TEXAS STUDENT DATA SYSTEM

Canonical Data Model
Design Guidelines

Phase 2 Update: December 6, 2010
Ed Comer

©2010 Michael & Susan Dell Foundation. All Rights Reserved.

This document was created by Double Line Partners, LLC with support from the Michael & Susan Dell Foundation.

This document is being shared in draft form and is based on Phase 2 of the District Connections Database project. As additional phases of the project are executed, this document will be updated to reflect refinements, experiences and observations from those efforts.
# Table of Contents

Introduction ........................................................................................................................................... 1  
What is a Canonical Data Model? ........................................................................................................ 1  
Target Audience and Stakeholders ................................................................................................. 2  
CDM Reference Model ..................................................................................................................... 3  
CDM Principles .................................................................................................................................. 3  
CDM Reference Model Notation ....................................................................................................... 3  
CDM Naming Conventions .............................................................................................................. 6  
CDM Domains ................................................................................................................................... 7  
CDM Design Rules and Patterns ....................................................................................................... 7  
CDM XML Schema .......................................................................................................................... 11  
CDM XML Schema Principles ......................................................................................................... 11  
Interchange Schema Design Pattern ............................................................................................... 11  
CDM Core Schema Organization ...................................................................................................... 12  
Domain Types ................................................................................................................................... 13  
Association Types ............................................................................................................................. 13  
Generalizations ............................................................................................................................... 14  
Enumerations ....................................................................................................................................... 15  
Enumerated Collections .................................................................................................................... 15  
Extensions to the CDM Core ............................................................................................................ 16
Introduction
This document outlines the design concepts, principles, patterns, and conventions used for the Canonical Data Model (CDM) and its associated XML schema. This document is a companion to the following:

- **CDM Technical Implementation Guide** – Provides an overview to the CDM XML schema and provides guidance to technical personnel on applying the standard.
- **CDM Reference Model** – Provides a reference for the detailed CDM artifacts and associated cross references.

What is a Canonical Data Model?
A *canonical data model* (CDM) defines the business entities relevant for a specific domain, their attributes, their associations, and their semantics. As a reference model, the CDM is an abstract framework from which concrete implementations are derived. The associated CDM XML schema defines a core set of domain, association, and attribute types, a method for extending the schema, and a method for composing concrete interchange schemas.

A *reference model*, in its broadest sense, is model of something that embodies the basic idea of something and can be looked at as a reference for various purposes. Reference models embody the following concepts:

- **Abstract**: a reference model is abstract. The things described by a reference model are not actual things, but an abstract representation of things.
- **Entities and Relationships**: A reference model contains both entities (things that exist) and relationships (how they interact with one another). A list of entities, by itself, is not sufficient to describe a reference model.
- **Within an Environment**: A reference model does not attempt to describe "all things." A reference model is used to clarify "things within an environment" or a scope or a problem space.
- **Technology Agnostic**: A reference model is a mechanism for understanding the problems faced, not the solutions involved, and as such, must be independent of the selected solutions in order to provide value to the practitioner.

The CDM is a *data reference model* that is meant to enable information sharing and reuse of education data. Specifically the scope of the CDM is focused on P-12 education data relevant to the Local Education Agency (LEA), which includes LEA to State Education Agency (SEA) or LEA to Federal reporting.

---

**Target Audience and Stakeholders**

The target audience for the CDM includes high level technical personnel consisting of enterprise architects and data architects.

The stakeholders for the CDM are:

- LEA personnel who produce or consume data, including superintendents, principals, teachers, counselors, IT managers, and SEA and Federal data submissions specialists
- SEA and Federal level personnel involved in setting data requirements for LEAs, accepting LEA data submissions and utilizing and interpreting LEA data for policy and decision making.
- Regional education service centers (ESCs) that provide data-related services to LEAs
- Vendors engaged in providing information technology support and tools to the education industry, particularly at the LEA level.
- Researchers designing or engaged in education data analysis.
CDM Reference Model

The canonical data model (CDM) defines the business entities relevant for the education domain, their associations, and their semantics. As a reference model, the CDM is an abstract framework from which concrete implementations are derived.

The associated CDM XML schema defines a core set of domain, association, and attribute types, a method for extending the schema, and a method for composing concrete interchange schemas.

CDM Principles

The CDM reference model is based upon the following core principles:

- **Understandability.** The CDM reflects the common vocabulary of the domain. It does not introduce obtuse abstractions. It is modularly organized into readily understood views.

- **Independence.** The CDM is independent of any specific application, but rather synthesizes the data integration requirements of all relevant applications.

- **Immutable Kernel.** The core structures (entities and relationships) of the CDM are logically common across all applications and are not expected to change.

- **Extensibility.** The CDM is designed to be extended for particular installations and for future change and evolution.

- **Commutative Mapping.** The process of mapping specific information models into the CDM must preserve data and operations.

- **Separation of Concerns.** The CDM logically separates the data by who controls or owns the data. For example, state-specific information (in our case information that applies only to Texas) is separated out into an “extension” on the model.

CDM Reference Model Notation

The CDM reference model is expressed as Unified Model Language (UML) class diagrams, using a subset of the symbology as explained in this section. UML Class diagrams capture the logical structure of a domain as a set of classes, their features (aka attributes) and the relationships (aka associations) between them. UML class diagrams were specifically selected over Entity-Relationship diagrams because of their ability to also capture the generalization associations between classes.

---

2 Unified Modeling Language Version 2.2, Object Management Group, Feb 2009,
[http://www.omg.org/spec/UML/2.2/](http://www.omg.org/spec/UML/2.2/)
We dispute the NEDM claim that ER and UML diagrams are only appropriate to represent logical or physical models. Both have been widely used for reference models like the CDM. We similarly disagree with the NEDM claim that addition of attribute detail cannot be represented using ER diagrams or UML object diagrams. ER and UML tools support the entry of attribute detail.

Classes represent the major entities or objects in the education domain. Physical entities such as students, teachers, campuses and locations are represented as classes. Non-physical entities in the education space are also represented in this manner, such as courses, sections, attendance events, and discipline action.

The attributes of classes represent features or characteristics of the entity that are important in our scoped education domain. For example, gender is an attribute of a student that is important to capture in the education domain.

Complex attributes that have many components are defined as separate classes and referenced as the type of the attribute. For example, the complex attribute type address has components street address, city, state, and zip code.

Unless otherwise indicated, attributes are of cardinality 1, meaning that they are single valued and required. Other cardinalities are shown in square brackets.

Relationships or associations represent logical connections between entities that are important in our scoped education domain. For example, students are associated with campuses through enrollment. The direction of an association indicates readability (as in Student HasAssociated Campus) in the domain model. The direction of the association has additional meaning in the XML schema, indicating the class from which the relationship is specified. Cardinalities (e.g., 1-to-1, 1-to-many, many-to-many) are shown for all associations.

An association class is used when an association has attributes. For example, the date of enrollment is an important attribute of the enrollment association between student and campus.

---

The CDM uses a subset of the NEDM association labels, as follows:4

- **HasAssociated** - The object entity has a relationship to the associated subject entity.
- **HasCausalRelationship** - A strong relationship in which one entity causes a change in another entity.
- **HasFunctionalComponent** - Reflects the construction of an entity through functional components represented by other entities.
- **IsDirectProviderOf** - Directly provides goods or services.
- **IsFunctionalComponentOf** - This relation indicates that subject entity makes up, in part or in whole, the function of the object entity.
- **isOrganizationalComponentOf** - This relation is used to indicate an organizational structure of non-person entities such as schools, districts, etc.
- **ParticipatesIn** - A person-type entity participates in an activity-type entity.
- **ProvidesServicesTo** - Subject provides services to the object.
- **ReceivesServicesFrom** - Subject receives services from the object.

The **IsA** or generalization association indicates that a more specialized subclass is a generalization of a broader super class. For example, a *campus* is a specialization of the more general *education organization*. The attributes and associations defined for the super class are *inherited* by the more specialized subclasses. The NEDM relationship **isDerivedFrom** has similar semantics. Generalization semantics are compatible with Type *extensions* in XML schemas.

State-specific attributes are separated into extension classes, shown as specializations of the more generic core education classes that encapsulate the attributes that would apply on a national level.

The CDM described in this document includes the extensions for Texas; other states will likely have a different set.

---

4 National Education Data Model, [http://nces.sifinfo.org/datamodel/eiebrowser/relationships.aspx](http://nces.sifinfo.org/datamodel/eiebrowser/relationships.aspx)
Notes are included in the CDM reference model to explain or elaborate on points not directly interpretable from the UML model.

CDM Naming Conventions
The naming conventions apply to both the CDM Reference Model and the associated CDM XML schema. Consistent naming is used across the CDM and is expected to be maintained in all uses of the CDM (e.g., logical data models, data warehouse models) to the extent that it is practical.

The following naming conventions are applied to the CDM:

- Concatenated title case is used for entities, relationships and attributes (e.g., “CampusId”)
- Names use common terminology for the education industry
- Do not end attribute names with “Type.” In the XML representation, their types may be suffixed with “Type.”
- By default, the entity, attribute, and association names match those of the National Education Data Model (NEDM) at [http://nces.sifinfo.org/datamodel/Index.aspx](http://nces.sifinfo.org/datamodel/Index.aspx)
  - Unless there is not an analogous name in the NEDM; or
  - Unless the NEDM name does not reflect common terminology
- Associations are named by concatenating the two entities they relate and ending with “Association;” for example: “StudentCampusAssociation”
- Entity names are not to be used in attribute names. (E.g., a District table shouldn’t have a field called DistrictName,” it should just be “Name.”) Exceptions are:
  - The primary “Id” field for an object (e.g., Districtld is okay); or
  - The attribute is an enumeration that is different from others of the same name in other entities; or
  - Field names that don’t make sense or are ambiguous without the name.

The conventions used for State-Specific Extensions are as follows:

- State-specific attributes are isolated into state-specific extensions of entities, labeled as <EntityName or AssociationName>Extension.
- State specific attributes are prefixed by their two letter state abbreviation (e.g., TX-
  <AttributeName>
- Attributes that are identified for rework or later retirement are identified as a “Legacy” attribute (e.g., TX-Legacy-<AttributeName>)

The exact same naming is used for the CDM Reference Model and the associated CDM XML schema. All uses of the CDM should carry forward the naming conventions.
CDM Domains
The CDM reference model is organized into domains in response to the principles of Understandability and Separation of Concerns. Domains serve to provide overlapping views of the CDM Reference Model to assist in its understanding and its application. The 15 domains are defined as follows:

- Alternative/Supplemental Services
- Assessment
- Bell Schedule
- Discipline
- Education Organization
- Enrollment
- Finance
- Graduation
- School Calendar
- Staff
- Student Academic Record
- Student Attendance
- Student Cohort
- Student Identification and Demographics
- Teaching and Learning

Additional subdomains are provided for specific cases of Alternative/Supplemental Services:

- Bilingual/ESL Programs
- Career and Technical Education
- Flexible Attendance
- Optional Extended Year Program
- Special Education
- Title I Part A Services

CDM Design Rules and Patterns
The building blocks of the CDM reference model are entities (classes), attributes, and relationships (associations). Even with such seemingly simple building blocks there are a multitude of design choices. This section explains the design rules and patterns applied in creating the CDM reference model.

1. **Stay within the defined scope.** Does the entity, attribute, or relationship exist within the defined scope? Is it important for a specific purpose? Is it available to be collected? Under what scenarios would it be interchanged? Is it required for visualization or reporting?

2. **Consider the “natural state and structure” of the data.** Entities are naturally the most important “things” in the domain that require representation in data. Attributes
naturally identify, describe, characterize or classify entities. Relationships between entities are not transient and typically persist over time.

3. **Include only generalizations that contribute critical inheritances and represent important generalizations within the defined scope.** Is the generalization a commonly used term within the domain? Does the generalized entity have common *important* attributes that should be inherited? Does the generalized entity have relationships that should be inherited? Are all of the specialized entities (subclasses) of common purpose and structure? The NEDM model has a rich taxonomy of generalizations. However most of these generalizations are not important when creating a reference model used for data storage, exchange, and visualization and reporting. For example, the entities for TeachingLearningResource, Assessment, and AssessmentResult are subclasses of the generalized InstructionalArtifact. Note the subclasses are of different purpose. Further examination would reveal that they do not share common attributes or relationships outside of perhaps a name, description and ID. TeachingLearningResource is a poor choice for a reference model. By contrast, the generalization EducationOrganization has important attributes and associations that are meaningfully inherited by Campus, LEA (district), and regional ESC. Note that all are of the same common purpose and structure.

4. **Create classes (complex types) to abstract cohesive groups of attributes.** To improve Understandability, the CDM abstracts collections of attributes into classes. This results in the CDM entities having much shorter and easier to understand lists of abstracted attributes. For example, student name could be represented a flat structure as attributes of Student, having FirstName, MiddleName, LastName, NameSuffix. Instead a new class (in XML a new complex type) is created for Name that includes the attributes above. This reduces the number of attributes directly shown for Student without loss of understandability. The CDM uses this technique for the following:
   - Address
   - Name
   - BirthData
   - Telephone
   - ElectronicMail

5. **Create classes (complex types) to group attributes that are multi-valued records.** Some attributes are expressed as multi-valued records. Rather than create a separate domain class, the record attributes are defined as a separate class (complex type in XML), and then referenced a multi-valued attribute. For example, students may have multiple disabilities, each specified by a disability type, a description, and an order of severity (primary, secondary, ...). The CDM uses this technique for:
   - AnalyticScore
   - Disability
   - EmploymentPeriod
   - RatingStandardsOutcome
6. **Collapse Multiple Boolean Indicators into a Multi-Valued Enumerated Collection.**

There are cases where there are sets of related collections of Boolean indicators. For example, the various race components in a student’s heritage are typically reflected as a set of Boolean indicator codes for American Indian – Alaskan Native, Asian, Black – African American, native Hawaiian – Pacific Islander, White. For extensibility, the CDM grouped these into a multi-valued enumerated collection. The CDM applied this technique to the following:

- CourseLevelCharacteristics
- CTEServices
- HomeSituations
- LinguisticAccommodations
- Race
- SpecialAccommodations
- SpecialEdServices
- TitleIPartAServices
- TitleIPartASupportServices

7. **Carefully consider the direction of associations in the context of possible interchanges.**

When associations do not have their own attributes, the ReferenceType embodying the association is contained in the complex type for the source Entity. It is important to consider entity is the source for the association based its likely use in interchanges. Consider the associations for the AttendanceEvent shown in the model below.

As shown, the association specifying the student associated with the attendance event is contained in the source AttendanceEvent. It is reasonable to assume that there could be an interchange loading just attendance events. Note that is the direction of this association would be reversed then loading attendance would need to always be accompanied by a load of Students. Note this same rationale applies to the Section association from the AttendanceEvent.
8. **Use real data to validate the CDM.** Data profiling is a recognized technique for designing database schemas and data warehouse structures. Similarly, analyzing real data as it is housed in different student information and other education systems provided significant validation of the CDM reference model.
CDM XML Schema

An XML schema is collection of type and element definitions, using an XML format that specifies the rules surrounding the logical structure of some collection data to be specified in XML. Based upon the CDM Reference Model, the CDM XML Schema provides:

- A core set of domain, association, and attribute types that directly map to the CDM Reference model
- A method and example for extending the core schema to account for state-specific, or even interchange-specific data
- A method and examples of composing concrete interchange (microformat) schemas, reusing the types defined in the core and extension schemas.

CDM XML Schema Principles

XML schemas enforce rules around the content of XML instance documents. However there are many different ways and styles for XML schemas to accomplish the same goal. The CDM XML schema was designed with consideration of the following principles:\(^5\)

- **Consistency** – The XML schema should have a consistent organization and design pattern
- **Extensibility** – The XML schema should be designed to easily allow for customizations to meet specific interchange requirements.
- **Flexibility** – The XML schema should be flexible to support the interchange of different subsets and collections of education data.
- **Reuse** – The XML schema should be designed to facilitate the reuse of types of elements in constructing the concrete schemas (microformats) used for interchange.
- **Composition Using XML Mechanisms** – The creation of concrete interchange schemas from core and extended schemas should be accomplished using native XML mechanisms, not through a cut ‘n paste operation.

Interchange Schema Design Pattern

The Venetian Blind Design is one of several popular and documented XML schema styles. It facilitates reuse while also hiding namespace complexities.\(^6\) For each interchange schema, the style defines a single global element that nests local elements that use types (simple or complex) that are defined within the global namespace.

---


\(^6\) One such reference is at: [http://developers.sun.com/jsenterprise/archive/nb_enterprise_pack/reference/techart/design_patterns.html](http://developers.sun.com/jsenterprise/archive/nb_enterprise_pack/reference/techart/design_patterns.html)
For example, consider the interchange schema for students, parents, and their relationships, depicted below.

```xml
<?xml version="1.0" encoding="UTF-8"?>
    targetNamespace="http://msdf.org/tsds" elementFormDefault="qualified"
    attributeFormDefault="unqualified">
    <xs:element name="InterchangeStudentParent">
        <xs:complexType>
            <xs:choice maxOccurs="unbounded">
                <xs:element name="Student" type="Student"></xs:element>
                <xs:element name="Parent" type="Parent"></xs:element>
                <xs:element name="StudentParentAssociation" type="StudentParentAssociation">
                </xs:element>
            </xs:choice>
        </xs:complexType>
    </xs:element>
</xs:schema>
```

A single element InterchangeStudentParent defines the interchange. The three elements of the interchange, Student, Parent and StudentParentAssociation, are nested within the single element, referencing types in CDM Core (see the include statement above).

Note that we chose to encapsulate the elements of the interchange in an unbounded choice statement rather than a sequence to provide maximum flexibility for the interchange. As a result the various elements of the interchange are optional and can be provided in any order.

**CDM Core Schema Organization**

To support the Venetian Blind design, the CDM Core schema provides a *flat* compendium of XML types. The types which are the *building blocks* of interchange schemas are:

- **Domain** types, XML complex types representing the major entities in the education domain, as modeled in the CDM reference model.
- **Association** types, XML complex types representing those associations between the domain entities that require attributes.

The domain and association types are composed from a second tier of XML simple and complex types, organized:

- **Common** types used in the domain, such as Name, Address or Telephone.
- **Enumeration** types defined the controlled value lists or controlled vocabulary used for many enumerated or enumerated collection attributes.
- **Simple string** types, defining specific constraints, such as length, for the various string typed attributes.
- **Extended reference** types, supporting associations via XML IDREFs or via a lookup

The CDM XML schema (without any state-specific extensions) is the file `CDM-Core.xsd`. 

---

TSDS Working Document / Phase 2  ■  December 6, 2010  ■  12
Domain Types
All domain types are XML extensions of the type ComplexObjectType. This type defines the common id attribute that holds the assignment of an XML IDREF to instances of the domain type used in interchange schemas, as shown below.

```xml
<xs:complexType name="ComplexObjectType" abstract="true">
  <xs:annotation>
    <xs:documentation>This is the base type from which all entity elements are extended.</xs:documentation>
  </xs:annotation>
  <xs:attribute name="id" type="xs:ID"/>
</xs:complexType>
```

Association Types
Each association will have a reference to two or more domain entities. These references will either be:

- **ReferenceType** – that defines a ref attribute to hold the IDREF of the referenced instance, as follows:

  ```xml
  <xs:complexType name="ReferenceType">
    <xs:attribute name="id" type="xs:ID"/>
    <xs:attribute name="ref" type="xs:IDREF"/>
  </xs:complexType>
  ```

- **An extended reference type** – to support cases where a data interchange schema must reference instances of data already loaded, allowing association instances to be specified either by an ID REF or by a set of attributes to be used for lookup by the receiving system. For example, the extended reference type for a Student is:

  ```xml
  <xs:complexType name="StudentReferenceType">
    <xs:complexContent>
      <xs:extension base="ReferenceType">
        <xs:sequence>
          <xs:element name="StudentIdentity" type="StudentIdentityType" minOccurs="0"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  ```

  ```xml
  <xs:complexType name="StudentIdentityType">
    <xs:sequence>
      <xs:element name="StudentUniqueStateId" type="UniqueStateIdentifier" minOccurs="0"/>
      <xs:element name="StudentId" type="StudentId" minOccurs="0"/>
      <xs:element name="CampusLocalId" type="CampusLocalIDType" minOccurs="0"/>
      <xs:element name="Name" type="Name" minOccurs="0"/>
      <xs:element name="Sex" type="SexType" minOccurs="0"/>
      <xs:element name="BirthDate" type="xs:date" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
  ```
For example, note the StudentParentAssociation has a StudentReference and a ParentReference. The ParentReference is of type ReferenceType, described above. However, the StudentReference is of type StudentReferenceType – an *extended reference* type.

```xml
<xs:complexType name="StudentParentAssociation">
  <xs:sequence>
    <xs:element name="StudentReference" type="StudentReferenceType"></xs:element>
    <xs:element name="ParentReference" type="ReferenceType"></xs:element>
    <xs:element name="Relation" type="RelationType" minOccurs="0"></xs:element>
    <xs:element name="PrimaryContactStatus" type="xs:boolean" minOccurs="0"></xs:element>
    <xs:element name="LivesWith" type="xs:boolean" minOccurs="0"></xs:element>
    <xs:element name="EmergencyContactStatus" type="xs:boolean" minOccurs="0"></xs:element>
  </xs:sequence>
</xs:complexType>
```

**Generalizations**

The XML schema *extension* is used to reflect generalizations in the CDM. For example, EducationOrganization is a generalization of Campus, LocalEducationAgency, and EducationServiceCenter.

The complex type EducationOrganization is identified as an abstract type.

```xml
<xs:complexType name="EducationOrganization" abstract="true">
  <xs:complexContent>
    <xs:extension base="ComplexObjectType">
      <xs:sequence>
        <xs:element name="StateOrganizationId" type="StateOrganizationId"></xs:element>
        <xs:element name="NCESId" type="NCESId"></xs:element>
        <xs:element name="NameOfInstitution" type="NameOfInstitution"></xs:element>
        <xs:element name="OrganizationCategory" type="OrganizationCategoryType"></xs:element>
        <xs:element name="PhysicalAddress" type="Address"></xs:element>
        <xs:element name="MailingAddress" type="Address" minOccurs="0"></xs:element>
        <xs:element name="Telephone" type="Telephone"></xs:element>
        <xs:element name="FaxNumber" type="Telephone" minOccurs="0"></xs:element>
        <xs:element name="WebSite" type="WebSite" minOccurs="0"></xs:element>
        <xs:element name="AccountabilityRatings" type="AccountabilityRating" minOccurs="0" maxOccurs="unbounded"></xs:element>
        <xs:element name="ProgramReference" type="ProgramReferenceType" minOccurs="0" maxOccurs="unbounded"></xs:element>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```
Specializations are reflected as extensions of the base type, as follows:

```xml
<xs:complexType name="Campus">
  <xs:complexContent>
    <xs:extension base="EducationOrganization">
      <xs:sequence>
        <xs:element name="LocalEducationAgencyReference" type="EducationalOrgReferenceType"/>
        <xs:element name="InstructionalGradeLevels" type="InstructionalGradeLevelsType"/>
        <xs:element name="SchoolCategory" type="SchoolCategoryType"/>
        <xs:element name="InstructionCategory" type="InstructionCategoryType" minOccurs="0"/>
        <xs:element name="CharterCategory" type="CharterCategoryType" minOccurs="0"/>
        <xs:element name="ClassPeriodReference" type="ReferenceType" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element name="LocationReference" type="ReferenceType" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

Enumerations

Single-valued enumerations are shown as restrictions of the base type xs:token, as illustrated below:

```xml
<xs:simpleType name="GraduationPlanType">
  <xs:restriction base="xs:token">
    <xs:enumeration value="Minimum"/>
    <xs:enumeration value="Recommended"/>
    <xs:enumeration value="Distinguished"/>
  </xs:restriction>
</xs:simpleType>
```

Enumerated Collections

Multi-valued enumerated collections are composed of a complex type that has multiple values of an enumeration, as illustrated below:

```xml
<xs:complexType name="LanguagesType">
  <xs:sequence>
    <xs:element name="Language" type="LanguageItemType" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

```xml
<xs:simpleType name="LanguageItemType">
  <xs:restriction base="xs:token">
    <xs:enumeration value="English"/>
    <xs:enumeration value="Spanish"/>
  </xs:restriction>
</xs:simpleType>
```
Extensions to the CDM Core

Extensions to the CDM Core types are explicitly supported for state-specific data or any other purpose, including reasons specific to an interchange. The CDM Core types may be extended in a separate XML schema file that has the *include* statement to the CDM Core schema. Within this file, the following types of extensions are supported:

- If additional attributes are required for a CDM Core type, a new type may be defined as an *extension* of the appropriate CDM Core type.
- New simple or complex types required for attributes may be added.
- New domain or association classes and their attributes and references may be added.

The Texas-specific extensions to support the state’s PEIMS accountability reporting by LEAs have been defined in the XML schema file `CDM-TX-Extension.xsd`. 