[MS-MSL]:
Mapping Specification Language File Format Structure Specification

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation for protocols, file formats, languages, standards as well as overviews of the interaction among each of these technologies.

- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you may make copies of it in order to develop implementations of the technologies described in the Open Specifications and may distribute portions of it in your implementations using these technologies or your documentation as necessary to properly document the implementation. You may also distribute in your implementation, with or without modification, any schema, IDL's, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications.

- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.

- **Patents.** Microsoft has patents that may cover your implementations of the technologies described in the Open Specifications. Neither this notice nor Microsoft's delivery of the documentation grants any licenses under those or any other Microsoft patents. However, a given Open Specification may be covered by Microsoft's Open Specification Promise (available here: http://www.microsoft.com/interop/osp) or the Community Promise (available here: http://www.microsoft.com/interop/cp/default.mspx). If you would prefer a written license, or if the technologies described in the Open Specifications are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.

- **Trademarks.** The names of companies and products contained in this documentation may be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights.

- **Fictitious Names.** The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

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

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

Revision Summary

Date	Revision History	Revision Class	Comments
09/03/2010	0.1	New	Released new document.
02/09/2011	0.1	No change	No changes to the meaning, language, or formatting of the technical content.

Contents

1   Introduction

This document specifies the file format of the **Mapping Specification Language (MSL)** for the **Entity Data Model (EDM)** version 2.0. MSL is an XML-based language that can be used to define mapping between a conceptual **schema** and a store schema.

The conceptual schema is defined in the EDM by using the **Conceptual Schema Definition Language (CSDL)**, as described in [MC-CSDL]. The EDM defines some well-known primitive types, such as **Edm.String**, that are used as the building blocks for structural types such as **complex types** and **entity types**. **Entities** are instances of entity types (for example, **Customer** or **Employee**) that are richly structured records that have a key.

The store schema defines a relational store containing constructs such as tables, views and foreign key constraints. The store schema is defined in the **Store Schema Definition Language (SSDL)**.

MSL defines the mapping between CSDL and SSDL. The scope of the mapping is contained within the confines of an **entity container**, which is itself a collection of **entity sets** and **association sets**. Each entity set mapping defines mapping for each entity property. Similarly, association set mapping defines mapping for both ends of the association. MSL supports other advanced concepts such as mapping entity set to a stored procedure that is defined in the server store and defining QueryViews that are evaluated against the store schema.

1.1   Glossary

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

XML namespace

The following terms are specific to this document:

association: A named, independent relationship between two **entity type** definitions. Associations in the **EDM** are top-level concept; indeed, the top-level nature of its associations helps distinguish the **EDM** from the Relational Model.

association end: An object that specifies the **entity types** that are related, the roles of each **entity type** in the association, and the cardinality rules for each end of the association. Every association includes two association ends.

association set: For a given association type, an association set can hold instances of that type. The association instance connects **entity** instances that are contained in the **entity sets** that are participating in the association set. An association set description includes the association and the corresponding **entity sets** of the **entity types** that are described in the association.

complex type: A type that represents a set of related information. Like the **entity type**, it consists of one or more properties of the **EDM simple type** or complex types; however, unlike the **entity type**, the complex type does not have an **EntityKey** element or a **NavigationProperty** element.

Conceptual Schema Definition Language (CSDL): An XML-based language that can be used for defining a conceptual schema based on the **Entity Data Model (EDM)**.

declared property: A property that is statically declared by a **Property** element as part of the definition of a structural type. For example, in the context of an **entity type**, the declared property includes all properties of an **entity type** that are represented by the **Property** child elements of the **EntityType** element that defines the **entity type**.

Entity Data Model (EDM): A data model that describes the structure of data, regardless of its stored form, in terms of **entities** and relationships in a way that is universal, portable, and long-lasting.

EDM type: A categorization that includes the following types: **association**, **complex type**, **EDM simple type**, and **entity type**.

EDM simple type: A primitive type (as opposed to a structural type) that is used along with a **complex type** as a building block for creating one or more structural type definitions. An **EDM simple type** can be referred to by name or by a **namespace qualified name** where the **namespace** is "EDM".

entity: An instance of an **entity type**.

entity container: A top-level concept that contains multiple **entity sets** and **association sets**.

entity set: A set for an **entity type** that holds instances of its **entity type** or any of its derived types. Multiple entity set instances can be defined for a given **entity type**.

entity type: A type that represents the structure of a top-level concept, such as a customer or order, in a conceptual model.

foreign key association: An association that allows for the inclusion of foreign key columns in an **entity type** definition as foreign key properties.

function import: A function signature in which function parameters and return types are defined by using one or more **EDM types** (except for associations).

Identifier: A string value that is used to uniquely identify a component of the **CSDL** and that is of type **SimpleIdentifier**.

Mapping Specification Language (MSL): An XML-based language that can be used to define mapping between a conceptual schema and a store schema.

namespace: A name that is defined on the schema and that is subsequently used to prefix **identifiers** to form the **namespace qualified name** of a structural type.

namespace qualified name: A **qualified name** that is used to refer to structural types by using the name of the **namespace** followed by a period and then by the name of the structural type.

qualified name: A string-based representation of the name of an element or attribute.

schema: A concept that defines a **namespace** that describes the scope of **EDM types**. All **EDM types** are contained within some **namespace**.

simple identifier: An **identifier** that conforms to the rules for **identifiers** that are valid in the C# programming language as defined in [ECMA-334]. **MSL** enforces a maximum length of 480 characters for simple identifier values.

Stored Schema Definition Language (SSDL): An XML-based language that can be used to define a store schema.

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

[ECMA-334] ECMA international, "C# Language Specification", ECMA-334, June 2006, http://www.ecma-international.org/publications/standards/Ecma-334.htm

[MC-CSDL] Microsoft Corporation, "Conceptual Schema Definition File Format", February 2009.

[MS-SSDL] Microsoft Corporation, "Store Schema Definition File Format".

[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

[XML1.0] Bray, T., Paoli, J., Sperberg-McQueen, C.M., and Maler, E., "Extensible Markup Language (XML) 1.0 (Second Edition)", W3C Recommendation, October 2000, http://www.w3.org/TR/2000/REC-xml-20001006

[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/

1.2.2   Informative References

[MC-EDMX] Microsoft Corporation, "Entity Data Model for Data Services Packaging Format", February 2009.

[MS-GLOS] Microsoft Corporation, "Windows Protocols Master Glossary", March 2007.

[MSDN-ESO] Microsoft Corporation, "Entity SQL Overview", http://msdn.microsoft.com/en-us/library/bb387145.aspx

1.3   Structure Overview (Synopsis)

Mapping Specification Language (MSL) is an XML-based file format that describes the mapping between a conceptual schema and a store schema and is based on standards that are described in [XML1.0] and [XMLSCHEMA1]. The root of MSL is a **Mapping** element that has the **EntityContainerMapping** child element which, in turn, has the following child elements:

- **EntitySetMapping**

- **AssociationSetMapping**

- **FunctionImportMapping**

Conceptually, an MSL file has an overall structure that resembles the following.

```
<Mapping>
    <EntityContainerMapping>
        <EntitySetMapping>
```

```
            <QueryView/>
            <EntityTypeMapping>
                <MappingFragment>
                    <ComplexProperty>
                        <ScalarProperty/>
                        <ComplexProperty/>
                        <ComplexTypeMapping/>
                        <Condition/>
                    </ComplexProperty>
                    <ScalarProperty/>
                    <Condition/>
                </MappingFragment>
                <ModificationFunctionMapping>
                    <DeleteFunction>
                        <ScalarProperty/>
                        <AssociationEnd/>
                        <ComplexProperty/>
                    </DeleteFunction>
                    <InsertFunction>
                        <ScalarProperty/>
                        <AssociationEnd/>
                        <ComplexProperty/>
                        <ResultBinding/>
                    </InsertFunction>
                    <UpdateFunction>
                        <ScalarProperty/>
                        <AssociationEnd/>
                        <ComplexProperty/>
                        <ResultBinding/>
                    </UpdateFunction>
                </ModificationFunctionMapping>
            </EntityTypeMapping>
            <MappingFragment
                <ComplexProperty
                    <ScalarProperty/>
                    <ComplexProperty/>
                    <ComplexTypeMapping/>
                    <Condition/>
                <ScalarProperty/>
                <Condition/>
    </EntitySetMapping>
    <AssociationSetMapping>
        <QueryView/>
        <EndProperty>
            <ScalarProperty/>
        </EndProperty>
        <Condition/>
        <ModificationFunctionMapping>
            <DeleteFunction>
                <EndProperty>
                    <ScalarProperty/>
                </EndProperty>
            </DeleteFunction>
            <InsertFunction>
                <EndProperty>
                    <ScalarProperty/>
                </EndProperty>
            </InsertFunction>
        </ModificationFunctionMapping>
```

```
                </AssociationSetMapping>
                <FunctionImportMapping>
                    <ResultMapping>
                        <EntityTypeMapping>
                            <ScalarProperty/>
                            <Condition/>
                        </EntityTypeMapping>
                        <ComplexTypeMapping>
                            <ScalarProperty/>
                        </ComplexTypeMapping>
                    </ResultMapping>
                </FunctionImportMapping>
            </EntityContainerMapping>
    </Mapping>
```
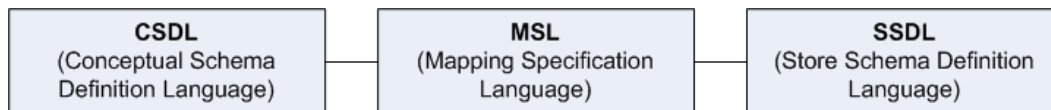
Note   The preceding code snippet is not a detailed specification; it is meant to provide a visual overview. For a detailed specification, see section 2.1.1.

The following figure shows how MSL defines the mapping between the conceptual schema, specified in [MC-CSDL], and the store schema, specified in [MS-SSDL].

CSDL (Conceptual Schema Definition Language)	**MSL** (Mapping Specification Language)	**SSDL** (Store Schema Definition Language)

Figure 1: MSL defines the mapping between CSDL and SSDL

1.4   Relationship to Protocols and Other Structures

None.

1.5   Applicability Statement

MSL is an XML format that describes the structure and semantics of mapping between the conceptual schema and the store schema.

1.6   Versioning and Localization

This document describes the structures for MSL version 1.0 and MSL version 2.0. Aspects of MSL 1.0 that do not apply to MSL 2.0 are specifically highlighted.

MSL 1.0 has a slightly reduced set of capabilities (which are called out in this document) than MSL 2.0. This version of MSL references the following **XML namespace**:

urn:schemas-microsoft-com:windows:storage:mapping:CS

The following rules apply to MSL version 1.0:

- The **EntityContainerMapping** element MUST NOT specify the **GeneratedUpdateViews** attribute.

- The **EntitySetMapping** element MUST NOT specify the **MakeColumnsDistinct** attribute.

- The **MappingFragment** element MUST NOT specify the **MakeColumnsDistinct** attribute.

- The **FunctionImportMapping** element MUST NOT specify the **ResultMapping** child element.

- In the **ModificationFunctionMapping** element, if one of the following child elements is specified, all three child elements MUST be specified.

- **DeleteFunction**

- **InsertFunction**

- **UpdateFunction**

MSL 2.0 is a superset of MSL version 1.0, and is the focus of this document. This version of MSL references the following XML namespace:

http://schemas.microsoft.com/ado/2008/09/mapping/cs

2 Structures

2.1 Elements

2.1.1 Mapping

The root level **Mapping** element can have zero or more **Alias** elements followed by an **EntityContainerMapping** element. The **Mapping** element in Mapping Specification Language (MSL) contains information for mapping objects that are specified in a conceptual schema to a database that is in a store schema. For related documents, see section 1.2.2.

The following is the XML schema definition of the **Mapping** element.

```
<xs:complexType name="TMapping">
  <xs:sequence>
    <xs:element minOccurs="0" maxOccurs="unbounded" name="Alias" type="csmsl:TAlias"/>
    <xs:element name="EntityContainerMapping" type="csmsl:TEntityContainerMapping"/>
  </xs:sequence>
  <xs:attribute name="Space" type="csmsl:TSpace" use="required" fixed="C-S" />
</xs:complexType>
<xs:simpleType name="TSpace">
  <xs:restriction base="xs:token">
    <xs:enumeration value="C-S" />
  </xs:restriction>
</xs:simpleType>
```

The following additional rules apply to the **Mapping** element:

- The MSL document MUST have the **Mapping** element as its root element.

- The **Mapping** element MUST have a **Space** attribute specified that is of type **SimpleIdentifier**. In MSL version 1.0 and in MSL version 2.0, "C-S" is the only valid value for the **Space** attribute.

- A mapping definition MUST NOT span multiple MSL documents.

- The **Mapping** element MUST contain only one **EntityContainerMapping** child element.

2.1.2 Alias

The **Alias** element contains two attributes, **Key** and **Value**. The **Key** attribute is a **simple identifier** that is typically used as a short name for a **namespace**. The **Value** attribute is the namespace. For example, if an entity type named "Person" is specified in the "Model.Business" namespace, and if that namespace has been given the alias "Self", the alias **qualified name** for the "Person" entity type is "Self.Person".

The following is the XML schema definition of the **Alias** element.

```
<xs:complexType name="TAlias">
  <xs:attribute name="Key" type="csmsl:TSimpleIdentifier" use="required" />
  <xs:attribute name="Value" type="xs:string" use="required" />
</xs:complexType>
```

2.1.3 EntityContainerMapping

The **EntityContainerMapping** element in Mapping Specification Language (MSL) maps the entity container in the conceptual schema to the entity container in the store schema. The **EntityContainerMapping** element is a child element of the **Mapping** element.

The following is the XML schema definition of the **EntityContainerMapping** element.

```
   <xs:complexType name="TEntityContainerMapping">
     <xs:sequence>
       <xs:choice maxOccurs="unbounded">
         <xs:element minOccurs="0" name="EntitySetMapping" type="csmsl:TEntitySetMapping"/>
         <xs:element minOccurs="0" name="AssociationSetMapping"
  type="csmsl:TAssociationSetMapping"/>
         <xs:element minOccurs="0" name="FunctionImportMapping"
  type="csmsl:TFunctionImportMapping"/>
       </xs:choice>
     </xs:sequence>
     <xs:attribute name="CdmEntityContainer" type="csmsl:TSimpleIdentifier" use="required" />
     <xs:attribute name="StorageEntityContainer" type="xs:string" use="required" />
     <xs:attribute name="GenerateUpdateViews" type="xs:boolean" use="optional" />
   </xs:complexType>
```

The following additional rules apply to the **EntityContainerMapping** element:

- The **CmdEntityContainer** element identifies the conceptual schema container that is participating in the **EntityContainerElement** element.

- The **StorageEntityContainer** element identifies the store schema container that is participating in the **EntityContainer** element.

- In MSL version 2.0, the **EntityContainerMapping** element MAY specify the **GenerateUpdateViews** attribute. Possible values for the **GenerateUpdateViews** attribute are true or false. The default value is true. If the value is false, update views are not generated and round-tripping validation is skipped.

2.1.4 EntitySetMapping

The **EntitySetMapping** element is a child element of the **EntityContainerMapping** element. The **EntitySetMapping** element specifies the mapping for all entity types in an entity set that is in a conceptual schema, to entity sets in the store schema.

An entity set in the conceptual schema is a logical container for entities of the same type and derived types. An entity set in the store schema represents a table or view in the underlying database. The entity set in the conceptual schema is specified by the value of the **Name** attribute of the **EntitySetMapping** element. The table or view that is mapped to is specified by the **StoreEntitySet** attribute, or it is specified as an attribute on **MappingFragment** child elements.

The following is the XML schema definition of the **EntitySetMapping** element.

```
   <xs:complexType name="TEntitySetMapping">
     <xs:choice>
       <xs:choice>
         <xs:sequence>
           <xs:element name="QueryView" type="csmsl:TQueryView" minOccurs="0"
  maxOccurs="unbounded"/>
```

```
            <xs:element name="EntityTypeMapping" type="csmsl:TEntityTypeMapping" minOccurs="0"
maxOccurs="unbounded"/>
          </xs:sequence>
          <xs:sequence>
            <xs:element name="MappingFragment" type="csmsl:TMappingFragment" minOccurs="0"
maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:choice>
        <xs:group ref="csmsl:TPropertyGroup"/>
      </xs:choice>
      <xs:attribute name="Name" type="csmsl:TSimpleIdentifier" use="required" />
      <xs:attribute name="TypeName" type="xs:string" use="optional" />
      <xs:attribute name="StoreEntitySet" type="xs:string" use="optional" />
      <xs:attribute name="MakeColumnsDistinct" type="xs:boolean" use="optional" />
  </xs:complexType>
  <xs:group name="TPropertyGroup">
    <xs:sequence>
      <xs:choice maxOccurs="unbounded">
        <xs:element minOccurs="0" name="ComplexProperty" type="csmsl:TComplexProperty"/>
        <xs:element minOccurs="0" name="ScalarProperty" type="csmsl:TScalarProperty"/>
        <xs:element minOccurs="0" name="Condition" type="csmsl:TCondition"/>
      </xs:choice>
    </xs:sequence>
  </xs:group>
```

The following additional rules apply to the **EntitySetMapping** element:

- The **Name** attribute represents the conceptual entity set that is participating in this **EntitySetMapping** element.

- The **EntitySetMapping** element MAY specify the **TypeName** attribute. If the **TypeName** attribute is not specified as an attribute on the **EntitySetMapping** element, it MUST be specified on the **EntitySetMapping** child element. **TypeName** is the fully qualified type name of the entity type that is participating in **EntitySetMapping**.

- The **EntitySetMapping** element MAY specify the **StoreEntitySet** attribute. If the **StoreEntitySet** attribute is not specified as an attribute on **EntitySetMapping**, it MUST be specified on the **MappingFragment** child element. The **StoreEntitySet** attribute is the name of the store entity set that is participating in **EntitySetMapping**.

- In **MSL** 2.0, the **EntitySetMapping** element MAY specify the **MakeColumnsDistinct** attribute. The possible values are true and false. The default value is false. If the **MakeColumnsDistinct** attribute is set to true, it enforces that only distinct rows are returned. If the value of this attribute is set to true, the value of the **GenerateUpdateViews** attribute of the **EntityContainerMapping** element MUST be set to false.

- The **EntitySetMapping** element MAY specify one or more **QueryView** child elements. If the **QueryView** child element is specified, the **EntitySetMapping** child element MAY be defined to specify the **insert**, **update**, or **delete** functions for the entity type but MUST NOT specify any other child elements, and **EntitySetMapping** MUST NOT specify the **StoreEntitySet** attribute.

- The **EntitySetMapping** element MAY contain any number of **EntityTypeMapping** child elements. If one or more **EntityTypeMapping** child elements are specified, the **EntitySetMapping** element MUST NOT specify any of the following as child elements:

  - **ComplexProperty**

- **Condition**

- **MappingFragment**

- **QueryView**

- **ScalarProperty**

- The **EntitySetMapping** element MAY contain any number of **MappingFragment** child elements. If one or more **MappingFragment** child elements are specified, the **EntitySetMapping** element MUST NOT specify any of the following as child elements:

- **ComplexProperty**

- **Condition**

- **EntityTypeMapping**

- **QueryView**

- **ScalarProperty**

2.1.5 EntityTypeMapping

The **EntityTypeMapping** element is a child element of the **EntitySetMapping** element and also of the **ResultMapping** element.

When it is a child element of the **EntitySetMapping** element, the **EntityTypeMapping** element specifies the mapping between an entity type in the conceptual schema and tables or views in the underlying database. The conceptual schema entity type that is being mapped is specified by the **TypeName** attribute of the **EntityTypeMapping** element. The table or view that is being mapped is specified by the **StoreEntitySet** attribute of the child **MappingFragment** element.

The following is the XML schema definition of the **EntityTypeMapping** element.

```
   <xs:complexType name="TEntityTypeMapping">
     <xs:sequence>
       <xs:element minOccurs="0" maxOccurs="unbounded" name="MappingFragment"
 type="csmsl:TMappingFragment"/>
       <xs:element minOccurs="0" maxOccurs="1" name="ModificationFunctionMapping"
 type="csmsl:TEntityTypeModificationFunctionMapping"/>
     </xs:sequence>
     <xs:attribute name="TypeName" type="xs:string" use="required" />
   </xs:complexType>
```

The following additional rules apply to the **EntityTypeMapping** element:

- The **EntityTypeMapping** element MUST have a **TypeName** attribute specified. The **TypeName** attribute specifies a single **TypeName** or set of **TypeNames**, which are separated by a semi-colon. A single **TypeName** or set of **TypeNames** can also appear as attribute(s) to the **IsTypeOf** keyword. If more than one **TypeName** attribute is specified inside the **IsTypeOf** keyword, they are separated by a semi-colon. The **TypeName** attribute specifies the entity type that is being mapped. The **IsTypeOf** keyword implies that the mapping applies to this given type or types and to all of its derived types.

- The **EntityTypeMapping** element MAY specify any number of **MappingFragment** child elements. Each **MappingFragment** child element specifies mapping to a certain store table or

view. If one or more **MappingFragment** child elements are specified, the **EntityTypeMapping** element MUST NOT specify **ComplexProperty**, **ScalarProperty** or **Condition** as its child elements.

- The **EntityTypeMapping** element MAY specify one **ModificationFunctionMapping** child element. The **ModificationFunctionMapping** child element is used to map the **insert**, **update**, or **delete** functions of entity types to stored procedures in the database.

2.1.6 MappingFragment

The **MappingFragment** element specifies the mapping between the properties of a conceptual schema entity type and a table or view in the database.

The following is the XML schema definition of the **MappingFragment** element.

```
<xs:complexType name="TMappingFragment">
  <xs:group ref="csmsl:TPropertyGroup" minOccurs="1" maxOccurs="1"/>
  <xs:attribute name="StoreEntitySet" type="xs:string" use="required" />
  <xs:attribute name="MakeColumnsDistinct" type="xs:boolean" use="optional" />
</xs:complexType>
<xs:group name="TPropertyGroup">
  <xs:sequence>
    <xs:choice maxOccurs="unbounded">
      <xs:element minOccurs="0" name="ComplexProperty" type="csmsl:TComplexProperty"/>
      <xs:element minOccurs="0" name="ScalarProperty" type="csmsl:TScalarProperty"/>
      <xs:element minOccurs="0" name="Condition" type="csmsl:TCondition"/>
    </xs:choice>
  </xs:sequence>
</xs:group>
```

The following additional rules apply to the **MappingFragment** element:

- The **StoreEntitySet** attribute specifies the store entity type that is being mapped in this **MappingFragment** element.

- In MSL 2.0, the **MappingFragment** element MAY specify the **MakeColumnsDistinct** attribute. The possible values of the **MakeColumnsDistinct** attribute are true and false. The default value is false. If the value is set to true, it enforces that only distinct rows are returned. If this attribute is set to true, the **GenerateUpdateViews** attribute of the **EntityContainerMapping** element MUST be set to false.

2.1.7 ComplexProperty

The **ComplexProperty** element specifies the mapping between a complex type property on a conceptual schema entity type and table columns in the underlying database. The property-column mappings are specified in child **ScalarProperty** elements.

The following is the XML schema definition of the **ComplexProperty** element.

```
<xs:complexType name="TComplexProperty">
  <xs:sequence>
    <xs:choice maxOccurs="unbounded">
      <xs:element name="ScalarProperty" type="csmsl:TScalarProperty"/>
      <xs:element name="ComplexProperty" type="csmsl:TComplexProperty"/>
      <xs:element name="ComplexTypeMapping" type="csmsl:TComplexTypeMapping"/>
      <xs:element name="Condition" type="csmsl:TCondition"/>
```

```
      </xs:choice>
    </xs:sequence>
    <xs:attribute name="Name" type="csmsl:TSimpleIdentifier" use="required" />
    <xs:attribute name="TypeName" type="xs:string" use="optional" />
    <xs:attribute name="IsPartial" type="xs:boolean" use="optional" />
  </xs:complexType>
```

The following additional rules apply to the **ComplexProperty** element:

▪ The **Name** attribute specifies the name of the complex type that is being mapped.

▪ The **TypeName** attribute specifies the fully qualified type of the complex type that is being
  mapped.

2.1.8 ComplexTypeMapping

The **ComplexTypeMapping** element is a child element of the **ComplexProperty** element and also
of the **ResultMapping** element.

The following is the XML schema definition of the **ComplexTypeMapping** element when it is a child
element of the **ComplexProperty** element.

```
    <xs:complexType name="TComplexTypeMapping">
      <xs:sequence>
        <xs:choice maxOccurs="unbounded">
          <xs:element name="ScalarProperty" type="csmsl:TScalarProperty"/>
          <xs:element name="ComplexProperty" type="csmsl:TComplexProperty"/>
          <xs:element name="Condition" type="csmsl:TCondition"/>
        </xs:choice>
      </xs:sequence>
      <xs:attribute name="TypeName" type="xs:string" use="optional" />
      <xs:attribute name="IsPartial" type="xs:boolean" use="optional" />
    </xs:complexType>
```

The following additional rules apply to the **ComplexTypeMapping** element when it is a child
element of the **ComplexProperty** element:

▪ The **ComplexTypeMapping** element MUST have a **TypeName** attribute specified. The
  TypeName attribute specifies a single **TypeName**, or a set of **TypeNames**, that are separated
  by a semicolon. A single **TypeName** attribute or set of **TypeName** attributes can also appear as
  attribute(s) to the **IsTypeOf** keyword. If more than one **TypeName** attribute is specified inside
  the **IsTypeOf** keyword, they are separated by a semicolon. The **TypeName** attribute specifies
  the entity type that is being mapped. The **IsTypeOf** keyword implies that the mapping applies to
  this given type or types and all of its derived types.

2.1.9 ScalarProperty

The **ScalarProperty** element maps a property of primitive type on a conceptual schema entity type,
complex type, or **association end** to a table column or stored procedure parameter in the
underlying database.

The following is the XML schema definition of the **ScalarProperty** element.

```
    <xs:complexType name="TScalarProperty">
      <xs:attribute name="Name" type="csmsl:TSimpleIdentifier" use="required" />
```

```
        <xs:attribute name="ColumnName" type="xs:string" use="required" />
    </xs:complexType>
```

The following additional rules apply to the **ScalarProperty** element:

- The **Name** attribute specifies the name of the scalar property that is specified on entity type, complex type, or **association** in the conceptual schema.

- The ColumnName attribute specifies the name of the table column or stored procedure parameter that is specified on entity type in the store schema.

2.1.10   AssociationSetMapping

The **AssociationSetMapping** element specifies the mapping between an association in the conceptual schema and a table column or columns in the underlying database. Associations in the conceptual schema are types whose properties represent primary and foreign key columns in the underlying database.

The **AssociationSetMapping** element uses two **EndProperty** elements to specify the mappings between association type properties and columns in the database. The user can place the **IsNull=false** condition attribute on these mappings by using the **Condition** element. The user can map the **insert**, **update**, and **delete** functions for associations to stored procedures in the database with the **ModificationFunctionMapping** element. Also, the user can specify read-only mappings between associations and table columns by using an Entity SQL string in a **QueryView** element. For more information about Entity SQL strings, see [MSDN-ESO].

The following is the XML schema definition of the **AssociationSetMapping** element.

```
    <xs:complexType name="TAssociationSetMapping">
      <xs:sequence>
        <xs:element name="QueryView" type="xs:string" minOccurs="0" maxOccurs="1"/>
        <xs:element name="EndProperty" type="csmsl:TEndProperty" minOccurs="0" maxOccurs="2"/>
        <xs:element name="Condition" type="csmsl:TCondition" minOccurs="0"
maxOccurs="unbounded"/>
        <xs:element name="ModificationFunctionMapping"
type="csmsl:TAssociationSetModificationFunctionMapping" minOccurs="0" maxOccurs="1"/>
      </xs:sequence>
      <xs:attribute name="Name" type="csmsl:TSimpleIdentifier" use="required" />
      <xs:attribute name="TypeName" type="csmsl:TQualifiedName" use="optional" />
      <xs:attribute name="StoreEntitySet" type="xs:string" use="optional" />
    </xs:complexType>
```

The following additional rules apply to the **AssociationSetMapping** element:

- The **Name** attribute represents the conceptual association set that is participating in this **AssociationSetMapping** element.

- The **AssociationSetMapping** element MAY specify one **QueryView** child element. If the **QueryView** child element is specified, the **ModificationFunctionMapping** child element MAY be defined to specify the **insert**, **update**, or **delete** functions for the entity type, but the **AssociationSetMapping** element MUST NOT specify any other child elements.

- The **AssociationSetMapping** element MAY specify one **ModificationFunctionMapping** child element or two **EndProperty** child elements but not both.

- The **ModificationFunctionMapping** child element under the **AssociationSetMapping** element MUST NOT have **UpdateFunction** as a child element.

2.1.11 FunctionImportMapping

The **FunctionImportMapping** element specifies the mapping between a **function import** in the conceptual schema and a stored procedure in the underlying database.

The following is the XML schema definition of the **FunctionImportMapping** element.

```
<xs:complexType name="TFunctionImportMapping">
  <xs:sequence>
    <xs:element minOccurs="0" maxOccurs="1" name="ResultMapping"
type="csmsl:TFunctionImportMappingResultMapping"/>
  </xs:sequence>
  <xs:attribute name="FunctionName" type="xs:string" use="required"/>
  <xs:attribute name="FunctionImportName" type="csmsl:TSimpleIdentifier" use="required"/>
</xs:complexType>
```

The following additional rules apply to the **FunctionImportMapping** element:

- The **FunctionImportMapping** attribute represents the function import, as specified in the conceptual schema, that is participating in this **FunctionImportMapping** element.

- The **FunctionName** attribute represents the function import, as specified in the store schema, that is participating in this **FunctionImportMapping** element.

- In MSL version 2.0, the **FunctionImportMapping** element MAY specify the **ResultMapping** child element. The **ResultMapping** child element is used to specify explicit mapping between conceptual types and to store function-returned results.

2.1.12 ModificationFunctionMapping for Entity Type

The **ModificationFunctionMapping** element maps the **insert**, **update**, and **delete** functions of a conceptual schema entity type to stored procedures in the underlying database.

The following is the XML schema definition of the **ModificationFunctionMapping** element.

```
<xs:complexType name="TEntityTypeModificationFunctionMapping">
  <xs:all>
    <xs:element minOccurs="0" maxOccurs="1" name="DeleteFunction"
type="csmsl:TEntityTypeModificationFunction"/>
    <xs:element minOccurs="0" maxOccurs="1" name="InsertFunction"
type="csmsl:TEntityTypeModificationFunctionWithResult"/>
    <xs:element minOccurs="0" maxOccurs="1" name="UpdateFunction"
type="csmsl:TEntityTypeModificationFunctionWithResult"/>
  </xs:all>
</xs:complexType>
```

The following additional rules apply to the **ModificationFunctionMapping** element:

- In MSL version 1.0, if one of the following child elements is defined, all three MUST be defined.

  ▪ **DeleteFunction**

  ▪ **InsertFunction**

▪ **UpdateFunction**

Note   This restriction does not exist in MSL version 2.0.

2.1.13   DeleteFunction for Entity Type

The **DeleteFunction** element maps the **delete** function of an entity type in the conceptual schema to a stored procedure in the underlying database.

The following is the XML schema definition of the **DeleteFunction** element.

```
   <xs:complexType name="TEntityTypeModificationFunction">
     <xs:group ref="csmsl:TEntityTypeFunctionMappingPropertyGroup" minOccurs="1"
 maxOccurs="1"/>
     <xs:attribute name="FunctionName" type="xs:string" use="required"/>
     <xs:attribute name="RowsAffectedParameter" type="xs:string" use="optional"/>
   </xs:complexType>
   <xs:group name="TEntityTypeFunctionMappingPropertyGroup">
     <xs:sequence>
       <xs:choice maxOccurs="unbounded">
         <xs:element minOccurs="0" name="ScalarProperty"
 type="csmsl:TFunctionMappingScalarProperty"/>
         <xs:element minOccurs="0" name="AssociationEnd"
 type="csmsl:TFunctionMappingAssociationEnd"/>
         <xs:element minOccurs="0" name="ComplexProperty"
 type="csmsl:TFunctionMappingComplexProperty"/>
       </xs:choice>
     </xs:sequence>
   </xs:group>
```

The following additional rules apply to the **DeleteFunction** element:

▪ **FunctionName** is the fully qualified name of the stored procedure to which the **delete** function is mapped. The stored procedure MUST be declared in the store schema.

▪ *RowsAffectedParameter* is the name of the output parameter that returns the number of rows that are affected. This output parameter is of type **string**.

2.1.14   InsertFunction for Entity Type

The **InsertFunction** element maps the **insert** function of an entity type in the conceptual schema to a stored procedure in the underlying database.

The following is the XML schema definition of the **InsertFunction** element.

```
   <xs:complexType name="TEntityTypeModificationFunctionWithResult">
     <xs:complexContent>
       <xs:extension base="csmsl:TEntityTypeModificationFunction">
         <xs:group ref="csmsl:TResultBindingGroup" minOccurs="1" maxOccurs="1"/>
       </xs:extension>
     </xs:complexContent>
   </xs:complexType>
   <xs:complexType name="TEntityTypeModificationFunction">
     <xs:group ref="csmsl:TEntityTypeFunctionMappingPropertyGroup" minOccurs="1"
 maxOccurs="1"/>
     <xs:attribute name="FunctionName" type="xs:string" use="required"/>
     <xs:attribute name="RowsAffectedParameter" type="xs:string" use="optional"/>
```

```
      </xs:complexType>
      <xs:group name="TEntityTypeFunctionMappingPropertyGroup">
        <xs:sequence>
          <xs:choice maxOccurs="unbounded">
            <xs:element minOccurs="0" name="ScalarProperty"
  type="csmsl:TFunctionMappingScalarProperty"/>
            <xs:element minOccurs="0" name="AssociationEnd"
  type="csmsl:TFunctionMappingAssociationEnd"/>
            <xs:element minOccurs="0" name="ComplexProperty"
  type="csmsl:TFunctionMappingComplexProperty"/>
          </xs:choice>
        </xs:sequence>
      </xs:group>
      <xs:group name="TResultBindingGroup">
        <xs:sequence>
          <xs:element minOccurs="0" maxOccurs="unbounded" name="ResultBinding"
  type="csmsl:TResultBinding"/>
        </xs:sequence>
      </xs:group>
```

The following additional rules apply to the **InsertFunction** element:

▪ **FunctionName** is the fully qualified name of the stored procedure to which the **insert** function is mapped. The stored procedure MUST be declared in the store schema.

▪ *RowsAffectedParameter* is the name of the output parameter that returns the number of rows affected. This output parameter is of type **string**.

▪ The **ResultBinding** element maps column values that are returned by stored procedures to entity properties in the conceptual schema.

2.1.15 UpdateFunction for Entity Type

The **UpdateFunction** element maps the **update** function of an entity type in the conceptual schema to a stored procedure in the underlying database. Stored procedures to which modification functions are mapped MUST be declared in the store schema.

The following is the XML schema definition of the **UpdateFunction** element.

```
  <xs:complexType name="TEntityTypeModificationFunctionWithResult">
    <xs:complexContent>
      <xs:extension base="csmsl:TEntityTypeModificationFunction">
        <xs:group ref="csmsl:TResultBindingGroup" minOccurs="1" maxOccurs="1"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="TEntityTypeModificationFunction">
    <xs:group ref="csmsl:TEntityTypeFunctionMappingPropertyGroup" minOccurs="1"
  maxOccurs="1"/>
    <xs:attribute name="FunctionName" type="xs:string" use="required"/>
    <xs:attribute name="RowsAffectedParameter" type="xs:string" use="optional"/>
  </xs:complexType>
  <xs:group name="TEntityTypeFunctionMappingPropertyGroup">
    <xs:sequence>
      <xs:choice maxOccurs="unbounded">
        <xs:element minOccurs="0" name="ScalarProperty"
  type="csmsl:TFunctionMappingScalarProperty"/>
```

```
            <xs:element minOccurs="0" name="AssociationEnd"
    type="csmsl:TFunctionMappingAssociationEnd"/>
          <xs:element minOccurs="0" name="ComplexProperty"
    type="csmsl:TFunctionMappingComplexProperty"/>
        </xs:choice>
      </xs:sequence>
    </xs:group>
    <xs:group name="TResultBindingGroup">
      <xs:sequence>
        <xs:element minOccurs="0" maxOccurs="unbounded" name="ResultBinding"
    type="csmsl:TResultBinding"/>
      </xs:sequence>
    </xs:group>
```

The following additional rules apply to the **UpdateFunction** element:

- **FunctionName** is the fully qualified name of the stored procedure to which the **update** function is mapped. The stored procedure MUST be declared in the store schema.

- *RowsAffectedParameter* is the name of the output parameter that returns the number of rows affected. This output parameter is of type **string**.

- The **ResultBinding** element maps column values that are returned by stored procedures to entity properties in the conceptual schema.

2.1.16   ScalarProperty for ModificationFunctionMapping

The **ScalarProperty** element maps a property of primitive type on a conceptual schema entity type, a complex type, or a parameter in the **store** function that is being mapped.

The following is the XML schema definition of the **ScalarProperty** element.

```
    <xs:complexType name="TFunctionMappingScalarProperty">
      <xs:attribute name="ParameterName" type="xs:string" use="required"/>
      <xs:attribute name="Name" type="csmsl:TSimpleIdentifier" use="required"/>
      <xs:attribute name="Version" type="csmsl:TVersion" use="optional"/>
    </xs:complexType>
    <!--Definition for Version, which can 'original' or 'current' as its value-->
    <xs:simpleType name="TVersion">
      <xs:restriction base="xs:token">
        <xs:enumeration value="Original"/>
        <xs:enumeration value="Current"/>
      </xs:restriction>
    </xs:simpleType>
```

The following additional rules apply to the **ScalarProperty** element:

- The **Name** attribute specifies the name of the scalar property that is defined on entity type and complex type.

- The **ColumnName** attribute specifies the name of the parameter on the **store** function that is being mapped in the **ModificationFunctionMapping** element.

- The **Version** attribute indicates whether the current value or the original value of the property SHOULD be used. If the **Version** value is under the **DeleteFunction** element, the value for the

Version attribute can only be **Original**. If the **Version** value is under the **InsertFunction** element, the value for the **Version** attribute can only be **Current**.

2.1.17   ResultBinding

The **ResultBinding** element is used in type modification functions to map column values that are returned by stored procedures to entity properties in the conceptual schema. For example, when the value of an identity column is returned by an **insert** stored procedure, the **ResultBinding** element maps the returned value to an entity type property in the conceptual schema.

The following is the XML schema definition of the **ResultBinding** element.

```
<xs:complexType name="TResultBinding">
  <xs:attribute name="ColumnName" type="xs:string" use="required"/>
  <xs:attribute name="Name" type="csmsl:TSimpleIdentifier" use="required"/>
</xs:complexType>
```

The following additional rules apply to the **ResultBinding** element:

- **?Name** is the name of the entity property in the conceptual schema that is being mapped.

- **?ColumnName** is the name of the column that is being mapped.

2.1.18   AssociationEnd

The **AssociationEnd** element is used when the modification functions of an entity type that is participating in an association are mapped to the **store** function. This element is valid only when the participating entity type is on the "many" side of a one-to-many or zero-one to many relationship.

The following is the XML schema definition of the **AssociationEnd** element.

```
<xs:complexType name="TFunctionMappingAssociationEnd">
  <xs:group ref="csmsl:TFunctionMappingAssociationEndPropertyGroup" minOccurs="1"
maxOccurs="1"/>
  <xs:attribute name="AssociationSet" type="csmsl:TSimpleIdentifier" use="required"/>
  <xs:attribute name="From" type="csmsl:TSimpleIdentifier" use="required"/>
  <xs:attribute name="To" type="csmsl:TSimpleIdentifier" use="required"/>
</xs:complexType>
<xs:group name="TFunctionMappingAssociationEndPropertyGroup">
  <xs:sequence>
    <xs:element minOccurs="0" maxOccurs="unbounded" name="ScalarProperty"
type="csmsl:TFunctionMappingScalarProperty"/>
  </xs:sequence>
</xs:group>
```

The following additional rules apply to the **AssociationEnd** element:

- **AssociationSet** is the name of the **AssociationSet** element that is participating in the mapping.

- "From" specifies one end of the association.

- "To" specifies the other end of the association.

- The **AssociationEnd** element MUST have at least one or more **ScalarProperty** child elements.

2.1.19   ModificationFunctionMapping for AssociationSetMapping

The **ModificationFunctionMapping** element maps the **insert** and **delete** functions of a conceptual schema association type to stored procedures in the underlying database.

The following is the XML schema definition of the **ModificationFunctionMapping** element.

```
<xs:complexType name="TAssociationSetModificationFunctionMapping">
  <xs:all>
    <xs:element minOccurs="0" maxOccurs="1" name="DeleteFunction"
type="csmsl:TAssociationSetModificationFunction"/>
    <xs:element minOccurs="0" maxOccurs="1" name="InsertFunction"
type="csmsl:TAssociationSetModificationFunction"/>
  </xs:all>
</xs:complexType>
```

The following additional rules apply to the **ModificationFunctionMapping** element:

▪ The following child elements MAY appear in any given order under the **ModificationFunctionMapping** element.

  ▪**DeleteFunction**

  ▪**InsertFunction**

Note   In MSL version 1.0, if one of the two child elements is defined, both child elements MUST be defined. This restriction does not exist in MSL version 2.0.

2.1.20   DeleteFunction for AssociationType

The **DeleteFunction** element maps the **delete** function of an association in the conceptual schema to a stored procedure in the underlying database.

The following is the XML schema definition of the **DeleteFunction** element.

```
<xs:complexType name="TAssociationSetModificationFunction">
  <xs:group ref="csmsl:TAssociationSetFunctionMappingPropertyGroup" minOccurs="1"
maxOccurs="1"/>
  <xs:attribute name="FunctionName" type="xs:string" use="required"/>
  <xs:attribute name="RowsAffectedParameter" type="xs:string" use="optional"/>
</xs:complexType>
<xs:group name="TAssociationSetFunctionMappingPropertyGroup">
  <xs:sequence>
    <xs:choice maxOccurs="unbounded">
      <xs:element minOccurs="1" name="EndProperty"
type="csmsl:TFunctionMappingEndProperty"/>
    </xs:choice>
  </xs:sequence>
</xs:group>
```

The following additional rules apply to the **DeleteFunction** element:

▪ **FunctionName** is the fully qualified name of the stored procedure to which the **delete** function is mapped. The stored procedure MUST be declared in the store schema.

- *RowsAffectedParameter* is the name of the output parameter that returns the number of rows that are affected. This output parameter is of type **string**.

- The **DeleteFunction** element MUST specify exactly two **EndProperty** child elements to map the two ends of the association.

2.1.21 InsertFunction for AssociationType

The **InsertFunction** element maps the **insert** function of an association in the conceptual schema to a stored procedure in the underlying database.

The following is the XML schema definition of the **InsertFunction** element.

```
<xs:complexType name="TAssociationSetModificationFunction">
  <xs:group ref="csmsl:TAssociationSetFunctionMappingPropertyGroup" minOccurs="1"
 maxOccurs="1"/>
  <xs:attribute name="FunctionName" type="xs:string" use="required"/>
  <xs:attribute name="RowsAffectedParameter" type="xs:string" use="optional"/>
</xs:complexType>
<xs:group name="TAssociationSetFunctionMappingPropertyGroup">
  <xs:sequence>
    <xs:choice maxOccurs="unbounded">
      <xs:element minOccurs="1" name="EndProperty"
 type="csmsl:TFunctionMappingEndProperty"/>
    </xs:choice>
  </xs:sequence>
</xs:group>
```

The following additional rules apply to the **InsertFunction** element.

- **FunctionName** is the fully qualified name of the stored procedure to which the **insert** function is mapped. The stored procedure MUST be declared in the store schema.

- *RowsAffectedParameter* is the name of the output parameter that returns the number of rows that are affected.

- The **InsertFunction** element MUST specify exactly two **EndProperty** child elements to map the two ends of the association.

2.1.22 Condition

The **Condition** element places conditions on mappings between the conceptual schema and the store schema.

The following is the XML schema definition of the **Condition** element.

```
<xs:complexType name="TCondition">
  <xs:attribute name="Value" type="xs:string" use="optional" />
  <xs:attribute name="Name" type="csmsl:TSimpleIdentifier" use="optional" />
  <xs:attribute name="ColumnName" type="xs:string" use="optional" />
  <xs:attribute name="IsNull" type="xs:boolean" use="optional" />
</xs:complexType>
```

The following additional rules apply to the **Condition** element:

- The **Condition** element MUST have exactly one of the following attributes from each of the following two pairs of attributes specified.

  ▪**Name** or **ColumnName**

  ▪**Value** or **IsNull**

- The **Value** attribute, which is of type **string**, MUST be used only with the **ColumnName** attribute.

- The only possible values for the **IsNull** attribute are true and false.

2.1.23 EndProperty

The **EndProperty** element specifies the mapping between an association end or a modification function of a conceptual schema association and the underlying database. The property-column mapping is specified in a **ScalarProperty** child element.

When an **EndProperty** element is used to specify the mapping for the end of a conceptual schema association, it is a child element of an **AssociationSetMapping** element. When the **EndProperty** element is used to specify the mapping for a modification function of a conceptual schema association, it is a child element of an **InsertFunction** element or **DeleteFunction** element.

The following is the XML schema definition of the **EndProperty** element.

```
<xs:complexType name="TEndProperty">
  <xs:sequence>
    <xs:element maxOccurs="unbounded" name="ScalarProperty" type="csmsl:TScalarProperty"/>
  </xs:sequence>
  <xs:attribute name="Name" type="csmsl:TSimpleIdentifier" use="required" />
</xs:complexType>
```

The following additional rules apply to the **EndProperty** element:

- The **Name** attribute specifies the name of the association end that is being mapped.

2.1.24 ResultMapping

The **ResultMapping** element specifies the mapping between a function import in the conceptual schema and a stored procedure in the underlying database when the function import returns a conceptual schema entity type or complex type and the names of the columns returned by the stored procedure do not exactly match the names of the properties on the entity type or complex type.

The following is the XML schema definition of the **RappingMapping** element.

```
<xs:complexType name="TFunctionImportMappingResultMapping">
  <xs:choice>
    <xs:element minOccurs="0" maxOccurs="unbounded" name="EntityTypeMapping"
type="csmsl:TFunctionImportEntityTypeMapping"/>
    <xs:element minOccurs="0" maxOccurs="unbounded" name="ComplexTypeMapping"
type="csmsl:TFunctionImportComplexTypeMapping"/>
  </xs:choice>
</xs:complexType>
```

The following additional rules apply to the **ResultMapping** element:

▪ The **ResultMapping** element MUST have only one kind of the following child elements.

  ▪**EntityTypeMapping**

  ▪**ComplexTypeMapping**

Note   The **ResultMapping** element MAY have multiple **EntityTypeMapping** child elements, because the **FunctionImport** element MAY return polymorphic results.

2.1.25   ComplexTypeMapping for ResultMapping

The **ComplexTypeMapping** element is a child element of the **ResultMapping** element. The **ComplexTypeMapping** element specifies the mapping between a function import in the conceptual schema and a stored procedure in the underlying database, when the function import returns a conceptual complex type and the names of the columns that are returned by the stored procedure do not exactly match the names of the properties that are on the complex type.

By default, the mapping between the columns that are returned by a stored procedure and a complex type is based on column and property names. If column names do not exactly match property names, the user MUST use the **ComplexTypeMapping** element to define the mapping.

The following is the XML schema definition of the **ComplexTypeMapping** element.

```
<xs:complexType name="TFunctionImportComplexTypeMapping">
  <xs:sequence>
    <xs:element minOccurs="0" maxOccurs="unbounded" name="ScalarProperty"
type="csmsl:TScalarProperty" />
  </xs:sequence>
  <xs:attribute name="TypeName" type="xs:string" use="required" />
</xs:complexType>
```

The following additional rules apply to the **ComplexTypeMapping** element when it is a child element of the **ResultMapping** element:

▪ The **TypeName** attribute specifies the fully qualified complex type that is being mapped.

2.1.26   EntityTypeMapping for ResultMapping in FunctionImportMapping

When used for entity type mapping in the function result mapping under the **ResultMapping** element, the **EntityTypeMapping** element has rules that differ slightly from the general purpose rules. This section covers those rules. The general-purpose rules for the **EntityTypeMapping** element are specified in section 2.1.4.

The following is the XML schema definition of the **EntityTypeMapping** element.

```
<xs:complexType name="TFunctionImportEntityTypeMapping">
  <xs:choice maxOccurs="unbounded">
    <xs:element minOccurs="0" maxOccurs="unbounded" name="ScalarProperty"
type="csmsl:TScalarProperty" />
    <xs:element minOccurs="0" maxOccurs="unbounded" name="Condition"
type="csmsl:TFunctionImportCondition" />
  </xs:choice>
  <xs:attribute name="TypeName" type="xs:string" use="required" />
```

```
    </xs:complexType>
```

The following additional rules apply to the **EntityTypeMapping** element:

- The **TypeName** attribute specifies the entity type that is being mapped.

- Each **ScalarProperty** child element specifies the mapping between the entity property and the store column.

2.1.27   Condition for FunctionImportMapping

When used for entity type mapping or complex type mapping in the function result mapping under the **FunctionImportMapping** element, the **Condition** element has rules that differ slightly from the rules for the **Condition** element that are specified in section 2.1.19. This section covers those different rules.

The following is the XML schema definition of the **Condition** element.

```
    <xs:complexType name="TFunctionImportCondition">
      <xs:attribute name="Value" type="xs:string" use="optional" />
      <xs:attribute name="ColumnName" type="xs:string" use="required" />
      <xs:attribute name="IsNull" type="xs:boolean" use="optional" />
    </xs:complexType>
```

The following additional rules apply to the **Condition** element:

- The **Condition** element MUST have exactly one of the two attributes defined.

  ▪**Value**

  ▪**IsNull**

- For the **IsNull** attribute, the only possible values are true and false.

2.1.28   QueryView

The **QueryView** element specifies a mapping between an entity type or association in the conceptual schema and a table in the underlying database, but it does not generate update views for the mapped entity type. The mapping is specified by using an Entity SQL query that is evaluated against the store schema. The resulting type of the Entity SQL query MUST be assignable to the entity type.

The following is the XML schema definition of the **QueryView** element.

```
    <xs:complexType name="TQueryView">
      <xs:simpleContent>
        <xs:extension base="xs:string">
          <xs:attribute name="TypeName" type="xs:string" use="optional" />
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>
```

The following additional rules apply to the **QueryView** element:

- The **TypeName** attribute specifies the name of the conceptual schema type that is being mapped by the query view.

- The **QueryView** element MUST have a body that is of type **string**.

2.2 Attributes

2.2.1 EDMSimpleType

The **EDMSimpleType** attribute is a primitive type (as opposed to a structural type) that is used along with **ComplexType** as a building block for creating one or more structural type definitions. An **EDMSimpleType** attribute can be referred to by name or by a **namespace qualified name** where the namespace is "EDM".QualifiedName.

2.2.2 QualifiedName

The **QualifiedName** attribute is a string-based representation of the name of an element or attribute.

The following is the XML schema definition of the **QualifiedName** attribute.

```
<xs:simpleType name="TQualifiedName">
  <xs:restriction base="xs:token">
      <!-- The below pattern represents the allowed identifiers in ECMA specification plus
the '.' for namespace qualification  -->
       <xs:pattern
value="[\p{L}\p{Nl}][\p{L}\p{Nl}\p{Nd}\p{Mn}\p{Mc}\p{Pc}\p{Cf}]{0,}(\.[\p{L}\p{Nl}][\p{L}\p{N
l}\p{Nd}\p{Mn}\p{Mc}\p{Pc}\p{Cf}]{0,}){0,}"/>
  </xs:restriction>
</xs:simpleType>
```

2.2.3 SimpleIdentifier

The **SimpleIdentifier** attribute specifies a string-based representation of the name of an element or attribute. The maximum length of the **identifier** MUST be less than 480 characters.

The following is the XML schema definition of the **SimpleIdentifier** attribute.

```
<xs:simpleType name="TSimpleIdentifier">
  <xs:restriction base="xs:token">
      <!-- The below pattern represents the allowed identifiers in ECMA specification -->
      <xs:pattern value="[\p{L}\p{Nl}][\p{L}\p{Nl}\p{Nd}\p{Mn}\p{Mc}\p{Pc}\p{Cf}]{0,}"/>
  </xs:restriction>
</xs:simpleType>
```

3   Structure Examples

3.1   Mapping

The following is an example of the **Mapping** element.

```xml
<Mapping Space="C-S"
        xmlns="http://schemas.microsoft.com/ado/2008/09/mapping/cs">
  <Alias Key="c" Value="SchoolModel"/>
  <EntityContainerMapping StorageEntityContainer="SchoolModelStoreContainer"
                          CdmEntityContainer="SchoolModelEntities">
    <EntitySetMapping Name="Courses">
      <EntityTypeMapping TypeName="c.Course">
        <MappingFragment StoreEntitySet="Course">
          <ScalarProperty Name="CourseID" ColumnName="CourseID" />
          <ScalarProperty Name="Title" ColumnName="Title" />
          <ScalarProperty Name="Credits" ColumnName="Credits" />
          <ScalarProperty Name="DepartmentID" ColumnName="DepartmentID" />
        </MappingFragment>
      </EntityTypeMapping>
    </EntitySetMapping>
    <EntitySetMapping Name="Departments">
      <EntityTypeMapping TypeName="c.Department">
        <MappingFragment StoreEntitySet="Department">
          <ScalarProperty Name="DepartmentID" ColumnName="DepartmentID" />
          <ScalarProperty Name="Name" ColumnName="Name" />
          <ScalarProperty Name="Budget" ColumnName="Budget" />
          <ScalarProperty Name="StartDate" ColumnName="StartDate" />
          <ScalarProperty Name="Administrator" ColumnName="Administrator" />
        </MappingFragment>
      </EntityTypeMapping>
    </EntitySetMapping>
  </EntityContainerMapping>
</Mapping>
```

The following is an example of a **Mapping** element that is specified by using MSL.

```xml
<?xml version="1.0"?>
<Mapping xmlns="http://schemas.microsoft.com/ado/2008/09/mapping/cs " Space="C-S"
xmlns:d1p1="http://schemas.microsoft.com/ado/2008/09/mapping/cs">
  <EntityContainerMapping CdmEntityContainer="CNorthwind_Container"
StorageEntityContainer="Northwind1_dbo" xmlns:cdm="urn:schemas-microsoft-
com:windows:storage:mapping:CS">
    <EntitySetMapping Name="Products">
      <EntityTypeMapping TypeName="CNorthwind.ProductsType">
        <MappingFragment StoreEntitySet="Products">
          <ScalarProperty Name="ProductID" ColumnName="ProductID" />
          <ScalarProperty Name="Discontinued" ColumnName="Discontinued" />
          <ScalarProperty Name="ProductName" ColumnName="ProductName" />
          <ScalarProperty Name="QuantityPerUnit" ColumnName="QuantityPerUnit" />
          <ScalarProperty Name="ReorderLevel" ColumnName="ReorderLevel" />
          <ScalarProperty Name="UnitPrice" ColumnName="UnitPrice" />
          <ScalarProperty Name="UnitsInStock" ColumnName="UnitsInStock" />
          <ScalarProperty Name="UnitsOnOrder" ColumnName="UnitsOnOrder" />
        </MappingFragment>
      </EntityTypeMapping>
```

```
        </EntitySetMapping>
        <EntitySetMapping Name="Suppliers">
          <EntityTypeMapping TypeName="CNorthwind.SuppliersType">
            <MappingFragment StoreEntitySet="Suppliers">
              <ScalarProperty Name="SupplierID" ColumnName="SupplierID" />
              <ScalarProperty Name="Address" ColumnName="Address" />
              <ScalarProperty Name="City" ColumnName="City" />
              <ScalarProperty Name="CompanyName" ColumnName="CompanyName" />
              <ScalarProperty Name="ContactName" ColumnName="ContactName" />
              <ScalarProperty Name="ContactTitle" ColumnName="ContactTitle" />
              <ScalarProperty Name="Country" ColumnName="Country" />
              <ScalarProperty Name="Fax" ColumnName="Fax" />
              <ScalarProperty Name="HomePage" ColumnName="HomePage" />
              <ScalarProperty Name="Phone" ColumnName="Phone" />
              <ScalarProperty Name="PostalCode" ColumnName="PostalCode" />
              <ScalarProperty Name="Region" ColumnName="Region" />
            </MappingFragment>
          </EntityTypeMapping>
        </EntitySetMapping>
        <EntitySetMapping Name="Categories">
          <EntityTypeMapping TypeName="CNorthwind.CategoriesType">
            <MappingFragment StoreEntitySet="Categories">
              <ScalarProperty Name="CategoryID" ColumnName="CategoryID" />
              <ScalarProperty Name="CategoryName" ColumnName="CategoryName" />
              <ScalarProperty Name="Description" ColumnName="Description" />
            </MappingFragment>
          </EntityTypeMapping>
        </EntitySetMapping>
        <AssociationSetMapping Name="ProductsSuppliers"
  TypeName="CNorthwind.ProductsSuppliersType" StoreEntitySet="Products">
          <EndProperty Name="SuppliersType">
            <ScalarProperty Name="SupplierID" ColumnName="SupplierID" />
          </EndProperty>
          <EndProperty Name="ProductsType">
            <ScalarProperty Name="ProductID" ColumnName="ProductID" />
          </EndProperty>
        </AssociationSetMapping>
        <AssociationSetMapping Name="ProductsCategories"
  TypeName="CNorthwind.ProductsCategoriesType" StoreEntitySet="Products">
          <EndProperty Name="CategoriesType">
            <ScalarProperty Name="CategoryID" ColumnName="CategoryID" />
          </EndProperty>
          <EndProperty Name="ProductsType">
            <ScalarProperty Name="ProductID" ColumnName="ProductID" />
          </EndProperty>
        </AssociationSetMapping>
      </EntityContainerMapping>
    </Mapping>
```

3.2   EntityContainerMapping

The following is an example of the **EntityContainerMapping** element.

```
<EntityContainerMapping StorageEntityContainer="SchoolModelStoreContainer"
                        CdmEntityContainer="SchoolModelEntities">
  <EntitySetMapping Name="Courses">
    <EntityTypeMapping TypeName="c.Course">
```

```
            <MappingFragment StoreEntitySet="Course">
              <ScalarProperty Name="CourseID" ColumnName="CourseID" />
              <ScalarProperty Name="Title" ColumnName="Title" />
              <ScalarProperty Name="Credits" ColumnName="Credits" />
              <ScalarProperty Name="DepartmentID" ColumnName="DepartmentID" />
            </MappingFragment>
          </EntityTypeMapping>
        </EntitySetMapping>
        <EntitySetMapping Name="Departments">
          <EntityTypeMapping TypeName="c.Department">
            <MappingFragment StoreEntitySet="Department">
              <ScalarProperty Name="DepartmentID" ColumnName="DepartmentID" />
              <ScalarProperty Name="Name" ColumnName="Name" />
              <ScalarProperty Name="Budget" ColumnName="Budget" />
              <ScalarProperty Name="StartDate" ColumnName="StartDate" />
              <ScalarProperty Name="Administrator" ColumnName="Administrator" />
            </MappingFragment>
          </EntityTypeMapping>
        </EntitySetMapping>
      </EntityContainerMapping>
```

3.3 EntitySetMapping

The following is an example of the **EntitySetMapping** element.

```
<EntitySetMapping Name="Courses">
  <EntityTypeMapping TypeName="IsTypeOf(SchoolModel1.Course)">
    <MappingFragment StoreEntitySet="Course">
      <ScalarProperty Name="CourseID" ColumnName="CourseID" />
      <ScalarProperty Name="DepartmentID" ColumnName="DepartmentID" />
      <ScalarProperty Name="Credits" ColumnName="Credits" />
      <ScalarProperty Name="Title" ColumnName="Title" />
    </MappingFragment>
  </EntityTypeMapping>
  <EntityTypeMapping TypeName="IsTypeOf(SchoolModel1.OnlineCourse)">
    <MappingFragment StoreEntitySet="OnlineCourse">
      <ScalarProperty Name="CourseID" ColumnName="CourseID" />
      <ScalarProperty Name="URL" ColumnName="URL" />
    </MappingFragment>
  </EntityTypeMapping>
  <EntityTypeMapping TypeName="IsTypeOf(SchoolModel1.OnsiteCourse)">
    <MappingFragment StoreEntitySet="OnsiteCourse">
      <ScalarProperty Name="CourseID" ColumnName="CourseID" />
      <ScalarProperty Name="Time" ColumnName="Time" />
      <ScalarProperty Name="Days" ColumnName="Days" />
      <ScalarProperty Name="Location" ColumnName="Location" />
    </MappingFragment>
  </EntityTypeMapping>
</EntitySetMapping>
```

3.4 EntityTypeMapping

The following is an example of the **EntityTypeMapping** element.

```
<EntitySetMapping Name="People">
  <EntityTypeMapping TypeName="SchoolModel.Person">
    <MappingFragment StoreEntitySet="Person">
      <ScalarProperty Name="PersonID" ColumnName="PersonID" />
      <ScalarProperty Name="LastName" ColumnName="LastName" />
      <ScalarProperty Name="FirstName" ColumnName="FirstName" />
      <ScalarProperty Name="HireDate" ColumnName="HireDate" />
      <ScalarProperty Name="EnrollmentDate" ColumnName="EnrollmentDate" />
    </MappingFragment>
  </EntityTypeMapping>
  <EntityTypeMapping TypeName="SchoolModel.Person">
    <ModificationFunctionMapping>
      <UpdateFunction FunctionName="SchoolModel.Store.UpdatePerson">
        <ScalarProperty Name="EnrollmentDate" ParameterName="EnrollmentDate"
                        Version="Current" />
        <ScalarProperty Name="HireDate" ParameterName="HireDate"
                        Version="Current" />
        <ScalarProperty Name="FirstName" ParameterName="FirstName"
                        Version="Current" />
        <ScalarProperty Name="LastName" ParameterName="LastName"
                        Version="Current" />
        <ScalarProperty Name="PersonID" ParameterName="PersonID"
                        Version="Current" />
      </UpdateFunction>
    </ModificationFunctionMapping>
  </EntityTypeMapping>
</EntitySetMapping>
```

3.5   MappingFragment

The following is an example of the **MappingFragment** element.

```
<EntitySetMapping Name="Courses">
  <EntityTypeMapping TypeName="SchoolModel.Course">
    <MappingFragment StoreEntitySet="Course">
      <ScalarProperty Name="CourseID" ColumnName="CourseID" />
      <ScalarProperty Name="Title" ColumnName="Title" />
      <ScalarProperty Name="Credits" ColumnName="Credits" />
      <ScalarProperty Name="DepartmentID" ColumnName="DepartmentID" />
    </MappingFragment>
  </EntityTypeMapping>
</EntitySetMapping>
```

3.6   ComplexProperty

The following is an example of the **ComplexProperty** element.

```
<EntitySetMapping Name="CCustomer1" TypeName="CNorthwind.CCustomer"
StoreEntitySet="SCustomers1">
  <ScalarProperty Name="CustomerId" ColumnName="CustomerId" />
  <ScalarProperty Name="CompanyName" ColumnName="CompanyName" />
  <ScalarProperty Name="ContactName" ColumnName="ContactName" />
  <ScalarProperty Name="ContactTitle" ColumnName="ContactTitle" />
  <ComplexProperty Name="Address" TypeName="CNorthwind.CAddress">
```

```
        <ScalarProperty Name="StreetAddress" ColumnName="Address" />
        <ScalarProperty Name="City" ColumnName="City" />
        <ScalarProperty Name="Region" ColumnName="Region" />
        <ScalarProperty Name="PostalCode" ColumnName="PostalCode" />
      </ComplexProperty>
    </EntitySetMapping>
```

3.7 ComplexTypeMapping

The following is an example of the **ComplexTypeMapping** element.

```
        <ComplexProperty Name="Name" IsPartial="true">
          <ComplexTypeMapping TypeName="IsTypeOf(CNorthwind.CFullName)">
            <ScalarProperty Name="LastName" ColumnName="LastName" />
            <ScalarProperty Name="FirstName" ColumnName="FirstName" />
          </ComplexTypeMapping>
        </ComplexProperty>
```

3.8 ScalarProperty

The following is an example of the **ScalarProperty** element.

```
  <EntitySetMapping Name="People">
    <EntityTypeMapping TypeName="SchoolModel.Person">
      <MappingFragment StoreEntitySet="Person">
        <ScalarProperty Name="PersonID" ColumnName="PersonID" />
        <ScalarProperty Name="LastName" ColumnName="LastName" />
        <ScalarProperty Name="FirstName" ColumnName="FirstName" />
        <ScalarProperty Name="HireDate" ColumnName="HireDate" />
        <ScalarProperty Name="EnrollmentDate"
                        ColumnName="EnrollmentDate" />
      </MappingFragment>
    </EntityTypeMapping>
  </EntitySetMapping>
```

3.9 AssociationSetMapping

The following is an example of the **AssociationSetMapping** element.

```
  <AssociationSetMapping Name="FK_Course_Department"
                         TypeName="SchoolModel.FK_Course_Department"
                         StoreEntitySet="Course">
    <EndProperty Name="Department">
      <ScalarProperty Name="DepartmentID" ColumnName="DepartmentID" />
    </EndProperty>
    <EndProperty Name="Course">
      <ScalarProperty Name="CourseID" ColumnName="CourseID" />
    </EndProperty>
  </AssociationSetMapping>
```

3.10 FunctionImportMapping

The following is an example of the **FunctionImportMapping** element.

```
<FunctionImportMapping FunctionImportName="GetGrades"
                       FunctionName="SchoolModel.Store.GetGrades" >
  <ResultMapping>
    <EntityTypeMapping TypeName="SchoolModel.StudentGrade">
      <ScalarProperty Name="EnrollmentID" ColumnName="enroll_id"/>
      <ScalarProperty Name="CourseID" ColumnName="course_id"/>
      <ScalarProperty Name="StudentID" ColumnName="student_id"/>
      <ScalarProperty Name="Grade" ColumnName="grade"/>
    </EntityTypeMapping>
  </ResultMapping>
</FunctionImportMapping>
```

3.11 ModificationFunctionMapping for Entity Type

The following is an example of the **ModificationFunctionMapping** element.

```
<EntitySetMapping Name="People">
  <EntityTypeMapping TypeName="SchoolModel.Person">
    <MappingFragment StoreEntitySet="Person">
      <ScalarProperty Name="PersonID" ColumnName="PersonID" />
      <ScalarProperty Name="LastName" ColumnName="LastName" />
      <ScalarProperty Name="FirstName" ColumnName="FirstName" />
      <ScalarProperty Name="HireDate" ColumnName="HireDate" />
      <ScalarProperty Name="EnrollmentDate"
                      ColumnName="EnrollmentDate" />
    </MappingFragment>
</EntityTypeMapping>
  <EntityTypeMapping TypeName="SchoolModel.Person">
    <ModificationFunctionMapping>
      <InsertFunction FunctionName="SchoolModel.Store.InsertPerson">
        <ScalarProperty Name="EnrollmentDate"
                        ParameterName="EnrollmentDate" />
        <ScalarProperty Name="HireDate" ParameterName="HireDate" />
        <ScalarProperty Name="FirstName" ParameterName="FirstName" />
        <ScalarProperty Name="LastName" ParameterName="LastName" />
        <ResultBinding Name="PersonID" ColumnName="NewPersonID" />
      </InsertFunction>
      <UpdateFunction FunctionName="SchoolModel.Store.UpdatePerson">
        <ScalarProperty Name="EnrollmentDate"
                        ParameterName="EnrollmentDate"
                        Version="Current" />
        <ScalarProperty Name="HireDate" ParameterName="HireDate"
                        Version="Current" />
        <ScalarProperty Name="FirstName" ParameterName="FirstName"
                        Version="Current" />
        <ScalarProperty Name="LastName" ParameterName="LastName"
                        Version="Current" />
        <ScalarProperty Name="PersonID" ParameterName="PersonID"
                        Version="Current" />
      </UpdateFunction>
      <DeleteFunction FunctionName="SchoolModel.Store.DeletePerson">
        <ScalarProperty Name="PersonID" ParameterName="PersonID" />
```

```
      </DeleteFunction>
    </ModificationFunctionMapping>
  </EntityTypeMapping>
</EntitySetMapping>
```

3.12   DeleteFunction for Entity Type

The following is an example of the **DeleteFunction** element.

```
<EntitySetMapping Name="People">
  <EntityTypeMapping TypeName="SchoolModel.Person">
    <MappingFragment StoreEntitySet="Person">
      <ScalarProperty Name="PersonID" ColumnName="PersonID" />
      <ScalarProperty Name="LastName" ColumnName="LastName" />
      <ScalarProperty Name="FirstName" ColumnName="FirstName" />
      <ScalarProperty Name="HireDate" ColumnName="HireDate" />
      <ScalarProperty Name="EnrollmentDate"
                      ColumnName="EnrollmentDate" />
    </MappingFragment>
</EntityTypeMapping>
  <EntityTypeMapping TypeName="SchoolModel.Person">
    <ModificationFunctionMapping>
      <InsertFunction FunctionName="SchoolModel.Store.InsertPerson">
        <ScalarProperty Name="EnrollmentDate"
                        ParameterName="EnrollmentDate" />
        <ScalarProperty Name="HireDate" ParameterName="HireDate" />
        <ScalarProperty Name="FirstName" ParameterName="FirstName" />
        <ScalarProperty Name="LastName" ParameterName="LastName" />
        <ResultBinding Name="PersonID" ColumnName="NewPersonID" />
      </InsertFunction>
      <UpdateFunction FunctionName="SchoolModel.Store.UpdatePerson">
        <ScalarProperty Name="EnrollmentDate"
                        ParameterName="EnrollmentDate"
                        Version="Current" />
        <ScalarProperty Name="HireDate" ParameterName="HireDate"
                        Version="Current" />
        <ScalarProperty Name="FirstName" ParameterName="FirstName"
                        Version="Current" />
        <ScalarProperty Name="LastName" ParameterName="LastName"
                        Version="Current" />
        <ScalarProperty Name="PersonID" ParameterName="PersonID"
                        Version="Current" />
      </UpdateFunction>
      <DeleteFunction FunctionName="SchoolModel.Store.DeletePerson">
        <ScalarProperty Name="PersonID" ParameterName="PersonID" />
      </DeleteFunction>
    </ModificationFunctionMapping>
  </EntityTypeMapping>
</EntitySetMapping>
```

3.13   InsertFunction for Entity Type

The following is an example of the **InsertFunction** element.

```
<EntityTypeMapping TypeName="SchoolModel.Person">
  <ModificationFunctionMapping>
    <InsertFunction FunctionName="SchoolModel.Store.InsertPerson">
      <ScalarProperty Name="EnrollmentDate"
                      ParameterName="EnrollmentDate" />
      <ScalarProperty Name="HireDate" ParameterName="HireDate" />
      <ScalarProperty Name="FirstName" ParameterName="FirstName" />
      <ScalarProperty Name="LastName" ParameterName="LastName" />
      <ResultBinding Name="PersonID" ColumnName="NewPersonID" />
    </InsertFunction>
    <UpdateFunction FunctionName="SchoolModel.Store.UpdatePerson">
      <ScalarProperty Name="EnrollmentDate"
                      ParameterName="EnrollmentDate"
                      Version="Current" />
      <ScalarProperty Name="HireDate" ParameterName="HireDate"
                      Version="Current" />
      <ScalarProperty Name="FirstName" ParameterName="FirstName"
                      Version="Current" />
      <ScalarProperty Name="LastName" ParameterName="LastName"
                      Version="Current" />
      <ScalarProperty Name="PersonID" ParameterName="PersonID"
                      Version="Current" />
    </UpdateFunction>
    <DeleteFunction FunctionName="SchoolModel.Store.DeletePerson">
      <ScalarProperty Name="PersonID" ParameterName="PersonID" />
    </DeleteFunction>
  </ModificationFunctionMapping>
</EntityTypeMapping>
```

3.14 UpdateFunction for Entity Type

The following is an example of the **UpdateFunction** element.

```
<EntityTypeMapping TypeName="SchoolModel.Person">
  <ModificationFunctionMapping>
    <InsertFunction FunctionName="SchoolModel.Store.InsertPerson">
      <ScalarProperty Name="EnrollmentDate"
                      ParameterName="EnrollmentDate" />
      <ScalarProperty Name="HireDate" ParameterName="HireDate" />
      <ScalarProperty Name="FirstName" ParameterName="FirstName" />
      <ScalarProperty Name="LastName" ParameterName="LastName" />
      <ResultBinding Name="PersonID" ColumnName="NewPersonID" />
    </InsertFunction>
    <UpdateFunction FunctionName="SchoolModel.Store.UpdatePerson">
      <ScalarProperty Name="EnrollmentDate"
                      ParameterName="EnrollmentDate"
                      Version="Current" />
      <ScalarProperty Name="HireDate" ParameterName="HireDate"
                      Version="Current" />
      <ScalarProperty Name="FirstName" ParameterName="FirstName"
                      Version="Current" />
      <ScalarProperty Name="LastName" ParameterName="LastName"
                      Version="Current" />
      <ScalarProperty Name="PersonID" ParameterName="PersonID"
                      Version="Current" />
    </UpdateFunction>
    <DeleteFunction FunctionName="SchoolModel.Store.DeletePerson">
```

```
            <ScalarProperty Name="PersonID" ParameterName="PersonID" />
      </DeleteFunction>
   </ModificationFunctionMapping>
</EntityTypeMapping>
```

3.15   ScalarProperty for ModificationFunctionMapping

The following is an example of the **ScalarProperty** element.

```
<EntitySetMapping Name="People">
  <EntityTypeMapping TypeName="SchoolModel.Person">
    <MappingFragment StoreEntitySet="Person">
      <ScalarProperty Name="PersonID" ColumnName="PersonID" />
      <ScalarProperty Name="LastName" ColumnName="LastName" />
      <ScalarProperty Name="FirstName" ColumnName="FirstName" />
      <ScalarProperty Name="HireDate" ColumnName="HireDate" />
      <ScalarProperty Name="EnrollmentDate"
                      ColumnName="EnrollmentDate" />
    </MappingFragment>
  </EntityTypeMapping>
  <EntityTypeMapping TypeName="SchoolModel.Person">
    <ModificationFunctionMapping>
      <InsertFunction FunctionName="SchoolModel.Store.InsertPerson">
        <ScalarProperty Name="EnrollmentDate"
                        ParameterName="EnrollmentDate" />
        <ScalarProperty Name="HireDate" ParameterName="HireDate" />
        <ScalarProperty Name="FirstName" ParameterName="FirstName" />
        <ScalarProperty Name="LastName" ParameterName="LastName" />
        <ResultBinding Name="PersonID" ColumnName="NewPersonID" />
      </InsertFunction>
      <UpdateFunction FunctionName="SchoolModel.Store.UpdatePerson">
        <ScalarProperty Name="EnrollmentDate"
                        ParameterName="EnrollmentDate"
                        Version="Current" />
        <ScalarProperty Name="HireDate" ParameterName="HireDate"
                        Version="Current" />
        <ScalarProperty Name="FirstName" ParameterName="FirstName"
                        Version="Current" />
        <ScalarProperty Name="LastName" ParameterName="LastName"
                        Version="Current" />
        <ScalarProperty Name="PersonID" ParameterName="PersonID"
                        Version="Current" />
      </UpdateFunction>
      <DeleteFunction FunctionName="SchoolModel.Store.DeletePerson">
        <ScalarProperty Name="PersonID" ParameterName="PersonID" />
      </DeleteFunction>
    </ModificationFunctionMapping>
  </EntityTypeMapping>
</EntitySetMapping>
```

3.16   ResultBinding

The following is an example of the **ResultBinding** element.

```
<EntityTypeMapping TypeName="SchoolModel.Person">
  <ModificationFunctionMapping>
    <InsertFunction FunctionName="SchoolModel.Store.InsertPerson">
      <ScalarProperty Name="EnrollmentDate"
                      ParameterName="EnrollmentDate" />
      <ScalarProperty Name="HireDate" ParameterName="HireDate" />
      <ScalarProperty Name="FirstName" ParameterName="FirstName" />
      <ScalarProperty Name="LastName" ParameterName="LastName" />
      <ResultBinding Name="PersonID" ColumnName="NewPersonID" />
    </InsertFunction>
    <UpdateFunction FunctionName="SchoolModel.Store.UpdatePerson">
      <ScalarProperty Name="EnrollmentDate"
                      ParameterName="EnrollmentDate"
                      Version="Current" />
      <ScalarProperty Name="HireDate" ParameterName="HireDate"
                      Version="Current" />
      <ScalarProperty Name="FirstName" ParameterName="FirstName"
                      Version="Current" />
      <ScalarProperty Name="LastName" ParameterName="LastName"
                      Version="Current" />
      <ScalarProperty Name="PersonID" ParameterName="PersonID"
                      Version="Current" />
    </UpdateFunction>
    <DeleteFunction FunctionName="SchoolModel.Store.DeletePerson">
      <ScalarProperty Name="PersonID" ParameterName="PersonID" />
    </DeleteFunction>
  </ModificationFunctionMapping>
</EntityTypeMapping>
```

3.17 AssociationEnd

The following is an example of the **AssociationEnd** element.

```
<EntitySetMapping Name="Courses">
  <EntityTypeMapping TypeName="SchoolModel.Course">
    <MappingFragment StoreEntitySet="Course">
      <ScalarProperty Name="Credits" ColumnName="Credits" />
      <ScalarProperty Name="Title" ColumnName="Title" />
      <ScalarProperty Name="CourseID" ColumnName="CourseID" />
    </MappingFragment>
  </EntityTypeMapping>
  <EntityTypeMapping TypeName="SchoolModel.Course">
    <ModificationFunctionMapping>
      <UpdateFunction FunctionName="SchoolModel.Store.UpdateCourse">
        <AssociationEnd AssociationSet="FK_Course_Department"
                        From="Course" To="Department">
          <ScalarProperty Name="DepartmentID"
                          ParameterName="DepartmentID"
                          Version="Current" />
        </AssociationEnd>
        <ScalarProperty Name="Credits" ParameterName="Credits"
                        Version="Current" />
        <ScalarProperty Name="Title" ParameterName="Title"
                        Version="Current" />
        <ScalarProperty Name="CourseID" ParameterName="CourseID"
                        Version="Current" />
```

```
            </UpdateFunction>
          </ModificationFunctionMapping>
        </EntityTypeMapping>
      </EntitySetMapping>
```

3.18   ModificationFunctionMapping for AssociationSetMapping

The following is an example of the **ModificationFunctionMapping** element.

```
    <AssociationSetMapping Name="CourseInstructor"
                           TypeName="SchoolModel.CourseInstructor"
                           StoreEntitySet="CourseInstructor">
      <EndProperty Name="Person">
        <ScalarProperty Name="PersonID" ColumnName="PersonID" />
      </EndProperty>
      <EndProperty Name="Course">
        <ScalarProperty Name="CourseID" ColumnName="CourseID" />
      </EndProperty>
      <ModificationFunctionMapping>
        <InsertFunction FunctionName="SchoolModel.Store.InsertCourseInstructor" >
          <EndProperty Name="Course">
            <ScalarProperty Name="CourseID" ParameterName="courseId"/>
          </EndProperty>
          <EndProperty Name="Person">
            <ScalarProperty Name="PersonID" ParameterName="instructorId"/>
          </EndProperty>
        </InsertFunction>
        <DeleteFunction FunctionName="SchoolModel.Store.DeleteCourseInstructor">
          <EndProperty Name="Course">
            <ScalarProperty Name="CourseID" ParameterName="courseId"/>
          </EndProperty>
          <EndProperty Name="Person">
            <ScalarProperty Name="PersonID" ParameterName="instructorId"/>
          </EndProperty>
        </DeleteFunction>
      </ModificationFunctionMapping>
    </AssociationSetMapping>
```

3.19   DeleteFunction for AssociationType

The following is an example of the **DeleteFunction** element.

```
    <AssociationSetMapping Name="CourseInstructor"
                           TypeName="SchoolModel.CourseInstructor"
                           StoreEntitySet="CourseInstructor">
      <EndProperty Name="Person">
        <ScalarProperty Name="PersonID" ColumnName="PersonID" />
      </EndProperty>
      <EndProperty Name="Course">
        <ScalarProperty Name="CourseID" ColumnName="CourseID" />
      </EndProperty>
      <ModificationFunctionMapping>
        <InsertFunction FunctionName="SchoolModel.Store.InsertCourseInstructor" >
          <EndProperty Name="Course">
```

```
          <ScalarProperty Name="CourseID" ParameterName="courseId"/>
        </EndProperty>
        <EndProperty Name="Person">
          <ScalarProperty Name="PersonID" ParameterName="instructorId"/>
        </EndProperty>
      </InsertFunction>
      <DeleteFunction FunctionName="SchoolModel.Store.DeleteCourseInstructor">
        <EndProperty Name="Course">
          <ScalarProperty Name="CourseID" ParameterName="courseId"/>
        </EndProperty>
        <EndProperty Name="Person">
          <ScalarProperty Name="PersonID" ParameterName="instructorId"/>
        </EndProperty>
      </DeleteFunction>
    </ModificationFunctionMapping>
  </AssociationSetMapping>
```

3.20 InsertFunction for AssociationType

The following is an example of the **InsertFunction** element.

```
  <AssociationSetMapping Name="CourseInstructor"
                         TypeName="SchoolModel.CourseInstructor"
                         StoreEntitySet="CourseInstructor">
    <EndProperty Name="Person">
      <ScalarProperty Name="PersonID" ColumnName="PersonID" />
    </EndProperty>
    <EndProperty Name="Course">
      <ScalarProperty Name="CourseID" ColumnName="CourseID" />
    </EndProperty>
    <ModificationFunctionMapping>
      <InsertFunction FunctionName="SchoolModel.Store.InsertCourseInstructor" >
        <EndProperty Name="Course">
          <ScalarProperty Name="CourseID" ParameterName="courseId"/>
        </EndProperty>
        <EndProperty Name="Person">
          <ScalarProperty Name="PersonID" ParameterName="instructorId"/>
        </EndProperty>
      </InsertFunction>
      <DeleteFunction FunctionName="SchoolModel.Store.DeleteCourseInstructor">
        <EndProperty Name="Course">
          <ScalarProperty Name="CourseID" ParameterName="courseId"/>
        </EndProperty>
        <EndProperty Name="Person">
          <ScalarProperty Name="PersonID" ParameterName="instructorId"/>
        </EndProperty>
      </DeleteFunction>
    </ModificationFunctionMapping>
  </AssociationSetMapping>
```

3.21 Condition

The following is an example of the **Condition** element.

```xml
<EntitySetMapping Name="People">
  <EntityTypeMapping TypeName="IsTypeOf(SchoolModel.Person)">
    <MappingFragment StoreEntitySet="Person">
      <ScalarProperty Name="PersonID" ColumnName="PersonID" />
      <ScalarProperty Name="FirstName" ColumnName="FirstName" />
      <ScalarProperty Name="LastName" ColumnName="LastName" />
    </MappingFragment>
  </EntityTypeMapping>
  <EntityTypeMapping TypeName="IsTypeOf(SchoolModel.Instructor)">
    <MappingFragment StoreEntitySet="Person">
      <ScalarProperty Name="PersonID" ColumnName="PersonID" />
      <ScalarProperty Name="HireDate" ColumnName="HireDate" />
      <Condition ColumnName="HireDate" IsNull="false" />
      <Condition ColumnName="EnrollmentDate" IsNull="true" />
    </MappingFragment>
  </EntityTypeMapping>
  <EntityTypeMapping TypeName="IsTypeOf(SchoolModel.Student)">
    <MappingFragment StoreEntitySet="Person">
      <ScalarProperty Name="PersonID" ColumnName="PersonID" />
      <ScalarProperty Name="EnrollmentDate"
                      ColumnName="EnrollmentDate" />
      <Condition ColumnName="EnrollmentDate" IsNull="false" />
      <Condition ColumnName="HireDate" IsNull="true" />
    </MappingFragment>
  </EntityTypeMapping>
</EntitySetMapping>
```

3.22   EndProperty

The following is an example of the **EndProperty** element.

```xml
<AssociationSetMapping Name="FK_Course_Department"
                       TypeName="SchoolModel.FK_Course_Department"
                       StoreEntitySet="Course">
  <EndProperty Name="Department">
    <ScalarProperty Name="DepartmentID" ColumnName="DepartmentID" />
  </EndProperty>
  <EndProperty Name="Course">
    <ScalarProperty Name="CourseID" ColumnName="CourseID" />
  </EndProperty>
</AssociationSetMapping>
```

3.23   ResultMapping

The following is an example of the **ResultMapping** element.

```xml
<FunctionImportMapping FunctionImportName="GetGrades"
                       FunctionName="SchoolModel.Store.GetGrades" >
  <ResultMapping>
    <EntityTypeMapping TypeName="SchoolModel.StudentGrade">
      <ScalarProperty Name="EnrollmentID" ColumnName="enroll_id"/>
      <ScalarProperty Name="CourseID" ColumnName="course_id"/>
      <ScalarProperty Name="StudentID" ColumnName="student_id"/>
      <ScalarProperty Name="Grade" ColumnName="grade"/>
    </EntityTypeMapping>
```

```
      </ResultMapping>
   </FunctionImportMapping>
```

3.24 ComplexTypeMapping for ResultMapping

The following is an example of the **ComplexTypeMapping** element.

```
<FunctionImportMapping FunctionImportName="GetGrades"
                       FunctionName="SchoolModel.Store.GetGrades" >
  <ResultMapping>
    <ComplexTypeMapping TypeName="SchoolModel.GradeInfo">
      <ScalarProperty Name="EnrollmentID" ColumnName="enroll_id"/>
      <ScalarProperty Name="CourseID" ColumnName="course_id"/>
      <ScalarProperty Name="StudentID" ColumnName="student_id"/>
      <ScalarProperty Name="Grade" ColumnName="grade"/>
    </ComplexTypeMapping>
  </ResultMapping>
</FunctionImportMapping>
```

3.25 EntityTypeMapping for ResultMapping

The following is an example of the **EntityTypeMapping** element.

```
<EntityContainerMapping StorageEntityContainer="StoreContainer"
CdmEntityContainer="CustomerEntityContainer">
  <FunctionImportMapping FunctionImportName="GetCustomerInfoById"
FunctionName="StoreNamespace.GetCustomerInfoById">
    <ResultMapping>
      <EntityTypeMapping TypeName="ModelNamespace.CustomerEntity">
        <ScalarProperty Name="f_name" ColumnName="first_name" />
        <ScalarProperty Name="l_name" ColumnName="last_name" />
        <ScalarProperty Name="address_city" ColumnName="city" />
      </EntityTypeMapping>
    </ResultMapping>
  </FunctionImportMapping>
</EntityContainerMapping>
```

3.26 Condition for FunctionImportMapping

The following is an example of the **Condition** element.

```
<FunctionImportMapping FunctionImportName="GetGrades"
                       FunctionName="SchoolModel.Store.GetGrades" >
  <ResultMapping>
    <EntityTypeMapping TypeName="SchoolModel.StudentGrade">
      <ScalarProperty Name="EnrollmentID" ColumnName="enroll_id"/>
      <ScalarProperty Name="CourseID" ColumnName="course_id"/>
      <ScalarProperty Name="StudentID" ColumnName="student_id"/>
      <ScalarProperty Name="Grade" ColumnName="grade"/>
      <Condition ColumnName="HireDate" IsNull="false" />
    </EntityTypeMapping>
  </ResultMapping>
</FunctionImportMapping>
```

3.27 QueryView

The following is an example of the **QueryView** element.

```xml
<EntityContainerMapping StorageEntityContainer="SchoolModelStoreContainer"
                        CdmEntityContainer="SchoolEntities">
  <EntitySetMapping Name="Courses" >
    <QueryView>
      SELECT VALUE SchoolModel.Course(c.CourseID,
                                      c.Title,
                                      c.Credits)
      FROM SchoolModelStoreContainer.Course AS c
    </QueryView>
  </EntitySetMapping>
  <EntitySetMapping Name="Departments" >
    <QueryView>
      SELECT VALUE SchoolModel.Department(d.DepartmentID,
                                          d.Name,
                                          d.Budget,
                                          d.StartDate)
      FROM SchoolModelStoreContainer.Department AS d
      WHERE d.Budget > 150000
    </QueryView>
  </EntitySetMapping>
  <AssociationSetMapping Name="FK_Course_Department" >
    <QueryView>
      SELECT VALUE SchoolModel.FK_Course_Department(
        CREATEREF(SchoolEntities.Departments, row(c.DepartmentID), SchoolModel.Department),
        CREATEREF(SchoolEntities.Courses, row(c.CourseID)) )
      FROM SchoolModelStoreContainer.Course AS c
    </QueryView>
  </AssociationSetMapping>
</EntityContainerMapping>
```

4   Security Considerations

None.

5   Appendix B: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include released service packs:

▪ Windows® 7 operating system

▪ Windows Server® 2008 R2 operating system

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

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

6   Change Tracking

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

Release: Wednesday, February 9, 2011

7 Index [MS-MSL]

Release: Wednesday, February 9, 2011