Section 3.1.4.4](#): SSAS2005 returns the value of the MdpropMdxDrillFunctions property as 3.
- <38> [Section 3.1.4.4](#): SSAS2005 returns the value of the MdpropMdxFormulas property as 63.
- <39> [Section 3.1.4.4](#): SSAS2005 returns the value of the MdpropMdxSetFunctions property as 524287.
- <40> [Section 3.1.4.4](#): SSAS2005 returns the value of the MdpropMdxSubqueries property as 3.
- <41> [Section 3.1.4.4](#): SSAS2005 returns the value of the MdpropNamedLevels property as 3.

<42> [Section 3.1.4.4](#): SSAS2005 returns the value of the ProviderType property as 6. This is a product defect and the expected value is 3.

<43> [Section 3.2.1](#): SSAS2005 supports one default instance and multiple named instances of the server on a single computer. The default TCP port number for the default instance is 2383. In order to connect to a named instance, the client MUST first connect to the SQL Browser service on port 2382, then get the list of named instances on the computer by sending a DISCOVER_INSTANCES request, and then examine the response to determine the TCP port number corresponding to the desired named instance.

Index

A

Abstract data model, 46
Applicability, 11
Attribute groups, 39
Attributes, 38

C

Capability negotiation, 11
Complex types, 12

D

Data model, abstract, 46

E

Examples, overview, 147

F

Fields, vendor-extensible, 11

G

Glossary, 5
Groups, 38

I

Informative references, 8
Initialization, 46

L

Local events, 142

M

Message processing, 48
Messages: attribute groups, 39; attributes,
38; complex types, 12; elements, 12;
groups, 38; namespaces, 12; overview,

11, 12; simple types, 38; syntax, 12;
transport, 11

Microsoft Behavior, 173

N

Namespaces, 12
Normative references, 6

O

Overview, 8

P

Preconditions, 11
Prerequisites, 11

R

References: informative, 8; normative, 6;
overview, 6

Relationship to other protocols, 9

S

Security: overview, 173
Sequencing rules, 48
Simple types, 38
Standards assignments, 11
Syntax, 12

T

Timer events, 141
Timers, 46
Transport, 11
Types: complex, 12; simple, 38

V

Vendor-extensible fields, 11
Versioning, 11