

IBM Cúram Social Program Management
Version 7.0.3

Creating Datastore Schemas



Note

Before using this information and the product it supports, read the information in [“Notices” on page 6](#)

Edition

This edition applies to IBM® Cúram Social Program Management v7.0.3 and to all subsequent releases unless otherwise indicated in new editions.

Licensed Materials - Property of IBM.

© **Copyright International Business Machines Corporation 2012, 2018.**

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

© .

Contents

List of Tables.....	iv
Chapter 1. Creating Datastore Schemas.....	1
Important Information Before Creating Datastore Schemas.....	1
The Purpose of Datastores.....	1
How to Properly Access Datastores.....	1
Creating a Datastore Schema.....	1
Datastore Schema Overview.....	1
Namespaces.....	2
Datatypes.....	2
Entity and Attribute Definitions.....	3
Identity Constraints.....	4
Compliance.....	5
Public API.....	5
Identifying the API.....	5
Outside the API.....	5
Notices.....	6
Privacy Policy considerations.....	7
Trademarks.....	7

List of Tables

- 1. Built-in Domain Definitions..... 2
- 2. Supported Annotation-based Constraints..... 3

Chapter 1. Creating Datastore Schemas

Use this information to create Datastore schemas. The datastore stores data that is collected from citizens during intake. A datastore schema defines the entities across which intake data is shredded, the fields or attributes of those entities, and any relationships between entity definitions.

Important Information Before Creating Datastore Schemas

The Purpose of Datastores

The Datastore stores data that is collected from citizens during online screening and intake of applications. The contents of the datastore are dynamically definable by way of a datastore schema definition. Other dynamically definable components depend on the Datastore Schema, in particular screening and intake scripts, and screening rules, so the datastore contents must be kept compatible with these. In particular, these other components might depend on certain fixed elements of Datastore definitions, which must therefore exist in order for these components to operate properly.

Data on the Datastore is stored in XML format. For reasons of efficiency, the data for a single application is shredded across multiple database rows so that updates to an application during intake do not result in large amounts of inefficient LOB I/O every time a page of an intake script is traversed. Each type of shredded row content is called a datastore *entity*. The datastore schema definition conforms to the World Wide Web Consortium (W3C) XML Schema Definition Language. In addition to having to conform to the W3C specification, a datastore schema must follow certain datastore-specific rules. These rules are described in [“Creating a Datastore Schema” on page 1](#).

How to Properly Access Datastores

The datastore is accessed through an API defined in the Java™ package `curam.datastore.impl`. This package allows for opening a datastore by locating its schema definition, and accessing its data.

For further details, see the Javadoc information for the API package. Note that other `curam.datastore.*` packages do not form part of the API (even where classes and methods are public) and are not to be used.

Creating a Datastore Schema

Datastore Schema Overview

A datastore schema defines the following:

- The entities across which screening and intake data are shredded - These are created as XML *complexType* definitions.
- The fields or *attributes* of those entities - These are created as XML *attribute* definitions.
- The simple datatypes, called *domains*, which define the allowable types of entity attributes - These are created as XML *simpleType* definitions.
- Any key relationships between entity definitions - These are created as XML *key* and *keyref* definitions.

Namespaces

A datastore schema must be defined in a single XML namespace, or in no namespace. It cannot be split across multiple namespaces.

The schema *may* be split across multiple schema fragments that include each other. Schema definitions contain no XML `import` elements other than importing the base domains namespace **`http://www.curamsoftware.com/BaseDomains`**, but they might contain `include` elements.

Datatypes

The attributes in a datastore schema definition cannot directly be of W3C XML built-in datatypes. Instead, they must be valid *Cúram domain definitions*.

This means that they must be of a simple type that is defined in the XML namespace **`http://www.curamsoftware.com/BaseDomains`**, or a simple type that is defined in the datastore schema, which inherits from one of those types. In the example above, `DECEASED_IND` is such a domain definition. The built-in domain definitions in **`http://www.curamsoftware.com/BaseDomains`** are:

Table 1: Built-in Domain Definitions		
Base Domain	Built-in XML Type	Remarks
SVR_BOOLEAN	Boolean	
SVR_INT8	Byte	
SVR_INT16	Short	
SVR_INT32	Int	
SVR_INT64	Long	
SVR_KEY	Long	
SVR_FLOAT	Float	
SVR_DOUBLE	Double	
SVR_DATE	Date	
SVR_DATETIME	dateTime	
SVR_MONEY	Decimal	fractionDigits value="2"
SVR_STRING	String	
SVR_CHAR	String	minLength value="1", maxLength value="1"
CODETABLE_CODE	String	minLength value="1", maxLength value="10"
SHORT_CODETABLE_CODE	String	minLength value="1", maxLength value="10"

When defining new domains, their names cannot conflict with domains that are modeled in the application as they might potentially be displayed with the wrong renderer.

Domain definitions support a limited number of value constraints. The following XML constraints are permitted:

- `minLength`
- `maxLength`
- `minInclusive`

- maxInclusive

In addition to the supported XML constraints, there are a number of Cúram-specific constraints that can be specified by using XML annotations, as follows:

```
<xsd:simpleType name="SOME_DOMAIN">
  <xsd:annotation>
    <xsd:appinfo>
      <d:options>
        <d:option name="min-size">3</d:option>
      </d:options>
    </xsd:appinfo>
  </xsd:annotation>
  <xsd:restriction base="d:SVR_STRING"/>
</xsd:simpleType>
```

The list of supported annotation-based constraints is as follows:

Table 2: Supported Annotation-based Constraints	
Annotation Option	Remark
min-size	Same as XML minLength
max-size	Same as XML maxLength
minimum	Same as XML minInclusive
maximum	Same as XML maxInclusive
default	Default value
compress	Reduce embedded runs of spaces to a single space
to-upper	Convert text to uppercase
trim-leading	Trim leading space
trim-trailing	Trim trailing spaces
code-table-name	Name of a Cúram code table that is used to translate code value
code-table-root	Do not use this option in user domain definitions.

Note that domain constraints have no effect on processing in the datastore when the datastore is accessed programmatically. They exist purely to provide metadata to screen processing facilities such as Cúram Intelligent Evidence Gathering(IEG), which can use the metadata to constrain user input.

Entity and Attribute Definitions

Entities in the Datastore are defined as elements with *complexType* definitions. Each entity's definition must appear at global level in the schema. If an entity incorporates another entity as part of its definition, the incorporated entity must still be defined at global level and *referred* to by the incorporating entity.

In the example below, the Application entity incorporates the Person entity, which in turn incorporates the Address entity. All three entities are defined at global level and use *element ref* to refer to the definitions of other entities.

```
<xs:element name="Application">
  <xs:complexType>
    <xs:sequence minOccurs="0">
      <xs:element ref="Person" minOccurs="0"
        maxOccurs="unbounded" />
    </xs:sequence>
  </xs:complexType>
</xs:element>

<xs:element name="Person">
```

```

        <xs:complexType>
          <xs:sequence minOccurs="0">
            <xs:element ref="Address"
              minOccurs="0"
              maxOccurs="1" />
            ...
          </xs:sequence>
          <xs:attribute name="firstName" type="D:SVR_STRING" />
          ...
        </xs:complexType>
      </xs:element>

      <xs:element name="Address">
        ...
      </xs:element>

```

There are other restrictions on entities. All complex type compositors must be sequences, that is, you can use the "sequence" compositor but not "all" or "choice". All sequences must have a "minOccurs" of zero, and all elements must have a "minOccurs" of zero. There is no restriction on the value of the "maxOccurs" constraint.

Attributes of entities must be of "domain" types, that is, they must be *simpleType* definitions that are defined elsewhere in the schema. An attribute can have a default value, which (unlike the "default" annotation on domains) *does* directly affect the values that get stored on the Datastore if the attribute value is not explicitly set.

The datastore definition usually consists of many *simpleType* (domain) and *complexType* (entity) definitions. All of these must appear at global level, that is, they cannot be nested inside each other.

```

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:D="http://www.curamsoftware.com/BaseDomains"
>

  <xs:import
    namespace="http://www.curamsoftware.com/BaseDomains" />

  <xs:simpleType name="DECEASED_IND">
    <xs:restriction base="D:SVR_BOOLEAN"/>
  </xs:simpleType>

  <xs:element name="Person">
    <xs:complexType>
      <xs:attribute name="firstName" type="D:SVR_STRING" />
      <xs:attribute name="middleInitial"
        type="D:SVR_STRING" />
      <xs:attribute name="lastName" type="D:SVR_STRING" />
      <xs:attribute name="deceased" type="DECEASED_IND" />
    </xs:complexType>
  </xs:element>

</xs:schema>

```

Identity Constraints

Consider the example that is shown here. It models a Person that can have an Address, and zero or more Relationships to other Persons. A Relationship does not "contain" a Person. Rather, it has a reference to a Person.

How do you ensure that one entity uniquely refers to another entity? You use XML identity constraints. These constraints are well-described in the relevant XML specifications and no elaborations are required, other than to point out more restrictions on the use of identity constraints in a Datastore Schema.

```

<xs:element name="Person">
  <xs:complexType>
    <xs:sequence minOccurs="0">
      <xs:element ref="Address"
        minOccurs="0"
        maxOccurs="1" />
      <xs:element ref="Relationship" minOccurs="0"
        maxOccurs="unbounded" />
    </xs:sequence>
    <xs:attribute name="personID"
      type="D:SVR_KEY"/>
    <xs:attribute name="firstName"

```



```

        type="D:SVR_STRING" />
        <xs:attribute name="middleInitial"
            type="D:SVR_STRING" />
        <xs:attribute name="lastName" type="D:SVR_STRING" />
    </xs:complexType>
    <xs:key name="PersonKey">
        <xs:selector xpath="./Person"/>
        <xs:field xpath="@personID"/>
    </xs:key>
    <xs:keyref name="RelationshipRef" refer="PersonKey">
        <xs:selector xpath="./Person/Relationship"/>
        <xs:field xpath="@personID"/>
    </xs:keyref>
</xs:element>
<xs:element name="Address">
    <xs:complexType>
        <xs:attribute name="street1" type="D:SVR_STRING" />
        <xs:attribute name="street2" type="D:SVR_STRING" />
        <xs:attribute name="city" type="D:SVR_STRING" />
        <xs:attribute name="state" type="D:SVR_STRING" />
        <xs:attribute name="zipCode" type="D:SVR_INT32" />
    </xs:complexType>
</xs:element>
<xs:element name="Relationship">
    <xs:complexType>
        <xs:attribute name="type" type="D:SVR_STRING"/>
        <xs:attribute name="personID" type="D:SVR_KEY"/>
    </xs:complexType>
</xs:element>

```

An attribute referred to in a *key* or *keyref* identity constraint must be defined to be of the base domain type SVR_KEY. Conversely, every attribute of type SVR_KEY must be referred to by some *key*/ or *keyref* definition. Key attributes have their values that are automatically populated upon insertion into the Datastore. They are set to a numeric value that is unique across all entities on the Datastore.

Compliance

This section explains how to develop in a compliant manner. By following these considerations, customers will also find it easier to upgrade to future versions of Cúram.

Public API

The Datastore has a public API which you may use in your application code. This public API does not have any components that are changed or removed without following Cúram standards for handling customer impact.

Identifying the API

The JavaDoc that is included is the sole means of identifying which public classes, interfaces, and methods form the public API.

Outside the API

The Datastore also contains some public classes, interfaces, and methods, which do not form part of the API.

Important: To be compliant, dependencies on any class or interface cannot be made. No methods are called other than those described in the Javadoc information.

Classes, interfaces, and methods outside of the public API are subject to change or removal without notice. Unless otherwise stated in the JavaDoc, you must not place any of your own classes or interfaces in the same package as the Datastore.

Notices

This information was developed for products and services offered in the United States.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing IBM Corporation North Castle Drive, MD-NC119 Armonk, NY 10504-1785 US

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing Legal and Intellectual Property Law IBM Japan Ltd. 19-21, Nihonbashi-Hakozakicho, Chuo-ku Tokyo 103-8510, Japan

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Director of Licensing IBM Corporation North Castle Drive, MD-NC119 Armonk, NY 10504-1785 US

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

The performance data and client examples cited are presented for illustrative purposes only. Actual performance results may vary depending on specific configurations and operating conditions.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Privacy Policy considerations

IBM Software products, including software as a service solutions, ("Software Offerings") may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering's use of cookies is set forth below.

Depending upon the configurations deployed, this Software Offering may use session cookies or other similar technologies that collect each user's name, user name, password, and/or other personally identifiable information for purposes of session management, authentication, enhanced user usability, single sign-on configuration and/or other usage tracking and/or functional purposes. These cookies or other similar technologies cannot be disabled.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, see IBM's Privacy Policy at <http://www.ibm.com/privacy> and IBM's Online Privacy Statement at <http://www.ibm.com/privacy/details> the section entitled "Cookies, Web Beacons and Other Technologies" and the "IBM Software Products and Software-as-a-Service Privacy Statement" at <http://www.ibm.com/software/info/product-privacy>.

Trademarks

IBM, the IBM logo, and [ibm.com](http://www.ibm.com) are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at <http://www.ibm.com/legal/copytrade.shtml>.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other names may be trademarks of their respective owners. Other company, product, and service names may be trademarks or service marks of others.



Part Number:

(1P) P/N: