TECHNICAL REPORT

ISO/IEC TR 19583-21

First edition
2022-08

Information technology — Concepts and usage of metadata —

Part 21:
11179-3 Data model in SQL

Reference number
ISO/IEC TR 19583-21:2022(E)

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Contents

Page

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives or www.iec.ch/members_experts/refdocs).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents) or the IEC list of patent declarations received (see https://patents.iec.ch).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see www.iso.org/iso/foreword.html. In the IEC, see www.iec.ch/understanding-standards.

This document was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 32, *Data management and interchange*.

A list of all parts in the ISO/IEC 19583 series can be found on the ISO and IEC websites.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html and www.iec.ch/national-committees.

Introduction

ISO/IEC 11179-3[1] provides a specification for a registry in which information about metadata can be recorded and maintained.

The metamodel to instantiate such a registry is expressed in text as a conceptual model. This conceptual model is illustrated with a series of diagrams which use the class diagram notation from the Unified Modeling Language (UML)[2][3].

Instantiaters and users of the registries described in ISO/IEC 11179-3 require further guidance to turn the conceptual models into concrete instantiations. This document provides a possible instantiation of the registry metamodel specified in ISO/IEC 11179-3 using the SQL database language as specified in ISO/IEC 9075[4].

This specimen instantiation is provided to increase the understanding of ISO/IEC 11179-3 and, hence, to promote its adoption.

Information technology — Concepts and usage of metadata —

Part 21:
11179-3 Data model in SQL

1 Scope

This document provides a possible instantiation of the registry metamodel specified in ISO/IEC 11179-3 using the SQL database language as specified in ISO/IEC 9075-2.

2 Normative references

There are no normative references in this document.

3 Terms and definitions

No terms and definitions are listed in this document.

ISO and IEC maintain terminology databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at https://www.iso.org/obp

— IEC Electropedia: available at https://www.electropedia.org/

4 Overview of the relationship between UML Class Diagrams and SQL

The Unified Modeling Language (UML) provides a family of graphical notations that can be used in the analysis and design of software systems. The UML is under the control of the Object Management Group (OMG) and, as such, it is (a) a relatively 'open' standard, and (b) firmly rooted in the object-oriented paradigm for software engineering. The UML is now at Version 2 and is the subject of two international standards: ISO/IEC 19505-1 and ISO/IEC 19505-2.

Within the UML, the Class Diagram notation is used to represent information (and, hence, data) requirements for a particular 'universe of discourse', a business area or the scope of a proposed information system.

A UML Object is often defined as a:

> ***construct within a system for which a set of attributes and operations can be specified.***

Whilst this is a reasonable definition within the context of object-oriented system development, a more appropriate definition of an Object for the purposes of this document is a:

> ***representation of something of interest within the universe of discourse about which information needs to be recorded.***

An Object Class in both contexts can then defined as a:

> ***definition of a set of Objects that share the same attributes, associations, and operations.***

The Database Language SQL is a, largely, declarative language used to manage structured data held in a database under the control of a Relational Database Management System (RDBMS). As such, it

was originally based on Edgar F. Codd's relational model of data published in 1970[5], but its scope has grown over the years. SQL is the subject of the multi-part set of International Standards, ISO/IEC 9075 series. Most commercial SQL products, however, deviate from the standards to some extent, some more than others.

5 Generating the SQL for the metamodel

5.1 Overview

The UML (and the Class Diagrams, in particular) and the SQL database language exist in two separate programming paradigms and there is, therefore, no direct translation from one (the UML) to the other (SQL). There are, however, approaches that can be taken to achieve a translation. This document uses one of those approaches to generate a set of SQL statements to instantiate the metadata registry metamodel, where the SQL statements enable easy reference back to the original UML Class Diagram and text of the metamodel. This is achieved by using names for the SQL objects that reflect the names of the UML artefacts and, also, by embedding comments referencing the metamodel within the SQL statements.

5.2 General principles for the translation of a UML Class diagram into SQL statements

It is good practice to distinguish between SQL keywords and the names given to the SQL objects. One convention is to use UPPER CASE for the keywords and lower_case (using snake case) for the object names.

Each UML class is represented by an SQL table. To make correlation to the model easier, the name of a table that represents a class is the same as that of the class.

Each composite datatype is also represented by an SQL table. To make correlation to the model easier, the name of the table is the same as that of the datatype.

Each single-valued attribute of a class or a composite datatype is represented by a column in the appropriate table. The name of this column is the same as that of the attribute in the class or datatype. The datatype of an attribute column is intuitively selected to be similar to that of the datatype of the attribute.

If the datatype of an attribute of a class is another class or a composite datatype, the column that represents that attribute is additionally declared as a foreign key column referencing the relevant table that represents the class or composite datatype.

Where an attribute is multivalued (that is, it has a multiplicity of [0..*] or [1..*] in the UML diagram) there are two possible instantiations available. These are:

a) Use one of the collection types, MULTISET or ARRAY, available in SQL.

b) Create a new table, a characteristic table, to hold the multiple values, with each row in the table having a foreign key referencing the kernel (prime) table and one of the values.

In object-orientation, and, hence, UML, there is no equivalent of the SQL primary key, so each table that represents a class or a composite datatype has an additional column that is used as a surrogate identifier. This column then becomes the primary key for the table.

UNIQUE or CHECK constraints may be added to a table where required. The latter are used, for example, to control the valid values for a column or to control which columns should, or should not, take values in different circumstances.

Specialization hierarchies (superclasses and their subclasses) can be instantiated in one of two ways using SQL structures.

1) It is possible to integrate all the classes (the superclass and its subclasses) into one table, named after the superclass, which includes a column for each attribute of the superclass, with those

columns representing mandatory attributes being declared NOT NULL, and columns for each of the attributes of the subclasses. None of these subclass columns are declared as NOT NULL, irrespective of whether they are mandatory or optional within their subclass. An additional column, often called a discriminator column, with possible values representing each of the subclasses and a CHECK constraint are provided to manage which columns are populated for each subclass. Where a subclass is related to another class via a one-to-many association, such that there is a foreign key in the table representing the other class, a cross-table constraint is needed to ensure that a row in the other table will only exist if the value of the discriminator column represents the relevant subclass in the inheritance relationship.

2) Another possibility is the creation of a separate table for the superclass and the creation of additional tables for each of the subclasses. When the hierarchy is complete, disjoint and static (the normal situation in most models), a fully mandatory 'one-to-one' relationship is provided between the table representing the superclass and each table representing the subclasses. To achieve this, the primary key of each subclass table is also declared as a foreign key referencing the superclass table. Cross-table constraints are needed to ensure that for each row in the superclass table there is a corresponding row in one of the subclass tables, and that there is no duplication of primary key values across the tables representing the subclasses.

There are a number of different approaches that can be used when translating UML Class Diagram associations into SQL. Since each association in the metadata registry metamodel is named, the approach used in this document is to create a table for each association, with the table named with the name of the association.

Some many-to-many associations are annotated with association classes. These association classes also become tables.

5.3 Specific approaches taken for the translation of the metadata registry metamodel

5.3.1 Overview

The following subclauses provide specific detail about the translation of the metamodel artefacts, where the information in 5.2 is either not applicable or insufficient.

5.3.2 Obligations

In the metamodel the obligation applicable to each attribute or association are described as one of "Mandatory", "Conditional" or "Optional", with these obligations being enforced if, and only if, the Registration Status of the associated metadata item is Recorded or higher, that is, if the Registration Status of the associated item is one of "Recorded", "Qualified", "Standard" or "Preferred Standard". The obligations are not enforced if the Registration Status of the associated item is one of "Candidate", "Incomplete", "Retired" and "Superseded".

Any registry instantiation has to be able to register items with a lower Registration Status than "Recorded", and the obligations cannot, therefore, be simply enforced.

The example SQL instantiation allows the attributes and associations to be optional so that items with a Registration Status lower than "Recorded" can be accommodated. The obligations applicable to items with a Registration Status of "Recorded" or higher will, therefore, need to be enforced in the registry application as opposed to the register (the database) itself.

5.3.3 Translation of datatypes

The datatypes used in the metamodel can be considered to be 'primitive' or more complex.

The primitive datatypes are translated as described in Table 1.

Table 1 — Translation of the primitive datatypes

Metadata registry metamodel datatype	Examples or Comment	SQL Datatype
Boolean		This simply translates as a column of type BOOLEAN
Date		This simply translates as a column of type DATE
Datetime		This simply translates as a column of type TIMESTAMP
Integer		This simply translates as a column of type INTEGER
Notation	XCL Common Logic, OWL-DL XML	This translates as a column of type CHARACTER VARYING (2500)
Sign	This could be a bit string, but, at a minimum, String must be supported.	This translates as a column of type CHARACTER VARYING (2500). If the SQL implementation supports the BINARY LARGE OBJECT type, this could be used instead.
String		This translates as a column of type CHARACTER VARYING (255)
Text		This translates as a column of type CHARACTER VARYING (2500). If the SQL implementation supports the CHARACTER LARGE OBJECT type, this could be used instead.

The more complex datatypes are translated as described in Table 2.

Table 2 — Translation of the more complex datatypes

Metadata registry metamodel datatype	Examples or Comment	SQL Datatype
Natural_Range	0, 1, 2, 1..2, 2..8, 0..*, 3..*	This is instantiated with three columns, one INTEGER column for the lower bound, one INTEGER column for the fixed upper bound, and a CHARACTER column, defaulting to 'many', for the many upper bound. The columns are then managed with a CHECK constraint.
Value	This represents a value of any of the types listed above	This is instantiated using many different columns, one, or more, for each of the datatypes listed and then a CHECK constraint implemented to ensure that only one datatype is represented.
Phone_Number		A table is created (named cdt_phone_number) with columns as specified in ISO/IEC 19773[6]; the table has a surrogate primary key of datatype INTEGER. Tables representing classes that have attributes specified with this datatype have foreign keys that reference cdt_phone_number.
Postal_Address		A table is created (named cdt_postal_address) with columns as specified in ISO/IEC 19773[6]; the table has a surrogate primary key of datatype INTEGER. Tables representing classes that have attributes specified with this datatype have foreign keys that reference cdt_postal_address.

5.3.4 Translation of the basic classes

Each basic class in the metamodel is represented by a table, with the name prefixed by cls_. Each table representing a basic class has a surrogate primary key of type INTEGER.

5.3.5 Translation of the <<type>> stereotypes

Some classes in the metamodel are used as stereotypes marked <<type>>, enabling other classes to be 'typed'; that is, to be identified, designated or classified. These classes are instantiated as tables with the names prefixed by typ_. Each of these tables representing a superclass in a specialization hierarchy has

a surrogate primary key of type INTEGER. Those tables that are then subclasses inherit the surrogate primary key of the superclass, with that same primary key also being declared as a foreign key.

Those classes that can be 'typed' are then subject to a set of ALTER TABLE statements to add columns and foreign keys to reference the tables representing the <<type>> classes.

5.3.6 Translation of the remaining classes

Each of the remaining classes in the metamodel is represented by a table, with the name prefixed by cls_. Each table representing a class has a surrogate primary key of type INTEGER.

5.3.7 Translation of specialization hierarchies

Each superclass and its subclasses are represented by separate tables. Those tables that represent subclasses inherit the surrogate primary key of the superclass, either as the primary key or as an additional column, with that inherited primary key also being declared as a foreign key. This is normally achieved by ALTER TABLE statements after the creation of the table.

5.3.8 Translation of the association classes

Each association class in the metamodel is represented by a table, with the name prefixed by asscls_. Each table representing an association class has a surrogate primary key of type INTEGER, a column, or columns representing the attributes of the association class, and two additional columns: one each for the roles of the association class, both of which have datatype INTEGER and are specified as foreign keys referencing the tables representing the relevant classes.

5.3.9 Translation of the attributes of the classes

Where the metamodel datatype of the attribute is listed in the first column of Table 1, the attribute is translated as a single column of the relevant table representing the parent class, with a datatype as specified in the third column of Table 1.

Where the metamodel datatype of the attribute is listed in the first column of Table 2, the attribute is instantiated as specified in the third column of Table 2.

Where the attribute is a multivalued attribute, there are three options available:

— Where the metamodel datatype is specified in Table 1 and the values are unordered, a separate characteristic table, prefixed mva_, is created with a column for the multivalued attribute specified with the SQL datatype identified in Table 1. The second column in this table is a foreign key column referencing the table representing the class that specifies the attribute. Both columns are declared as the primary key. If the SQL implementation supports the MULTISET collection type, this can be used instead of creating the characteristic table.

— Where the datatype is specified in Table 1 and the values are specified as being ordered, or it makes sense to order them, a separate characteristic table, prefixed mva_, is created with a column for the multivalued attribute specified with the SQL datatype identified in Table 1. The second column in this table is a column of datatype INTEGER named 'priority' to indicate the order of the value. The third column in this table is a foreign key column referencing the table representing the class that specifies the attribute. All three columns are declared as the primary key. If the SQL implementation supports the ARRAY collection type, this could be used instead of creating the characteristic table.

— Where the datatype is specified in Table 2, a separate characteristic table, prefixed mva_, is created with a column for the multivalued attribute specified with the datatype INTEGER and as a foreign key referencing the table representing the complex datatype. The second column in this table is a foreign key column referencing the table representing the class that specifies the attribute. Both columns are declared as the primary key.

Where the datatype of the attribute is specified as another class specified in the metamodel, the attribute is instantiated as a column specified with the datatype INTEGER and as a foreign key referencing the table representing the class that is specified as the datatype in the metamodel. Where feasible, this column is included in the CREATE TABLE statement. If this is not feasible, later ALTER TABLE statements are used to add the column and the foreign key.

5.3.10 Translation of the associations

Each association in the metamodel is represented by a table, with the name prefixed by ass_. Each table representing an association has two columns: one each for the roles of the association, both of which have datatype INTEGER and are specified as foreign keys referencing the tables representing the relevant classes.

Each association is one of three basic types, as follows:

— A one-to-many association: In this case, the column referencing the class at the many end is specified as the primary key of the table.

— A many-to-many association: In this case, both columns are declared as the primary key of the table.

— A one-to-one association: In this case, both columns are declared as the primary key of the table, as for the many-to-many association, but, in addition, each column is declared with a UNIQUE constraint to enforce the one-to-oneness.

5.3.11 Cross-table constraints

Because obligations specified in the metamodel are only applicable if, and only if, the Registration Status of the associated metadata item is Recorded or higher, very few of the constraints needed to enforce the obligations are included in the example SQL.

The only cross-table constraints included are those to ensure that, where appropriate, subclasses are disjoint. This is done by checking that there are no duplicate values for the primary keys across all of the subclasses in the hierarchy. There are only two disjoint hierarchies in the metamodel: the hierarchy with Registered_Item as the superclass and the hierarchy with Concept as the superclass.

To check for 'disjointness', it is necessary to ensure that, for each of the subclasses, no other subclasses within the hierarchy has a primary key value the same as a primary key value for the subclass being checked.

In the example SQL this is achieved by creating a trigger for each subclass. If the SQL implementation supports ASSERTIONs, they can be used instead.

6 Example SQL for instantiation of the metamodel

The complete set of SQL statements needed to provide this example SQL instantiation is at Annex A.

The statements are provided in the following order:

a) CREATE TABLE statements to create the tables to represent the two complex datatypes, the Phone_Number class and the Postal_Address class.

b) CREATE TABLE statements to create the tables to represent the basic classes, with additional characteristic tables to represent multi-valued attributes where needed.

c) CREATE TABLE statements to create the tables to represent the <<type>> classes. These tables are created in an order that allows for the instantiation of specialization of the Identified_Item class.

d) CREATE TABLE statements to create the tables to represent the remaining classes. These tables are created in alphabetical order.

e) CREATE TABLE statements to create the tables to represent the association classes. These tables are also created in alphabetical order. Additional characteristic tables are created to represent multi-valued attributes where needed

f) ALTER TABLE statements to add columns and foreign keys as necessary to the already created tables to add the attributes that were omitted when the tables were created because the datatype of the attribute is another class.

g) ALTER TABLE statements to add columns and foreign keys as necessary to the already created tables to instantiate subclassing where appropriate.

h) ALTER TABLE statements to add columns and foreign keys to the already created tables for those classes in the metamodel that can be 'typed' so that they can be identified, designated or classified.

i) CREATE TABLE statements to create the tables to represent the metamodel associations.

j) CREATE TRIGGER statements to create the triggers to ensure 'disjointness' within specialization hierarchies.

This set of SQL statements does not provide the most optimal instantiation of a database for a metadata registry, but it does provide an instantiation that can easily be traced back to the metamodel.

Annex A
(informative)

Example SQL to instantiate the ISO/IEC 11179-3 metamodel

```
/* ------------------------------------------------------------------- */
/* example ISO/IEC 9075 SQL statements to instantiate the              */
/* ISO/IEC 11179-3:2013 metamodel as an SQL database                   */
/* ------------------------------------------------------------------- */


/* ------------------------------------------------------------------- */
/* create tables for 11179-3 complex datatypes                         */
/* ------------------------------------------------------------------- */

CREATE TABLE cdt_phone_number
  (
    /* surrogate primary key */
    phone_number_id                    INTEGER             PRIMARY KEY  ,
    /* columns to represent attributes */
    international_numbering_plan_prefix  CHARACTER VARYING(255)          ,
    country_code                       CHARACTER VARYING(255)          ,
    city_code                          CHARACTER VARYING(255)          ,
    local_number                       CHARACTER VARYING(255)  NOT NULL ,
    extension                          CHARACTER VARYING(255)
  );

CREATE TABLE cdt_postal_address
  (
    /* surrogate primary key */
    postal_address_id                  INTEGER             PRIMARY KEY  ,
    /* columns to represent attributes */
    sub_building_name                  CHARACTER VARYING(255)          ,
    building_name                      CHARACTER VARYING(255)          ,
    throughfare                        CHARACTER VARYING(255)          ,
    dependent_locality                 CHARACTER VARYING(255)          ,
    post_town                          CHARACTER VARYING(255)          ,
    region                             CHARACTER VARYING(255)          ,
    postcode                           CHARACTER VARYING(255)          ,
    country                            CHARACTER VARYING(255)
  );


/* ------------------------------------------------------------------- */
/* create tables for 11179-3 basic classes                             */
/* ------------------------------------------------------------------- */

CREATE TABLE cls_organization
  (
    /* surrogate primary key */
    organization_id                    INTEGER             PRIMARY KEY  ,
    /* columns to represent attributes */
    mail_address                       INTEGER
                    REFERENCES cdt_postal_address (postal_address_id)   ,
    uri                                CHARACTER VARYING(250)
    /* the multi-valued attributes name, email_address and phone_number are */
    /* instantiated as the characteristic tables mva_organization_names, */
    /* mva_organization_email_addresses and mva_organization_phone_numbers */
  );
```

```
CREATE TABLE mva_organization_names
  (
    /* column to represent multi-valued attribute */
    name                                CHARACTER VARYING(2500)          ,
    /* identification of the organization this name belongs to */
    owning_organization_id              INTEGER
                          REFERENCES cls_organization (organization_id)  ,
    PRIMARY KEY (name, owning_organization_id)
  );

CREATE TABLE mva_organization_email_addresses
  (
    /* column to represent multi-valued attribute */
    email_address                       CHARACTER VARYING(250)           ,
    /* identification of the organization this email address belongs to */
    owning_organization_id              INTEGER
                          REFERENCES cls_organization (organization_id)  ,
    PRIMARY KEY (email_address, owning_organization_id)
  );

CREATE TABLE mva_organization_phone_numbers
  (
    /* column to represent multi-valued attribute */
    phone_number                        INTEGER
                          REFERENCES cdt_phone_number (phone_number_id)  ,
    /* identification of the organization this phone number belongs to */
    owning_organization_id              INTEGER
                          REFERENCES cls_organization (organization_id)  ,
    PRIMARY KEY (phone_number, owning_organization_id)
  );

CREATE TABLE cls_role
  (
    /* surrogate primary key */
    role_id                             INTEGER            PRIMARY KEY   ,
    /* columns to represent attributes */
    title                               CHARACTER VARYING(2500)          ,
    mail_address                        INTEGER
                      REFERENCES cdt_postal_address (postal_address_id)
    /* the multi-valued attributes email_address and phone_number are */
    /* instantiated as the characteristic tables mva_role_email_addresses */
    /* and mva_role_phone_numbers */
  );

CREATE TABLE mva_role_email_addresses
  (
    /* column to represent multi-valued attribute */
    email_address                        CHARACTER VARYING(255)          ,
    /* identification of the role this email address belongs to */
    owning_role_id                      INTEGER
                                          REFERENCES cls_role (role_id)  ,
    PRIMARY KEY (email_address, owning_role_id)
  );

CREATE TABLE mva_role_phone_numbers
  (
    /* column to represent multi-valued attribute */
    phone_number                        INTEGER
                          REFERENCES cdt_phone_number (phone_number_id)  ,
    /* identification of the role this phone number belongs to */
    owning_role_id                      INTEGER
                                          REFERENCES cls_role (role_id)  ,
    PRIMARY KEY (phone_number, owning_role_id)
  );
```

```
CREATE TABLE cls_individual
  (
    /* surrogate primary key */
    individual_id                       INTEGER           PRIMARY KEY  ,
    /* columns to represent attributes */
    name                                CHARACTER VARYING(255)         ,
    title                               CHARACTER VARYING(255)         ,
    mail_address                        INTEGER
                      REFERENCES cdt_postal_address (postal_address_id)
    /* the multi-valued attributes email_address, phone_number and role are */
    /* instantiated as the characteristic tables mva_individual_email_addresses, */
    /* mva_individual_phone_numbers and mva_individual_roles */
  );

CREATE TABLE mva_individual_email_addresses
  (
    /* column to represent multi-valued attribute */
    email_address                       CHARACTER VARYING(255)         ,
    /* identification of the individual this email address belongs to */
    owning_individual_id                INTEGER
                              REFERENCES cls_individual (individual_id)  ,
    PRIMARY KEY (email_address, owning_individual_id)
  );

CREATE TABLE mva_individual_phone_numbers
  (
    /* column to represent multi-valued attribute */
    phone_number                        INTEGER
                         REFERENCES cdt_phone_number (phone_number_id)  ,
    /* identification of the individual this phone number belongs to */
    owning_individual_id                INTEGER
                              REFERENCES cls_individual (individual_id)  ,
    PRIMARY KEY (phone_number, owning_individual_id)
  );

CREATE TABLE mva_individual_roles
  (
    /* column to represent multi-valued attribute */
    role                                INTEGER
                                         REFERENCES cls_role (role_id)  ,
     /* identification of the individual this phone number belongs to */
    owning_individual_id                INTEGER
                              REFERENCES cls_individual (individual_id)  ,
    PRIMARY KEY (role, owning_individual_id)
  );

CREATE TABLE cls_contact
  (
    /* surrogate primary key */
    contact_id                          INTEGER             PRIMARY KEY  ,
    /* columns to represent attributes */
    individual                          INTEGER
                              REFERENCES cls_individual (individual_id)  ,
    organization                        INTEGER
                         REFERENCES cls_organization (organization_id)  ,
    role                                INTEGER
                                         REFERENCES cls_role (role_id)
  );
```

```
CREATE TABLE cls_language_identification
  (
    /* surrogate primary key */
    language_identification_id           INTEGER             PRIMARY KEY  ,
    /* columns to represent attributes */
    language_identifier                  CHARACTER VARYING(255)  NOT NULL ,
    script_identifier                    CHARACTER VARYING(255)           ,
    region_identifier                    CHARACTER VARYING(255)           ,
    private_use_qualifier                CHARACTER VARYING(255)
    /* the multi-valued attributes variant_identifiers and extension_identifiers */
    /* are instantiated as the characteristic tables */
    /* mva_language_identification_variant_identifiers and */
    /* mva_language_identification_extension_identifiers */
  );

CREATE TABLE mva_language_identification_variant_identifiers
  (
    /* column to represent multi-valued attribute */
    variant_identifier                   CHARACTER VARYING(255)           ,
    /* column to represent the priority of this multi-valued attribute */
    priority                             INTEGER                          ,
    /* identification of the language identification this */
    /* variant identifier belongs to */
    owning_language_identification_id      INTEGER
      REFERENCES cls_language_identification (language_identification_id)  ,
    PRIMARY KEY (variant_identifier, priority, owning_language_identification_id)
  );

CREATE TABLE mva_language_identification_extension_identifiers
  (
    /* column to represent multi-valued attribute */
    extension_identifier                 CHARACTER VARYING(255)           ,
    /* column to represent the priority of this multi-valued attribute */
    priority                             INTEGER                          ,
    /* identification of the language identification this */
    /* extension identifier belongs to */
    owning_language_identification_id      INTEGER
      REFERENCES cls_language_identification (language_identification_id)  ,
    PRIMARY KEY (extension_identifier, priority, owning_language_identification_id)
  );

CREATE TABLE cls_document_type
  (
    /* surrogate primary key */
    document_type_id                     INTEGER            PRIMARY KEY  ,
    /* columns to represent attributes */
    identifier                           CHARACTER VARYING(255)          ,
    description                          CHARACTER VARYING(255)          ,
    scheme_reference                     CHARACTER VARYING(2500)         ,
    /* constraint to check that document can be */
    /* identified or described */
    CONSTRAINT document_type_identification
      CHECK
        ( ( identifier IS NOT NULL ) OR ( description IS NOT NULL ) )
  );

CREATE TABLE cls_reference_document
  (
    /* surrogate primary key */
    reference_document_id                INTEGER             PRIMARY KEY  ,
    /* columns to represent attributes */
    identifier                           CHARACTER VARYING(255)           ,
    document_type                        INTEGER
                       REFERENCES cls_document_type (document_type_id)    ,
    notation                             CHARACTER VARYING(255)           ,
    title                                CHARACTER VARYING(2500)          ,
    provider                             INTEGER
                          REFERENCES cls_organization (organization_id)   ,
    uri                                  CHARACTER VARYING(255)
```

```
  );

CREATE TABLE cls_registration_authority_identifier
  (
    /* surrogate primary key */
    registration_authority_identifier_id
                                    INTEGER             PRIMARY KEY  ,
    /* columns to represent attributes */
    international_code_designator   CHARACTER VARYING(255)          ,
    organization_identifier         CHARACTER VARYING(255)          ,
    organization_part_identifier    CHARACTER VARYING(255)          ,
    organization_part_identifier_source CHARACTER VARYING(255)      ,
    /* constraint to check that if the organization_part_identifier */
    /* is present, then the organization_part_identifier_source     */
    /* shall be present                                             */
    CONSTRAINT organization_part_identifier_source_presence
      CHECK
       ( ( ( organization_part_identifier is NOT NULL )
           AND ( organization_part_identifier_source IS NOT NULL ) ) OR
         ( ( organization_part_identifier is NULL )
           AND ( organization_part_identifier_source IS NULL ) ) )
  );


/* ------------------------------------------------------------------- */
/* create tables for 11179-3 stereotypes                               */
/* ------------------------------------------------------------------- */

CREATE TABLE typ_identified_item
  (
    /* surrogate primary key */
    identified_item_id              INTEGER             PRIMARY KEY
  );

CREATE TABLE typ_registered_item
/* subclass of typ_identified_item */
  (
    /* inherited surrogate primary key */
    identified_item_id              INTEGER             PRIMARY KEY  ,
    /* foreign key to instantiate subclass */
    FOREIGN KEY (identified_item_id)
      REFERENCES typ_identified_item (identified_item_id)
  );

CREATE TABLE typ_administered_item
/* subclass of typ_registered_item */
  (
    /* inherited surrogate primary key */
    identified_item_id              INTEGER              PRIMARY KEY  ,
    /* columns to represent attributes */
    version                         CHARACTER VARYING(255)          ,
    creation_date                   TIMESTAMP                       ,
    last_change_date                TIMESTAMP                       ,
    change_description              CHARACTER VARYING(255)          ,
    explanatory_comment             CHARACTER VARYING(255)          ,
    origin                          CHARACTER VARYING(255)          ,
    /* column check constraint for enumeration of object */
    CONSTRAINT administered_item_origin_enumeration
        CHECK ( origin IN
              ( 'document'  , 'project' , 'discipline' , 'model' ) )      ,
    /* foreign key to instantiate subclass */
    FOREIGN KEY (identified_item_id)
      REFERENCES typ_registered_item (identified_item_id)
  );
```

```
CREATE TABLE typ_attached_item
/* subclass of typ_registered_item */
  (
    /* inherited surrogate primary key */
    identified_item_id                  INTEGER            PRIMARY KEY  ,
    /* foreign key to instantiate subclass */
    FOREIGN KEY (identified_item_id)
      REFERENCES typ_registered_item (identified_item_id)
  );

CREATE TABLE typ_classifiable_item
  (
    /* surrogate primary key */
    classifiable_item_id                INTEGER            PRIMARY KEY
  );

CREATE TABLE typ_designatable_item
  (
    /* surrogate primary key */
    designatable_item_id                INTEGER            PRIMARY KEY
  );


/* ------------------------------------------------------------------ */
/* create tables for other 11179-3 classes                            */
/* ------------------------------------------------------------------ */

CREATE TABLE cls_assertion
  (
    /* surrogate primary key */
    assertion_id                        INTEGER             PRIMARY KEY  ,
    /* column to represent attribute */
    formula                             CHARACTER VARYING(255)
  );

CREATE TABLE cls_binary_relation
/* subclass of cls_relation */
  (
    /* inherited surrogate primary key */
    relation_id                         INTEGER             PRIMARY KEY  ,
    /* columns to represent attributes */
    reflexivity                         CHARACTER VARYING(255)           ,
    symmetry                            CHARACTER VARYING(255)           ,
    transitivity                        CHARACTER VARYING(255)           ,
    /* column check constraint for enumeration of reflexivity */
    CONSTRAINT binary_relation_reflexivity_enumeration
        CHECK ( reflexivity IN
              ( 'reflexive' , 'irreflexive' , 'antireflexive' ) )        ,
    /* column check constraint for enumeration of symmetry */
    CONSTRAINT binary_relation_symmetry_enumeration
        CHECK ( symmetry IN
              ( 'symmetric' , 'asymmetric' , 'antisymmetric' ) )         ,
    /* column check constraint for enumeration of transitivity */
    CONSTRAINT binary_relation_transitivity_enumeration
        CHECK ( transitivity IN
              ( 'transitive' , 'intransitive' , 'antitransitive' ) )
  );

CREATE TABLE cls_concept
  (
    /* surrogate primary key */
    concept_id                          INTEGER             PRIMARY KEY
  );
```

```
CREATE TABLE cls_concept_system
  (
    /* surrogate primary key */
    concept_system_id                   INTEGER             PRIMARY KEY  ,
    /* column to represent attribute */
    notation                            CHARACTER VARYING(255)
  );

CREATE TABLE cls_conceptual_domain
  (
    /* surrogate primary key */
   conceptual_domain_id                 INTEGER           PRIMARY KEY
    /* the column and foreign key for the dimensionality attribute */
    /* is added using alter table statements */
  );

CREATE TABLE cls_context
  (
    /* surrogate primary key */
    context_id                          INTEGER             PRIMARY KEY
  );

CREATE TABLE cls_data_element
  (
    /* surrogate primary key */
    data_element_id                     INTEGER             PRIMARY KEY  ,
    /* column to represent attribute */
    precision                           INTEGER
  );

CREATE TABLE cls_data_element_concept
  (
    /* surrogate primary key */
    data_element_concept_id             INTEGER             PRIMARY KEY
  );

CREATE TABLE cls_data_element_derivation
  (
    /* surrogate primary key */
    data_element_derivation_id          INTEGER             PRIMARY KEY
  );

CREATE TABLE cls_data_element_example
  (
    /* surrogate primary key */
    data_element_example_id             INTEGER             PRIMARY KEY
    /* the multi-valued attribute example_item is instantiated */
    /* as the characteristic table mva_data_element_example_items */
  );

CREATE TABLE mva_data_element_example_items
  (
    /* column to represent multi-valued attribute */
    example_item                       CHARACTER VARYING(2500)          ,
    /* identification of the data element example this */
    /* example item belongs to */
    owning_data_element_example_id      INTEGER
        REFERENCES cls_data_element_example_id (data_element_example_id)  ,
    PRIMARY KEY (example_item, owning_data_element_example_id)
  );
```

```
CREATE TABLE cls_datatype
  (
    /* surrogate primary key */
    datatype_id                         INTEGER             PRIMARY KEY  ,
    /* columns to represent attributes */
    name                                CHARACTER VARYING(255)           ,
    description                         CHARACTER VARYING(2500)          ,
    annotation                          CHARACTER VARYING(2500)
    /* the column and foreign key for the scheme_reference attribute */
    /* are added using alter table statements */
  );

CREATE TABLE cls_definition
  (
    /* surrogate primary key */
    definition_id                       INTEGER             PRIMARY KEY  ,
    /* column to represent attribute */
    text                                CHARACTER VARYING(2500)
    /* the columns and foreign keys for the language and source attributes */
    /* are added using alter table statements */
  );

CREATE TABLE cls_derivation_rule
  (
    /* surrogate primary key */
    derivation_rule_id                  INTEGER             PRIMARY KEY  ,
    /* columns to represent attributes */
    specification                       CHARACTER VARYING(2500)          ,
    notation                            CHARACTER VARYING(255)
  );

CREATE TABLE cls_described_conceptual_domain
/* subclass of cls_conceptual_domain */
  (
    /* inherited surrogate primary key */
    conceptual_domain_id                INTEGER             PRIMARY KEY  ,
    /* column to represent attribute */
    description                         CHARACTER VARYING(2500)
  );

CREATE TABLE cls_described_value_domain
/* subclass of cls_value_domain */
  (
    /* inherited surrogate primary key */
    value_domain_id                     INTEGER             PRIMARY KEY  ,
    /* column to represent attribute */
    description                         CHARACTER VARYING(2500)
  );

CREATE TABLE cls_designation
  (
    /* surrogate primary key */
    designation_id                      INTEGER             PRIMARY KEY  ,
    /* column to represent attribute */
    sign                                CHARACTER VARYING(2500)
  );

CREATE TABLE cls_dimensionality
  (
    /* surrogate primary key */
    dimensionality_id                   INTEGER             PRIMARY KEY  ,
    /* columns to represent native attributes */
    coordinate_indicator                BOOLEAN
  );
```

```
CREATE TABLE cls_enumerated_conceptual_domain
/* subclass of cls_conceptual_domain */
  (
    /* inherited surrogate primary key */
    conceptual_domain_id                INTEGER            PRIMARY KEY
  );

CREATE TABLE cls_enumerated_value_domain
/* subclass of cls_value_domain */
  (
    /* inherited surrogate primary key */
    value_domain_id                     INTEGER            PRIMARY KEY
  );

CREATE TABLE cls_link
/* subclass of cls_assertion */
  (
    /* inherited surrogate primary key */
    assertion_id                        INTEGER            PRIMARY KEY
  );

CREATE TABLE cls_link_end
  (
    /* surrogate primary key */
    link_end_id                         INTEGER            PRIMARY KEY
  );

CREATE TABLE cls_measure_class
  (
    /* surrogate primary key */
    measure_class_id                   INTEGER          PRIMARY KEY
  );

CREATE TABLE cls_namespace
  (
    /* surrogate primary key */
    namespace_id                        INTEGER            PRIMARY KEY  ,
    /* columns to represent native attributes */
    one_name_per_item_indicator         BOOLEAN                         ,
    one_item_per_name_indicator         BOOLEAN                         ,
    mandatory_naming_convention_indicator
                                        BOOLEAN                         ,
    shorthand_prefix                    CHARACTER VARYING(255)          ,
    scheme_reference                    CHARACTER VARYING(2500)
    /* the column and foreign key for the naming_authority attribute */
    /* are added using alter table statements */
  );

CREATE TABLE cls_naming_convention
  (
    /* surrogate primary key */
    naming_convention_id                INTEGER            PRIMARY KEY  ,
    /* columns to represent native attributes */
    scope_rule                          CHARACTER VARYING(2500)         ,
    authority_rule                      CHARACTER VARYING(2500)         ,
    semantic_rule                       CHARACTER VARYING(2500)         ,
    syntactic_rule                      CHARACTER VARYING(2500)         ,
    lexical_rule                        CHARACTER VARYING(2500)
  );

CREATE TABLE cls_object_class
  (
    /* surrogate primary key */
    object_class_id                     INTEGER            PRIMARY KEY
  );
```

```
CREATE TABLE cls_permissible_value
  (
    /* surrogate primary key */
    permissible_value_id                INTEGER             PRIMARY KEY  ,
    /* the permitted_value attribute has datatype 'value' which can be of any */
    /* datatype - this is instantiated by a different column for each datatype */
    /* and a check constraint to ensure that only one 'datatype' has a value */
    boolean_permitted_value             BOOLEAN                          ,
    date_permitted_value                DATE                             ,
    datetime_permitted_value            TIMESTAMP                        ,
    integer_permitted_value             INTEGER                          ,
    lower_bound_natural_range_permitted_value
                                        INTEGER                          ,
    fixed_upper_bound_natural_range_permitted_value
                                        INTEGER
    many_upper_bound_natural_range_permitted_value
                                        CHARACTER(4)   DEFAULT 'many'    ,
    notation_permitted_value            CHARACTER VARYING(2500)          ,
    phone_number_permitted_value        INTEGER
                          REFERENCES cdt_phone_number (phone_number_id)  ,
    postal_address_permitted_value      INTEGER
                     REFERENCES cdt_postal_address (postal_address_id)   ,
    sign_permitted_value                CHARACTER VARYING(2500)          ,
    string_permitted_value              CHARACTER VARYING(255)           ,
    text_permitted_value                CHARACTER VARYING(2500)          ,
    /* columns to represent other attributes */
    begin_date                          DATE                             ,
    end_date                            DATE                             ,
    /* check constraint to manage datatype columns */
    CONSTRAINT datatype_of_permitted_value
      CHECK
       ( ( ( boolean_permitted_value is NOT NULL )
            AND ( date_permitted_value IS NULL )
            AND ( datetime_permitted_value IS NULL )
            AND ( integer_permitted_value IS NULL )
            AND ( lower_bound_natural_range_permitted_value IS NULL )
            AND ( fixed_upper_bound_natural_range_permitted_value IS NULL )
            AND ( many_upper_bound_natural_range_permitted_value IS NULL )
            AND ( phone_number_permitted_value IS NULL )
            AND ( postal_address_permitted_value IS NULL )
            AND ( sign_permitted_value IS NULL )
            AND ( string_permitted_value IS NULL )
            AND ( text_permitted_value IS NULL ) )
         OR
         ( ( boolean_permitted_value is NULL )
            AND ( date_permitted_value IS NOT NULL )
            AND ( datetime_permitted_value IS NULL )
            AND ( integer_permitted_value IS NULL )
            AND ( lower_bound_natural_range_permitted_value IS NULL )
            AND ( fixed_upper_bound_natural_range_permitted_value IS NULL )
            AND ( many_upper_bound_natural_range_permitted_value IS NULL )
            AND ( phone_number_permitted_value IS NULL )
            AND ( postal_address_permitted_value IS NULL )
            AND ( sign_permitted_value IS NULL )
            AND ( string_permitted_value IS NULL )
            AND ( text_permitted_value IS NULL ) )
         OR
         ( ( boolean_permitted_value is NULL )
            AND ( date_permitted_value IS NULL )
            AND ( datetime_permitted_value IS NOT NULL )
            AND ( integer_permitted_value IS NULL )
            AND ( lower_bound_natural_range_permitted_value IS NULL )
            AND ( fixed_upper_bound_natural_range_permitted_value IS NULL )
            AND ( many_upper_bound_natural_range_permitted_value IS NULL )
           AND ( phone_number_permitted_value IS NULL )
            AND ( postal_address_permitted_value IS NULL )
            AND ( sign_permitted_value IS NULL )
            AND ( string_permitted_value IS NULL )
            AND ( text_permitted_value IS NULL ) )
```

```
OR
( ( boolean_permitted_value is NULL )
    AND ( date_permitted_value IS NULL )
    AND ( datetime_permitted_value IS NOT NULL )
    AND ( integer_permitted_value IS NULL )
    AND ( lower_bound_natural_range_permitted_value IS NULL )
    AND ( fixed_upper_bound_natural_range_permitted_value IS NULL )
    AND ( many_upper_bound_natural_range_permitted_value IS NULL )
    AND ( phone_number_permitted_value IS NULL )
    AND ( postal_address_permitted_value IS NULL )
    AND ( sign_permitted_value IS NULL )
    AND ( string_permitted_value IS NULL )
    AND ( text_permitted_value IS NULL ) )
OR
( ( boolean_permitted_value is NULL )
    AND ( date_permitted_value IS NULL )
    AND ( datetime_permitted_value IS NULL )
    AND ( integer_permitted_value IS NOT NULL )
    AND ( fixed_upper_bound_natural_range_permitted_value IS NULL )
    AND ( many_upper_bound_natural_range_permitted_value IS NULL )
    AND ( phone_number_permitted_value IS NULL )
    AND ( postal_address_permitted_value IS NULL )
    AND ( sign_permitted_value IS NULL )
    AND ( string_permitted_value IS NULL )
    AND ( text_permitted_value IS NULL ) )
OR
( ( boolean_permitted_value is NULL )
    AND ( date_permitted_value IS NULL )
    AND ( datetime_permitted_value IS NULL )
    AND ( integer_permitted_value IS NULL )
    AND ( lower_bound_natural_range_permitted_value IS NOT NULL )
    AND ( fixed_upper_bound_natural_range_permitted_value IS NOT NULL )
    AND ( many_upper_bound_natural_range_permitted_value IS NULL )
    AND ( phone_number_permitted_value IS NULL )
    AND ( postal_address_permitted_value IS NULL )
    AND ( sign_permitted_value IS NULL )
    AND ( string_permitted_value IS NULL )
    AND ( text_permitted_value IS NULL ) )
OR
( ( boolean_permitted_value is NULL )
    AND ( date_permitted_value IS NULL )
    AND ( datetime_permitted_value IS NULL )
    AND ( integer_permitted_value IS NULL )
    AND ( lower_bound_natural_range_permitted_value IS NOT NULL )
    AND ( fixed_upper_bound_natural_range_permitted_value IS NULL )
    AND ( many_upper_bound_natural_range_permitted_value IS NOT NULL )
    AND ( phone_number_permitted_value IS NULL )
    AND ( postal_address_permitted_value IS NULL )
    AND ( sign_permitted_value IS NULL )
    AND ( string_permitted_value IS NULL )
    AND ( text_permitted_value IS NULL ) )
OR
( ( boolean_permitted_value is NULL )
    AND ( date_permitted_value IS NULL )
    AND ( datetime_permitted_value IS NULL )
    AND ( integer_permitted_value IS NULL )
    AND ( lower_bound_natural_range_permitted_value IS NULL )
    AND ( fixed_upper_bound_natural_range_permitted_value IS NULL )
    AND ( many_upper_bound_natural_range_permitted_value IS NULL )
    AND ( phone_number_permitted_value IS NOT NULL )
    AND ( postal_address_permitted_value IS NULL )
    AND ( sign_permitted_value IS NULL )
    AND ( string_permitted_value IS NULL )
    AND ( text_permitted_value IS NULL ) )
OR
( ( boolean_permitted_value is NULL )
    AND ( date_permitted_value IS NULL )
    AND ( datetime_permitted_value IS NULL )
    AND ( integer_permitted_value IS NULL )
    AND ( lower_bound_natural_range_permitted_value IS NULL )
    AND ( fixed_upper_bound_natural_range_permitted_value IS NULL )
```

```
            AND ( many_upper_bound_natural_range_permitted_value IS NULL )
            AND ( phone_number_permitted_value IS NULL )
            AND ( postal_address_permitted_value IS NOT NULL )
            AND ( sign_permitted_value IS NULL )
            AND ( string_permitted_value IS NULL )
            AND ( text_permitted_value IS NULL ) )
        OR
        ( ( boolean_permitted_value is NULL )
            AND ( date_permitted_value IS NULL )
            AND ( datetime_permitted_value IS NULL )
            AND ( integer_permitted_value IS NULL )
            AND ( lower_bound_natural_range_permitted_value IS NULL )
            AND ( fixed_upper_bound_natural_range_permitted_value IS NULL )
            AND ( many_upper_bound_natural_range_permitted_value IS NULL )
            AND ( phone_number_permitted_value IS NULL )
            AND ( postal_address_permitted_value IS NULL )
            AND ( sign_permitted_value IS NOT NULL )
            AND ( string_permitted_value IS NULL )
            AND ( text_permitted_value IS NULL ) )
        OR
        ( ( boolean_permitted_value is NULL )
            AND ( date_permitted_value IS NULL )
            AND ( datetime_permitted_value IS NULL )
            AND ( integer_permitted_value IS NULL )
            AND ( lower_bound_natural_range_permitted_value IS NULL )
            AND ( fixed_upper_bound_natural_range_permitted_value IS NULL )
            AND ( many_upper_bound_natural_range_permitted_value IS NULL )
            AND ( phone_number_permitted_value IS NULL )
            AND ( postal_address_permitted_value IS NULL )
            AND ( sign_permitted_value IS NULL )
            AND ( string_permitted_value IS NOT NULL )
            AND ( text_permitted_value IS NULL ) )
        OR
        ( ( boolean_permitted_value is NULL )
            AND ( date_permitted_value IS NULL )
            AND ( datetime_permitted_value IS NULL )
            AND ( integer_permitted_value IS NULL )
            AND ( lower_bound_natural_range_permitted_value IS NULL )
            AND ( fixed_upper_bound_natural_range_permitted_value IS NULL )
            AND ( many_upper_bound_natural_range_permitted_value IS NULL )
            AND ( phone_number_permitted_value IS NULL )
            AND ( postal_address_permitted_value IS NULL )
            AND ( sign_permitted_value IS NULL )
            AND ( string_permitted_value IS NULL )
            AND ( text_permitted_value IS NOT NULL ) ) )                ,
   CONSTRAINT permissible_value_cannot_have_an_end_date_without_a_begin_date
      CHECK
       ( ( ( begin_date IS NULL ) AND ( end_date IS NULL ) OR
         ( ( begin_date IS NOT NULL ) AND ( end_date IS NULL ) OR
         ( ( begin_date IS NOT NULL ) AND ( end_date IS NOT NULL ) ) )
  );

CREATE TABLE cls_property
  (
    /* surrogate primary key */
    property_id                         INTEGER             PRIMARY KEY
  );

CREATE TABLE cls_registrar
/* subclass of contact */
  (
    /* inherited surrogate primary key */
    contact_id                          INTEGER             PRIMARY KEY  ,
    /* columns to represent native attributes */
    identifier                          CHARACTER VARYING(255) NOT NULL  ,
  );
```

```
CREATE TABLE cls_registration_authority
/* subclass of cls_organization */
  (
    /* inherited surrogate primary key */
    organization_id                     INTEGER             PRIMARY KEY  ,
    /* columns to represent attributes */
    registration_authority_identifier   INTEGER             UNIQUE
                  REFERENCES cls_registration_authority_identifier
                               (registration_authority_identifier_id)  ,
    document_language_identifier        INTEGER
                  REFERENCES cls_language_identification
                                       (language_identification_id)
  );

CREATE TABLE cls_registration_state
  (
    /* surrogate primary key */
    registration_state_id               INTEGER             PRIMARY KEY  ,
    /* columns to represent attributes */
    registration_status                 CHARACTER VARYING(255)           ,
    effective_date                      TIMESTAMP                        ,
    until_date                          TIMESTAMP                        ,
    administrative_note                 CHARACTER VARYING(2500)          ,
    unresolved_issue                    CHARACTER VARYING(2500)          ,
    administrative_status               CHARACTER VARYING(255)           ,
    previous_state                      INTEGER         UNIQUE
                  REFERENCES cls_registration_state (registration_state_id)
      /* this reflexive association to registration is one-to-one */
  );

CREATE TABLE cls_registry_specification
  (
    /* surrogate primary key */
    registry_specification_id           INTEGER             PRIMARY KEY  ,
    /* columns to represent native attributes */
    name                                CHARACTER VARYING(2500)          ,
    web_address                         CHARACTER VARYING(255)           ,
    standard                            CHARACTER VARYING(255)           ,
    conformance_level                   CHARACTER VARYING(255)           ,
    character_repertoire                CHARACTER VARYING(255)           ,
    reference_document_identifier_form  CHARACTER VARYING(255)           ,
    primary_language                    INTEGER
                  REFERENCES cls_language_identification
                                       (language_identification_id)  ,
    /* the column and foreign keys for the representation_class_scheme */
    /* and context attributes are added using alter table statements */
    comment                             CHARACTER VARYING(2500)
  );
```

```
CREATE TABLE cls_relation
/* subclass of cls_concept */
  (
    /* surrogate primary key */
    relation_id                          INTEGER             PRIMARY KEY  ,
    /* columns to represent attribute with natural range datatype*/
    arity_lower_bound_natural_range      INTEGER                          ,
    arity_fixed_upper_bound_natural_range
                                         INTEGER                          ,
    arity_many_upper_bound_natural_range
                                         CHARACTER(4)   DEFAULT 'many'    ,
    CONSTRAINT relation_arity_natural_range_check
      CHECK
       ( ( ( arity_lower_bound_natural_range IS NULL )
           AND ( arity_fixed_upper_bound_natural_range IS NULL )
           AND ( arity_many_upper_bound_natural_range IS NULL ) )
         OR
         ( ( arity_lower_bound_natural_range IS NOT NULL )
           AND ( arity_fixed_upper_bound_natural_range IS NULL )
           AND ( arity_many_upper_bound_natural_range IS NULL ) )
         OR
         ( ( arity_lower_bound_natural_range IS NOT NULL )
           AND ( arity_fixed_upper_bound_natural_range IS NOT NULL )
           AND ( arity_many_upper_bound_natural_range IS NULL ) )
         OR
         ( ( arity_lower_bound_natural_range IS NOT NULL )
           AND ( arity_fixed_upper_bound_natural_range IS NULL )
           AND ( arity_many_upper_bound_natural_range IS NOT NULL ) ) )
  );

CREATE TABLE cls_relation_role
/* subclass of cls_concept */
  (
    /* surrogate primary key */
    relation_role_id                     INTEGER             PRIMARY KEY  ,
    /* columns to represent attributes */
    multiplicity_lower_bound_natural_range
                                         INTEGER                          ,
    multiplicity_fixed_upper_bound_natural_range
                                         INTEGER                          ,
    multiplicity_many_upper_bound_natural_range
                                         CHARACTER(4)   DEFAULT 'many'    ,
    ordinal                              INTEGER                          ,
    CONSTRAINT relation_role_multiplicity_natural_range_check
      CHECK
       ( ( ( multiplicity_lower_bound_natural_range IS NULL )
           AND ( multiplicity_fixed_upper_bound_natural_range IS NULL )
           AND ( multiplicity_many_upper_bound_natural_range IS NULL ) )
         OR
         ( ( multiplicity_lower_bound_natural_range IS NOT NULL )
           AND ( multiplicity_fixed_upper_bound_natural_range IS NULL )
           AND ( multiplicity_many_upper_bound_natural_range IS NULL ) )
         OR
         ( ( multiplicity_lower_bound_natural_range IS NOT NULL )
           AND ( multiplicity_fixed_upper_bound_natural_range IS NOT NULL )
           AND ( multiplicity_many_upper_bound_natural_range IS NULL ) )
         OR
         ( ( multiplicity_lower_bound_natural_range IS NOT NULL )
           AND ( multiplicity_fixed_upper_bound_natural_range IS NULL )
           AND ( multiplicity_many_upper_bound_natural_range IS NOT NULL ) ) )
  );
```

```
CREATE TABLE cls_scoped_identifier
  (
    /* surrogate primary key */
    scoped_identifier_id                INTEGER              PRIMARY KEY  ,
    /* columns to represent attributes */
    identifier                          CHARACTER VARYING(255)            ,
    full_expansion                      CHARACTER VARYING(255)            ,
    short_expansion                     CHARACTER VARYING(255)
  );

CREATE TABLE cls_slot
  (
    /* surrogate primary key */
    slot_id                             INTEGER              PRIMARY KEY  ,
    /* columns to represent attributes */
    name                                CHARACTER VARYING(255)            ,
    type                                CHARACTER VARYING(255)
    /* the multi-valued attribute value is instantiated */
    /* as the characteristic table mva_slot_values */
  );

CREATE TABLE mva_slot_values
  (
    /* column to represent multi-valued attribute */
    Value                               CHARACTER VARYING(255)            ,
    /* column to represent the priority of this multi-valued attribute */
    priority                            INTEGER                           ,
    /* identification of the slot this value belongs to */
    owning_slot_id                      INTEGER
                                          REFERENCES cls_slot (slot_id)   ,
    PRIMARY KEY (value, priority, owning_lslot_id)
  );

CREATE TABLE cls_stewardship_record
  (
    /* surrogate primary key */
    stewardship_record_id               INTEGER              PRIMARY KEY  ,
    /* columns to represent attributes */
    organization                        INTEGER
                          REFERENCES cls_organization (organization_id)   ,
    contact                             INTEGER
                                    REFERENCES cls_contact (contact_id)
  );

CREATE TABLE cls_submission_record
  (
    /* surrogate primary key */
    submission_record_id                INTEGER              PRIMARY KEY  ,
    /* columns to represent attributes */
    organization                        INTEGER
                          REFERENCES cls_organization (organization_id)   ,
    contact                             INTEGER
                                    REFERENCES cls_contact (contact_id)
  ) ;

CREATE TABLE cls_unit_of_measure
  (
    /* surrogate primary key */
    unit_of_measure_id                  INTEGER               PRIMARY KEY
  );
```

```
CREATE TABLE cls_value_domain
  (
    /* surrogate primary key */
    value_domain_id                     INTEGER               PRIMARY KEY  ,
    /* columns to represent attributes */
    /* the column and foreign key for the datatype */
    /* attribute are added using alter table statements */
    format                              CHARACTER VARYING(255)             ,
    maximum_character_quantity          INTEGER
    /* the column and foreign key for the unit_of_measure */
    /* attribute are added using alter table statements */
  );

CREATE TABLE cls_value_meaning
  (
    /* surrogate primary key */
    value_meaning_id                    INTEGER               PRIMARY KEY  ,
    /* columns to represent native attributes */
    begin_date                          DATE                               ,
    end_date                            DATE                               ,
    CONSTRAINT value_meaning_cannot_have_an_end_date_without_a_begin_date
      CHECK
       ( ( ( begin_date IS NULL ) AND ( end_date IS NULL ) ) OR
         ( ( begin_date IS NOT NULL ) AND ( end_date IS NULL ) ) OR
         ( ( begin_date IS NOT NULL ) AND ( end_date IS NOT NULL ) ) )
  );


/* ------------------------------------------------------------------- */
/* create tables (and alter table statements) to instantiate the 11179-3 */
/* association classes                                                 */
/* ------------------------------------------------------------------- */

CREATE TABLE asscls_classification
  (
    /* surrogate primary key */
    classification_id                   INTEGER             PRIMARY KEY ,
    /* columns and foreign keys for roles of the association class */
    classifier                          INTEGER                         ,
    classified_item                     INTEGER                         ,
    CONSTRAINT classifications_have_classifiers
      FOREIGN KEY (classifier)
        REFERENCES cls_concept (concept_id)                           ,
    CONSTRAINT classifications_have_classified_items
      FOREIGN KEY (classified_item)
        REFERENCES typ_classifiable_item (classifiable_item_id)       ,
   );

CREATE TABLE asscls_definition_context
  (
    definition_context_id               INTEGER            PRIMARY KEY  ,
    /* column to represent attribute */
    acceptability                       CHARACTER VARYING(255)          ,
    /* columns and foreign keys for roles of the association class */
    relevant_definition                 INTEGER                         ,
    scope                               INTEGER                         ,
    CONSTRAINT definition_contexts_have_relevant_definitions
      FOREIGN KEY (relevant_definition)
        REFERENCES cls_definition (definition_id)                     ,
    CONSTRAINT definition_contexts_have_scopes
      FOREIGN KEY (scope)
        REFERENCES cls_context (context_id)                           ,
    /* column check constraint for enumeration of */
    /* definition_acceptability */
    CONSTRAINT definition_context_acceptability_enumeration
      CHECK ( acceptability IN
            ( 'preferred'  , 'admitted'  , 'deprecated'  ,
              'obsolete'  , 'superseded' ) )
  );
```

```
CREATE TABLE asscls_designation_context
  (
    /* surrogate primary key */
    designation_context_id               INTEGER            PRIMARY KEY  ,
    /* column to represent attribute */
    acceptability                        CHARACTER VARYING(255)          ,
    /* columns and foreign keys for roles of the association class */
    relevant_definition                  INTEGER                         ,
    scope                                INTEGER                         ,
    CONSTRAINT designation_contexts_have_relevant_definitions
      FOREIGN KEY (relevant_definition)
        REFERENCES cls_definition (definition_id)                        ,
    CONSTRAINT designation_contexts_have_scopes
      FOREIGN KEY (scope)
        REFERENCES cls_context (context_id)                              ,
    /* column check constraint for enumeration of */
    /* designation_acceptability */
    CONSTRAINT designation_context_acceptability_enumeration
        CHECK ( acceptability IN
              ( 'preferred'  , 'admitted'  , 'deprecated'  ,
                'obsolete'  , 'superseded' ) )
  );

CREATE TABLE asscls_reference
  (
    /* surrogate primary key */
    reference_id                        INTEGER            PRIMARY KEY  ,
    /* column to represent attribute */
    reference_type                      CHARACTER VARYING(255)          ,
    /* columns and foreign keys for roles of the association class */
    linked_reference_document            INTEGER                        ,
    linked_registered_item               INTEGER                        ,
    CONSTRAINT references_have_linked_reference_documents
      FOREIGN KEY (linked_reference_document)
        REFERENCES cls_reference_document (reference_document_id)        ,
    CONSTRAINT references_have_linked_registered_items
      FOREIGN KEY (linked_registered_item)
        REFERENCES typ_registered_item (identified_item_id)
  );

CREATE TABLE asscls_registration
  (
    /* surrogate primary key */
    registration_id                     INTEGER            PRIMARY KEY  ,
    /* the column and foreign key for the registration_state attribute */
    /* are added using alter table statements */
  );


/* ------------------------------------------------------------------- */
/* add columns and foreign keys to the tables for 11179-3 classes and  */
/* subclasses for 'missing' attributes                                 */
/* ------------------------------------------------------------------- */

/* ------------------------------------------------------------------- */
/* add column and foreign key to cls_conceptual_domain                 */
/* to represent the dimensionality attribute                           */
/* ------------------------------------------------------------------- */

ALTER TABLE cls_conceptual_domain
  ADD COLUMN
      dimensionality                    INTEGER
    ;

ALTER TABLE cls_conceptual_domain
  ADD CONSTRAINT conceptual_domains_have_dimensionalities
    FOREIGN KEY (dimensionality)
      REFERENCES cls_dimensionality (dimensionality_id);
```

```
/* ------------------------------------------------------------------- */
/* add column and foreign key to cls_datatype                          */
/* to represent the scheme_reference attribute                         */
/* ------------------------------------------------------------------- */

ALTER TABLE cls_datatype
  ADD COLUMN
      scheme_reference                  INTEGER
    ;

ALTER TABLE cls_datatype
  ADD CONSTRAINT datatypes_have_scheme_references
    FOREIGN KEY (scheme_reference)
      REFERENCES cls_reference_document (reference_document_id);

/* ------------------------------------------------------------------- */
/* add column and foreign key to cls_definition                        */
/* to represent the language attribute                                 */
/* ------------------------------------------------------------------- */

ALTER TABLE cls_definition
  ADD COLUMN
      language                          INTEGER
    ;

ALTER TABLE cls_definition
  ADD CONSTRAINT definitions_have_languages
    FOREIGN KEY (language)
      REFERENCES cls_language_identification (language_identification_id);

/* ------------------------------------------------------------------- */
/* add column and foreign key to cls_definition                        */
/* to represent the source attribute                                   */
/* ------------------------------------------------------------------- */

ALTER TABLE cls_definition
  ADD COLUMN
      source                            INTEGER
    ;

ALTER TABLE cls_definition
  ADD CONSTRAINT definitions_have_sources
    FOREIGN KEY (source)
      REFERENCES cls_reference_document (reference_document_id);

/* ------------------------------------------------------------------- */
/* add column and foreign key to cls_namespace                         */
/* to represent the naming_authority attribute                         */
/* ------------------------------------------------------------------- */

ALTER TABLE cls_namespace
  ADD COLUMN
      naming_authority                  INTEGER
    ;

ALTER TABLE cls_namespace
  ADD CONSTRAINT namespaces_have_naming_authorities
    FOREIGN KEY (naming_authority)
      REFERENCES cls_organization (organization_id);

/* ------------------------------------------------------------------- */
/* add column and foreign key to cls_registry_specification            */
/* to represent the representation_class_scheme attribute              */
/* ------------------------------------------------------------------- */

ALTER TABLE cls_registry_specification
  ADD COLUMN
       representation_class_scheme      INTEGER          UNIQUE
    ;
```

```
ALTER TABLE cls_registry_specification
  ADD CONSTRAINT registry_specifications_have_representation_class_schemes
    FOREIGN KEY (representation_class_scheme)
      REFERENCES cls_concept_system (concept_system_id);

/* ------------------------------------------------------------------- */
/* add column and foreign key to cls_registry_specification            */
/* to represent the context attribute                                  */
/* ------------------------------------------------------------------- */

ALTER TABLE cls_registry_specification
  ADD COLUMN
      context                          INTEGER          UNIQUE
    ;

ALTER TABLE cls_registry_specification
  ADD CONSTRAINT registry_specifications_have_contexts
    FOREIGN KEY (context)
      REFERENCES cls_context (context_id);

/* ------------------------------------------------------------------- */
/* add column and foreign key to cls_value_domain                      */
/* to represent the datatype attribute                                 */
/* ------------------------------------------------------------------- */

ALTER TABLE cls_value_domain
  ADD COLUMN
      datatype                         INTEGER
    ;

ALTER TABLE cls_value_domain
  ADD CONSTRAINT value_domains_have_datatypes
    FOREIGN KEY (datatype)
      REFERENCES cls_datatype (datatype_id);

/* ------------------------------------------------------------------- */
/* add column and foreign key to cls_value_domain                      */
/* to represent the unit_of_measure attribute                          */
/* ------------------------------------------------------------------- */

ALTER TABLE cls_value_domain
  ADD COLUMN
      unit_of_measure                  INTEGER
    ;

ALTER TABLE cls_value_domain
  ADD CONSTRAINT value_domains_have_unit_of_measures
    FOREIGN KEY (unit_of_measure)
      REFERENCES cls_unit_of_measure (unit_of_measure_id);

/* ------------------------------------------------------------------- */
/* add column and foreign key to asscls_registration                   */
/* to represent the registration_state attribute                       */
/* ------------------------------------------------------------------- */

ALTER TABLE asscls_registration
  ADD COLUMN
      registration_state               INTEGER          UNIQUE
    ;

ALTER TABLE asscls_registration
  ADD CONSTRAINT registrations_have_registration_states
    FOREIGN KEY (registration_state)
      REFERENCES cls_registration_state (registration_state_id);


/* ------------------------------------------------------------------- */
/* add columns and foreign keys to the tables for 11179-3              */
/* classes to instantiate subclassing where appropriate                */
/* ------------------------------------------------------------------- */
```

```
/* ------------------------------------------------------------------- */
/* add column and foreign key to cls_binary_relation because           */
/* cls_binary_relation is a subclass of cls_relation                   */
/* ------------------------------------------------------------------- */

ALTER TABLE cls_binary_relation
  ADD COLUMN
      superclass_relation_id             INTEGER      NOT NULL    UNIQUE
    ;

ALTER TABLE cls_binary_relation
  ADD CONSTRAINT binary_relation_is_a_subclass_of_relation
    FOREIGN KEY (superclass_relation_id)
      REFERENCES cls_relation (relation_id);

/* ------------------------------------------------------------------- */
/* add column and foreign key to cls_conceptual_domain because         */
/* cls_conceptual_domain is a subclass of cls_concept                  */
/* ------------------------------------------------------------------- */

ALTER TABLE cls_conceptual_domain
  ADD COLUMN
      superclass_concept_id              INTEGER      NOT NULL    UNIQUE
    ;

ALTER TABLE cls_conceptual_domain
  ADD CONSTRAINT conceptual_domain_is_a_subclass_of_concept
    FOREIGN KEY (superclass_concept_id)
      REFERENCES cls_concept (concept_id);

/* ------------------------------------------------------------------- */
/* add column and foreign key to cls_data_element_concept because      */
/* cls_data_element_concept is a subclass of cls_concept               */
/* ------------------------------------------------------------------- */

ALTER TABLE cls_data_element_concept
  ADD COLUMN
      superclass_concept_id              INTEGER      NOT NULL    UNIQUE
    ;

ALTER TABLE cls_data_element_concept
  ADD CONSTRAINT data_element_concept_is_a_subclass_of_concept
    FOREIGN KEY (superclass_concept_id)
      REFERENCES cls_concept (concept_id);

/* ------------------------------------------------------------------- */
/* add foreign key to cls_described_conceptual_domain because          */
/* cls_described_conceptual_domain is a subclass of cls_conceptual_domain */
/* ------------------------------------------------------------------- */

ALTER TABLE cls_described_conceptual_domain
  ADD CONSTRAINT described_conceptual_domain_is_a_subclass_of_conceptual_domain
    FOREIGN KEY (conceptual_domain_id)
      REFERENCES cls_conceptual_domain (conceptual_domain_id);

/* ------------------------------------------------------------------- */
/* add foreign key to cls_described_value_domain because               */
/* cls_described_value_domain is a subclass of cls_value_domain        */
/* ------------------------------------------------------------------- */

ALTER TABLE cls_described_value_domain
  ADD CONSTRAINT described_value_domain_is_a_subclass_of_value_domain
    FOREIGN KEY (value_domain_id)
      REFERENCES cls_value_domain (value_domain_id);

/* ------------------------------------------------------------------- */
/* add column and foreign key to cls_dimensionality because            */
/* cls_dimensionality is a subclass of cls_concept                     */
/* ------------------------------------------------------------------- */
```

```
ALTER TABLE cls_dimensionality
  ADD COLUMN
      superclass_concept_id              INTEGER      NOT NULL    UNIQUE
    ;

ALTER TABLE cls_dimensionality
  ADD CONSTRAINT dimensionality_is_a_subclass_of_concept
    FOREIGN KEY (superclass_concept_id)
      REFERENCES cls_concept (concept_id);

/* ------------------------------------------------------------------------ */
/* add foreign key to cls_enumerated_conceptual_domain because              */
/* cls_enumerated_conceptual_domain is a subclass of cls_conceptual_domain */
/* ------------------------------------------------------------------------ */

ALTER TABLE cls_enumerated_conceptual_domain
  ADD CONSTRAINT enumerated_conceptual_domain_is_a_subclass_of_conceptual_domain
    FOREIGN KEY (conceptual_domain_id)
      REFERENCES cls_conceptual_domain (conceptual_domain_id);

/* ------------------------------------------------------------------------ */
/* add foreign key to cls_enumerated_value_domain because                   */
/* cls_enumerated_value_domain is a subclass of cls_value_domain            */
/* ------------------------------------------------------------------------ */

ALTER TABLE cls_enumerated_value_domain
  ADD CONSTRAINT enumerated_value_domain_is_a_subclass_of_value_domain
    FOREIGN KEY (value_domain_id)
      REFERENCES cls_value_domain (value_domain_id);

/* ------------------------------------------------------------------------ */
/* add foreign key to cls_link because                                      */
/* cls_link is a subclass of cls_assertion                                  */
/* ------------------------------------------------------------------------ */

ALTER TABLE cls_link
  ADD CONSTRAINT link_is_a_subclass_of_assertion
    FOREIGN KEY (assertion_id)
      REFERENCES cls_assertion (assertion_id);

/* ------------------------------------------------------------------------ */
/* add column and foreign key to cls_object_class because                   */
/* cls_object_class is a subclass of cls_concept                            */
/* ------------------------------------------------------------------------ */

ALTER TABLE cls_object_class
  ADD COLUMN
      superclass_concept_id              INTEGER      NOT NULL    UNIQUE
    ;

ALTER TABLE cls_object_class
  ADD CONSTRAINT object_class_is_a_subclass_of_concept
    FOREIGN KEY (superclass_concept_id)
      REFERENCES cls_concept (concept_id);

/* ------------------------------------------------------------------------ */
/* add column and foreign key to cls_property because                       */
/* cls_property is a subclass of cls_concept                                */
/* ------------------------------------------------------------------------ */

ALTER TABLE cls_property
  ADD COLUMN
      superclass_concept_id              INTEGER      NOT NULL    UNIQUE
    ;

ALTER TABLE cls_property
  ADD CONSTRAINT property_is_a_subclass_of_concept
    FOREIGN KEY (superclass_concept_id)
      REFERENCES cls_concept (concept_id);
```

```
/* ------------------------------------------------------------------- */
/* add foreign key to cls_registrar because                            */
/* cls_registrar is a subclass of cls_contact                          */
/* ------------------------------------------------------------------- */

ALTER TABLE cls_registrar
  ADD CONSTRAINT registrar_is_a_subclass_of_contact
    FOREIGN KEY (contact_id)
      REFERENCES cls_contact (contact_id);

/* ------------------------------------------------------------------- */
/* add column and foreign key to cls_relation because                  */
/* cls_relation is a subclass of cls_concept                           */
/* ------------------------------------------------------------------- */

ALTER TABLE cls_relation
  ADD COLUMN
      superclass_concept_id             INTEGER      NOT NULL    UNIQUE
    ;

ALTER TABLE cls_relation
  ADD CONSTRAINT relation_is_a_subclass_of_concept
    FOREIGN KEY (superclass_concept_id)
      REFERENCES cls_concept (concept_id);

/* ------------------------------------------------------------------- */
/* add column and foreign key to cls_relation_role because             */
/* cls_relation_role is a subclass of cls_concept                      */
/* ------------------------------------------------------------------- */

ALTER TABLE cls_relation_role
  ADD COLUMN
      superclass_concept_id             INTEGER      NOT NULL    UNIQUE
    ;

ALTER TABLE cls_relation_role
  ADD CONSTRAINT relation_role_is_a_subclass_of_concept
    FOREIGN KEY (superclass_concept_id)
      REFERENCES cls_concept (concept_id);

/* ------------------------------------------------------------------- */
/* add column and foreign key to cls_unit_of_measure because           */
/* cls_unit_of_measure is a subclass of cls_concept                    */
/* ------------------------------------------------------------------- */

ALTER TABLE cls_unit_of_measure
  ADD COLUMN
      superclass_concept_id             INTEGER      NOT NULL    UNIQUE
    ;

ALTER TABLE cls_unit_of_measure
  ADD CONSTRAINT unit_of_measure_is_a_subclass_of_concept
    FOREIGN KEY (superclass_concept_id)
      REFERENCES cls_concept (concept_id);

/* ------------------------------------------------------------------- */
/* add column and foreign key to cls_value_meaning because             */
/* cls_value_meaning is a subclass of cls_concept                      */
/* ------------------------------------------------------------------- */

ALTER TABLE cls_value_meaning
  ADD COLUMN
      superclass_concept_id             INTEGER      NOT NULL    UNIQUE
    ;

ALTER TABLE cls_value_meaning
  ADD CONSTRAINT value_meaning_is_a_subclass_of_concept
    FOREIGN KEY (superclass_concept_id)
      REFERENCES cls_concept (concept_id);
```

```
/* ------------------------------------------------------------------------ */
/* add columns and foreign keys to the tables for 11179-3 classes           */
/* because metadata items can be identified, classified and designated      */
/* ------------------------------------------------------------------------ */

/* ------------------------------------------------------------------------ */
/* add columns and foreign keys to cls_assertion because assertions         */
/* can be identified, classified and designated                             */
/* ------------------------------------------------------------------------ */

ALTER TABLE cls_assertion
  ADD COLUMN
      linked_identified_item_id          INTEGER                  UNIQUE
    ;

ALTER TABLE cls_assertion
  ADD COLUMN
    (
      linked_classifiable_item_id        INTEGER                  UNIQUE
    );

ALTER TABLE cls_assertion
  ADD COLUMN
      linked_designatable_item_id        INTEGER                  UNIQUE
    ;

ALTER TABLE cls_assertion
  ADD CONSTRAINT each_assertion_may_be_identified
    FOREIGN KEY (linked_identified_item_id)
      REFERENCES typ_identified_item (identified_item_id);

ALTER TABLE cls_assertion
  ADD CONSTRAINT each_assertion_may_be_classified
    FOREIGN KEY (linked_classifiable_item_id)
      REFERENCES typ_classifiable_item (classifiable_item_id);

ALTER TABLE cls_assertion
  ADD CONSTRAINT each_assertion_may_be_designated
    FOREIGN KEY (linked_designatable_item_id)
      REFERENCES typ_designatable_item (designatable_item_id);

/* ------------------------------------------------------------------------ */
/* add columns and foreign keys to asscls_classification because            */
/* classifications can be identified, classified and designated             */
/* ------------------------------------------------------------------------ */

ALTER TABLE asscls_classification
  ADD COLUMN
      linked_identified_item_id          INTEGER                  UNIQUE
    ;

ALTER TABLE asscls_classification
  ADD COLUMN
      linked_classifiable_item_id        INTEGER                  UNIQUE
    ;

ALTER TABLE asscls_classification
  ADD COLUMN
      linked_designatable_item_id        INTEGER                  UNIQUE
    ;

ALTER TABLE asscls_classification
  ADD CONSTRAINT each_classification_may_be_identified
    FOREIGN KEY (linked_identified_item_id)
      REFERENCES typ_identified_item (identified_item_id);
```

```
ALTER TABLE asscls_classification
  ADD CONSTRAINT each_classification_may_be_classified
    FOREIGN KEY (linked_classifiable_item_id)
      REFERENCES typ_classifiable_item (classifiable_item_id);

ALTER TABLE asscls_classification
  ADD CONSTRAINT each_classification_may_be_designated
    FOREIGN KEY (linked_designatable_item_id)
      REFERENCES typ_designatable_item (designatable_item_id);

/* ------------------------------------------------------------------- */
/* add columns and foreign keys to cls_concept because concepts        */
/* can be identified, classified and designated                        */
/* ------------------------------------------------------------------- */

ALTER TABLE cls_concept
  ADD COLUMN
      linked_identified_item_id          INTEGER                 UNIQUE
    ;

ALTER TABLE cls_concept
  ADD COLUMN
     linked_classifiable_item_id        INTEGER                UNIQUE
    ;

ALTER TABLE cls_concept
  ADD COLUMN
      linked_designatable_item_id        INTEGER                 UNIQUE
    ;

ALTER TABLE cls_concept
  ADD CONSTRAINT each_concept_may_be_identified
    FOREIGN KEY (linked_identified_item_id)
      REFERENCES typ_identified_item (identified_item_id);

ALTER TABLE cls_concept
  ADD CONSTRAINT each_concept_may_be_classified
    FOREIGN KEY (linked_classifiable_item_id)
      REFERENCES typ_classifiable_item (classifiable_item_id);

ALTER TABLE cls_concept
  ADD CONSTRAINT each_concept_may_be_designated
    FOREIGN KEY (linked_designatable_item_id)
      REFERENCES typ_designatable_item (designatable_item_id);

/* ------------------------------------------------------------------- */
/* add columns and foreign keys to cls_concept_system because concept  */
/* systems can be identified, classified and designated                */
/* ------------------------------------------------------------------- */

ALTER TABLE cls_concept_system
  ADD COLUMN
      linked_identified_item_id          INTEGER                 UNIQUE
    ;

ALTER TABLE cls_concept_system
  ADD COLUMN
      linked_classifiable_item_id        INTEGER                 UNIQUE
    ;

ALTER TABLE cls_concept_system
  ADD COLUMN
      linked_designatable_item_id        INTEGER                 UNIQUE
    ;

ALTER TABLE cls_concept_system
  ADD CONSTRAINT each_concept_system_may_be_identified
    FOREIGN KEY (linked_identified_item_id)
      REFERENCES typ_identified_item (identified_item_id);
```

```
ALTER TABLE cls_concept_system
  ADD CONSTRAINT each_concept_system_may_be_classified
    FOREIGN KEY (linked_classifiable_item_id)
      REFERENCES typ_classifiable_item (classifiable_item_id);

ALTER TABLE cls_concept_system
  ADD CONSTRAINT each_concept_system_may_be_designated
    FOREIGN KEY (linked_designatable_item_id)
      REFERENCES typ_designatable_item (designatable_item_id);

/* ------------------------------------------------------------------- */
/* add columns and foreign keys to cls_conceptual_domain because       */
/* conceptual domains can be identified, classified and designated     */
/* ------------------------------------------------------------------- */

ALTER TABLE cls_conceptual_domain
  ADD COLUMN
      linked_identified_item_id          INTEGER                  UNIQUE
    ;

ALTER TABLE cls_conceptual_domain
  ADD COLUMN

      linked_classifiable_item_id        INTEGER                  UNIQUE
    ;

ALTER TABLE cls_conceptual_domain
  ADD COLUMN
      linked_designatable_item_id        INTEGER                  UNIQUE
    ;

ALTER TABLE cls_conceptual_domain
  ADD CONSTRAINT each_conceptual_domain_may_be_identified
    FOREIGN KEY (linked_identified_item_id)
      REFERENCES typ_identified_item (identified_item_id);

ALTER TABLE cls_described_conceptual_domain
  ADD CONSTRAINT each_conceptual_domain_may_be_classified
    FOREIGN KEY (linked_classifiable_item_id)
      REFERENCES typ_classifiable_item (classifiable_item_id);

ALTER TABLE cls_described_conceptual_domain
  ADD CONSTRAINT each_conceptual_domain_may_be_designated
    FOREIGN KEY (linked_designatable_item_id)
      REFERENCES typ_designatable_item (designatable_item_id);

/* ------------------------------------------------------------------- */
/* add columns and foreign keys to cls_context because                 */
/* contexts can be identified, classified and designated               */
/* ------------------------------------------------------------------- */

ALTER TABLE cls_context
  ADD COLUMN
      linked_identified_item_id          INTEGER                  UNIQUE
    ;

ALTER TABLE cls_context
  ADD COLUMN
      linked_classifiable_item_id        INTEGER                  UNIQUE
    ;

ALTER TABLE cls_context
  ADD COLUMN
      linked_designatable_item_id        INTEGER                  UNIQUE
    ;

ALTER TABLE cls_context
  ADD CONSTRAINT each_context_may_be_identified
    FOREIGN KEY (linked_identified_item_id)
      REFERENCES typ_identified_item (identified_item_id);
```

```
ALTER TABLE cls_context
  ADD CONSTRAINT each_context_may_be_classified
    FOREIGN KEY (linked_classifiable_item_id)
      REFERENCES typ_classifiable_item (classifiable_item_id);

ALTER TABLE cls_context
  ADD CONSTRAINT each_context_may_be_designated
    FOREIGN KEY (linked_designatable_item_id)
      REFERENCES typ_designatable_item (designatable_item_id);

/* ------------------------------------------------------------------- */
/* add columns and foreign keys to cls_data_element because            */
/* data elements can be identified, classified and designated          */
/* ------------------------------------------------------------------- */

ALTER TABLE cls_data_element
  ADD COLUMN
      linked_identified_item_id          INTEGER                 UNIQUE
    ;

ALTER TABLE cls_data_element
  ADD COLUMN
      linked_classifiable_item_id        INTEGER                 UNIQUE
    ;

ALTER TABLE cls_data_element
  ADD COLUMN
      linked_designatable_item_id        INTEGER                 UNIQUE
    ;

ALTER TABLE cls_data_element
  ADD CONSTRAINT each_data_element_may_be_identified
    FOREIGN KEY (linked_identified_item_id)
      REFERENCES typ_identified_item (identified_item_id);

ALTER TABLE cls_data_element
  ADD CONSTRAINT each_data_element_may_be_classified
    FOREIGN KEY (linked_classifiable_item_id)
      REFERENCES typ_classifiable_item (classifiable_item_id);

ALTER TABLE cls_data_element
  ADD CONSTRAINT each_data_element_may_be_designated
    FOREIGN KEY (linked_designatable_item_id)
      REFERENCES typ_designatable_item (designatable_item_id);

/* ------------------------------------------------------------------- */
/* add columns and foreign keys to cls_data_element_concept because    */
/* data element concepts can be identified, classified and designated  */
/* ------------------------------------------------------------------- */

ALTER TABLE cls_data_element_concept
  ADD COLUMN
      linked_identified_item_id          INTEGER                 UNIQUE


ALTER TABLE cls_data_element_concept
  ADD COLUMN
      linked_classifiable_item_id        INTEGER                 UNIQUE
    ;

ALTER TABLE cls_data_element_concept
  ADD COLUMN
      linked_designatable_item_id        INTEGER                 UNIQUE
    ;

ALTER TABLE cls_data_element_concept
  ADD CONSTRAINT each_data_element_concept_may_be_identified
    FOREIGN KEY (linked_identified_item_id)
      REFERENCES typ_identified_item (identified_item_id);
```

```
ALTER TABLE cls_data_element_concept
  ADD CONSTRAINT each_data_element_concept_may_be_classified
    FOREIGN KEY (linked_classifiable_item_id)
      REFERENCES typ_classifiable_item (classifiable_item_id);

ALTER TABLE cls_data_element_concept
  ADD CONSTRAINT each_data_element_concept_may_be_designated
    FOREIGN KEY (linked_designatable_item_id)
      REFERENCES typ_designatable_item (designatable_item_id);

/* ------------------------------------------------------------------- */
/* add columns and foreign keys to cls_data_element_derivation because */
/* data element derivations can be identified, classified and designated */
/* ------------------------------------------------------------------- */

ALTER TABLE cls_data_element_derivation
  ADD COLUMN
      linked_identified_item_id          INTEGER                 UNIQUE
    ;

ALTER TABLE cls_data_element_derivation
  ADD COLUMN
      linked_classifiable_item_id        INTEGER                 UNIQUE
    ;

ALTER TABLE cls_data_element_derivation
  ADD COLUMN
      linked_designatable_item_id        INTEGER                 UNIQUE
    ;

ALTER TABLE cls_data_element_derivation
  ADD CONSTRAINT each_data_element_derivation_may_be_identified
    FOREIGN KEY (linked_identified_item_id)
      REFERENCES typ_identified_item (identified_item_id);

ALTER TABLE cls_data_element_derivation
  ADD CONSTRAINT each_data_element_derivation_may_be_classified
    FOREIGN KEY (linked_classifiable_item_id)
      REFERENCES typ_classifiable_item (classifiable_item_id);

ALTER TABLE cls_data_element_derivation
  ADD CONSTRAINT each_data_element_derivation_may_be_designated
    FOREIGN KEY (linked_designatable_item_id)
      REFERENCES typ_designatable_item (designatable_item_id);

/* ------------------------------------------------------------------- */
/* add columns and foreign keys to cls_data_element_example because    */
/* data element examples can be identified, classified and designated  */
/* ------------------------------------------------------------------- */

ALTER TABLE cls_data_element_example
  ADD COLUMN
      linked_identified_item_id          INTEGER                 UNIQUE
    ;

ALTER TABLE cls_data_element_example
  ADD COLUMN
      linked_classifiable_item_id        INTEGER                 UNIQUE
    ;

ALTER TABLE cls_data_element_example
  ADD COLUMN
      linked_designatable_item_id        INTEGER                 UNIQUE
    ;

ALTER TABLE cls_data_element_example
  ADD CONSTRAINT each_data_element_example_may_be_identified
    FOREIGN KEY (linked_identified_item_id)
      REFERENCES typ_identified_item (identified_item_id);
```

```
ALTER TABLE cls_data_element_example
  ADD CONSTRAINT each_data_element_example_may_be_classified
    FOREIGN KEY (linked_classifiable_item_id)
      REFERENCES typ_classifiable_item (classifiable_item_id);

ALTER TABLE cls_data_element_example
  ADD CONSTRAINT each_data_element_example_may_be_designated
    FOREIGN KEY (linked_designatable_item_id)
      REFERENCES typ_designatable_item (designatable_item_id);

/* ------------------------------------------------------------------- */
/* add columns and foreign keys to cls_datatype because                */
/* datatypes can be identified, classified and designated              */
/* ------------------------------------------------------------------- */

ALTER TABLE cls_datatype
  ADD COLUMN
      linked_identified_item_id          INTEGER                 UNIQUE
    ;

ALTER TABLE cls_datatype
  ADD COLUMN
      linked_classifiable_item_id        INTEGER                 UNIQUE
    ;

ALTER TABLE cls_datatype
  ADD COLUMN
      linked_designatable_item_id        INTEGER                 UNIQUE
    ;

ALTER TABLE cls_datatype
  ADD CONSTRAINT each_datatype_may_be_identified
    FOREIGN KEY (linked_identified_item_id)
      REFERENCES typ_identified_item (identified_item_id);

ALTER TABLE cls_datatype
  ADD CONSTRAINT each_datatype_may_be_classified
    FOREIGN KEY (linked_classifiable_item_id)
      REFERENCES typ_classifiable_item (classifiable_item_id);

ALTER TABLE cls_datatype
  ADD CONSTRAINT each_datatype_may_be_designated
    FOREIGN KEY (linked_designatable_item_id)
      REFERENCES typ_designatable_item (designatable_item_id);

/* ------------------------------------------------------------------- */
/* add columns and foreign keys to cls_derivation_rule because         */
/* derivation rules can be identified, classified and designated       */
/* ------------------------------------------------------------------- */

ALTER TABLE cls_derivation_rule
  ADD COLUMN
      linked_identified_item_id          INTEGER                 UNIQUE
    ;

ALTER TABLE cls_derivation_rule
  ADD COLUMN
      linked_classifiable_item_id        INTEGER                 UNIQUE
    ;

ALTER TABLE cls_derivation_rule
  ADD COLUMN
      linked_designatable_item_id        INTEGER                 UNIQUE
    ;

ALTER TABLE cls_derivation_rule
  ADD CONSTRAINT each_derivation_rule_may_be_identified
    FOREIGN KEY (linked_identified_item_id)
      REFERENCES typ_identified_item (identified_item_id);
```

```
ALTER TABLE cls_derivation_rule
  ADD CONSTRAINT each_derivation_rule_may_be_classified
    FOREIGN KEY (linked_classifiable_item_id)
      REFERENCES typ_classifiable_item (classifiable_item_id);

ALTER TABLE cls_derivation_rule
  ADD CONSTRAINT each_derivation_rule_may_be_designated
    FOREIGN KEY (linked_designatable_item_id)
      REFERENCES typ_designatable_item (designatable_item_id);

/* ------------------------------------------------------------------- */
/* add columns and foreign keys to cls_dimensionality because          */
/* dimensionalities can be identified, classified and designated       */
/* ------------------------------------------------------------------- */

ALTER TABLE cls_dimensionality
  ADD COLUMN
      linked_identified_item_id          INTEGER                  UNIQUE
    ;

ALTER TABLE cls_dimensionality
  ADD COLUMN
      linked_classifiable_item_id        INTEGER                  UNIQUE
    ;

ALTER TABLE cls_dimensionality
  ADD COLUMN
      linked_designatable_item_id        INTEGER                  UNIQUE
    ;

ALTER TABLE cls_dimensionality
  ADD CONSTRAINT each_dimensionality_may_be_identified
    FOREIGN KEY (linked_identified_item_id)
      REFERENCES typ_identified_item (identified_item_id);

ALTER TABLE cls_dimensionality
  ADD CONSTRAINT each_dimensionality_may_be_classified
    FOREIGN KEY (linked_classifiable_item_id)
      REFERENCES typ_classifiable_item (classifiable_item_id);

ALTER TABLE cls_dimensionality
  ADD CONSTRAINT each_dimensionality_may_be_designated
    FOREIGN KEY (linked_designatable_item_id)
      REFERENCES typ_designatable_item (designatable_item_id);

/* ------------------------------------------------------------------- */
/* add columns and foreign keys to cls_link_end because                */
/* link ends can be identified, classified and designated              */
/* ------------------------------------------------------------------- */

ALTER TABLE cls_link_end
  ADD COLUMN
      linked_identified_item_id          INTEGER                  UNIQUE
    ;

ALTER TABLE cls_link_end
  ADD COLUMN
      linked_classifiable_item_id        INTEGER                  UNIQUE
    ;

ALTER TABLE cls_link_end
  ADD COLUMN
      linked_designatable_item_id        INTEGER                  UNIQUE
    ;

ALTER TABLE cls_link_end
  ADD CONSTRAINT each_link_end_may_be_identified
    FOREIGN KEY (linked_identified_item_id)
      REFERENCES typ_identified_item (identified_item_id);
```

```
ALTER TABLE cls_link_end
  ADD CONSTRAINT each_link_end_may_be_classified
    FOREIGN KEY (linked_classifiable_item_id)
      REFERENCES typ_classifiable_item (classifiable_item_id);

ALTER TABLE cls_link_end
  ADD CONSTRAINT each_link_end_may_be_designated
    FOREIGN KEY (linked_designatable_item_id)
      REFERENCES typ_designatable_item (designatable_item_id);

/* ------------------------------------------------------------------- */
/* add columns and foreign keys to cls_namespace because               */
/* namespaces can be identified, classified and designated             */
/* ------------------------------------------------------------------- */

ALTER TABLE cls_namespace
  ADD COLUMN
      linked_identified_item_id          INTEGER                  UNIQUE
    ;

ALTER TABLE cls_namespace
  ADD COLUMN
      linked_classifiable_item_id        INTEGER                  UNIQUE
    ;

ALTER TABLE cls_namespace
  ADD COLUMN
      linked_designatable_item_id        INTEGER                  UNIQUE
    ;

ALTER TABLE cls_namespace
  ADD CONSTRAINT each_namespace_may_be_identified
    FOREIGN KEY (linked_identified_item_id)
      REFERENCES typ_identified_item (identified_item_id);

ALTER TABLE cls_namespace
  ADD CONSTRAINT each_namespace_may_be_classified
    FOREIGN KEY (linked_classifiable_item_id)
      REFERENCES typ_classifiable_item (classifiable_item_id);

ALTER TABLE cls_namespace
  ADD CONSTRAINT each_namespace_may_be_designated
    FOREIGN KEY (linked_designatable_item_id)
      REFERENCES typ_designatable_item (designatable_item_id);

/* ------------------------------------------------------------------- */
/* add columns and foreign keys to cls_naming_convention because       */
/* naming conventions can be identified, classified and designated     */
/* ------------------------------------------------------------------- */

ALTER TABLE cls_naming_convention
  ADD COLUMN
      linked_identified_item_id          INTEGER                  UNIQUE
    ;

ALTER TABLE cls_naming_convention
  ADD COLUMN
      linked_classifiable_item_id        INTEGER                  UNIQUE
    ;

ALTER TABLE cls_naming_convention
  ADD COLUMN
      linked_designatable_item_id        INTEGER                  UNIQUE
    ;

ALTER TABLE cls_naming_convention
  ADD CONSTRAINT each_naming_convention_may_be_identified
    FOREIGN KEY (linked_identified_item_id)
      REFERENCES typ_identified_item (identified_item_id);
```

```
ALTER TABLE cls_naming_convention
  ADD CONSTRAINT each_naming_convention_may_be_classified
    FOREIGN KEY (linked_classifiable_item_id)
      REFERENCES typ_classifiable_item (classifiable_item_id);

ALTER TABLE cls_naming_convention
  ADD CONSTRAINT each_naming_convention_may_be_designated
    FOREIGN KEY (linked_designatable_item_id)
      REFERENCES typ_designatable_item (designatable_item_id);

/* ------------------------------------------------------------------- */
/* add columns and foreign keys to cls_permissible_value because       */
/* permissible values can be identified, classified and designated     */
/* ------------------------------------------------------------------- */

ALTER TABLE cls_permissible_value
  ADD COLUMN
      linked_identified_item_id          INTEGER                 UNIQUE
    ;

ALTER TABLE cls_permissible_value
  ADD COLUMN
      linked_classifiable_item_id        INTEGER                 UNIQUE
    ;

ALTER TABLE cls_permissible_value
  ADD COLUMN
      linked_designatable_item_id        INTEGER                 UNIQUE
    ;

ALTER TABLE cls_permissible_value
  ADD CONSTRAINT each_permissible_value_may_be_identified
    FOREIGN KEY (linked_identified_item_id)
      REFERENCES typ_identified_item (identified_item_id);

ALTER TABLE cls_permissible_value
  ADD CONSTRAINT each_permissible_value_may_be_classified
    FOREIGN KEY (linked_classifiable_item_id)
      REFERENCES typ_classifiable_item (classifiable_item_id);

ALTER TABLE cls_permissible_value
  ADD CONSTRAINT each_permissible_value_may_be_designated
    FOREIGN KEY (linked_designatable_item_id)
      REFERENCES typ_designatable_item (designatable_item_id);

/* ------------------------------------------------------------------- */
/* add columns and foreign keys to cls_unit_of_measure because         */
/* units of measure can be identified, classified and designated       */
/* ------------------------------------------------------------------- */

ALTER TABLE cls_unit_of_measure
  ADD COLUMN
      linked_identified_item_id          INTEGER                 UNIQUE
    ;

ALTER TABLE cls_unit_of_measure
  ADD COLUMN
      linked_classifiable_item_id        INTEGER                 UNIQUE
    ;

ALTER TABLE cls_unit_of_measure
  ADD COLUMN
      linked_designatable_item_id        INTEGER                 UNIQUE
    ;

ALTER TABLE cls_unit_of_measure
  ADD CONSTRAINT each_unit_of_measure_may_be_identified
    FOREIGN KEY (linked_identified_item_id)
      REFERENCES typ_identified_item (identified_item_id);
```

```
ALTER TABLE cls_unit_of_measure
  ADD CONSTRAINT each_unit_of_measure_may_be_classified
    FOREIGN KEY (linked_classifiable_item_id)
      REFERENCES typ_classifiable_item (classifiable_item_id);

ALTER TABLE cls_unit_of_measure
  ADD CONSTRAINT each_unit_of_measure_may_be_designated
    FOREIGN KEY (linked_designatable_item_id)
      REFERENCES typ_designatable_item (designatable_item_id);

/* ------------------------------------------------------------------ */
/* add columns and foreign keys to cls_value_domain because           */
/* value domains can be identified, classified and designated         */
/* ------------------------------------------------------------------ */

ALTER TABLE cls_value_domain
  ADD COLUMN
      linked_identified_item_id          INTEGER                  UNIQUE
    ;

ALTER TABLE cls_value_domain
  ADD COLUMN
      linked_classifiable_item_id        INTEGER                  UNIQUE
    ;

ALTER TABLE cls_value_domain
  ADD COLUMN
      linked_designatable_item_id        INTEGER                  UNIQUE
    ;

ALTER TABLE cls_value_domain
  ADD CONSTRAINT each_value_domain_may_be_identified
    FOREIGN KEY (linked_identified_item_id)
      REFERENCES typ_identified_item (identified_item_id);

ALTER TABLE cls_value_domain
  ADD CONSTRAINT each_value_domain_may_be_classified
    FOREIGN KEY (linked_classifiable_item_id)
      REFERENCES typ_classifiable_item (classifiable_item_id);

ALTER TABLE cls_value_domain
  ADD CONSTRAINT each_value_domain_may_be_designated
    FOREIGN KEY (linked_designatable_item_id)
      REFERENCES typ_designatable_item (designatable_item_id);

/* ------------------------------------------------------------------ */
/* add columns and foreign keys to cls_value_meaning because          */
/* value meanings can be identified, classified and designated        */
/* ------------------------------------------------------------------ */

ALTER TABLE cls_value_meaning
  ADD COLUMN
      linked_identified_item_id          INTEGER                  UNIQUE


ALTER TABLE cls_value_meaning
  ADD COLUMN
      linked_classifiable_item_id        INTEGER                  UNIQUE
    ;

ALTER TABLE cls_value_meaning
  ADD COLUMN
       linked_designatable_item_id        INTEGER                  UNIQUE
    ;

ALTER TABLE cls_value_meaning
  ADD CONSTRAINT each_value_meaning_may_be_identified
    FOREIGN KEY (linked_identified_item_id)
      REFERENCES typ_identified_item (identified_item_id);
```

```
ALTER TABLE cls_value_meaning
  ADD CONSTRAINT each_value_meaning_may_be_classified
    FOREIGN KEY (linked_classifiable_item_id)
      REFERENCES typ_classifiable_item (classifiable_item_id);

ALTER TABLE cls_value_meaning
  ADD CONSTRAINT each_value_meaning_may_be_designated
    FOREIGN KEY (linked_designatable_item_id)
      REFERENCES typ_designatable_item (designatable_item_id);


/* ------------------------------------------------------------------------ */
/* create tables for the 11179-3 associations                               */
/* ------------------------------------------------------------------------ */

CREATE TABLE ass_assertion_inclusion
/* a many-to-many association */
  (
    /* columns and foreign keys for roles of the association */
    assertor                                    INTEGER
                                  REFERENCES cls_concept_system (concept_system_id)  ,
    included_assertion                          INTEGER
                                REFERENCES cls_assertion (assertion_id)  ,
    PRIMARY KEY (assertor, included_assertion)
  );

CREATE TABLE ass_assertion_term
/* a one-to-many association */
  (
    /* columns and foreign keys for roles of the association */
    term                                        INTEGER            PRIMARY KEY
                                          REFERENCES cls_concept (concept_id)  ,
    assertion                                   INTEGER
                                        REFERENCES cls_assertion (assertion_id)
  );

CREATE TABLE ass_attachment
/* a one-to-many association */
  (
    /* columns and foreign keys for roles of the association */
    attached_item                     INTEGER            PRIMARY KEY
                                REFERENCES typ_attached_item (attached_item_id)  ,
    owner                             INTEGER
                REFERENCES typ_administered_item (administered_item_id)
  );

CREATE TABLE ass_classification_scheme
/* a many-to-many association */
  (
    /* columns and foreign keys for roles of the association */
    classification                              INTEGER
                            REFERENCES asscls_classification (classification_id)  ,
    scheme                                      INTEGER
                              REFERENCES cls_concept_system (concept_system_id)  ,
    PRIMARY KEY (classification, scheme)
  );

CREATE TABLE ass_concept_source
/* a one-to-many association */
  (
    /* columns and foreign keys for roles of the association */
    sourced_concept                             INTEGER            PRIMARY KEY
                                          REFERENCES cls_concept (concept_id)  ,
    source                                      INTEGER
                                  REFERENCES cls_concept_system (concept_system_id)
  );
```

```
CREATE TABLE ass_concept_system_importation
/* a many-to-many reflexive association */
  (
    /* columns and foreign keys for roles of the association */
    imported_concept_system                          INTEGER
                                REFERENCES cls_concept_system (concept_system_id)  ,
    importing_concept_system                         INTEGER
                                REFERENCES cls_concept_system (concept_system_id)  ,
    PRIMARY KEY (imported_concept_system, importing_concept_system)
  );

CREATE TABLE ass_concept_system_membership
/* a many-to-many association */
  (
    /* columns and foreign keys for roles of the association */
    including_concept_system                         INTEGER
                                REFERENCES cls_concept_system (concept_system_id)  ,
    member_concept                                   INTEGER
                                              REFERENCES cls_concept (concept_id)  ,
    PRIMARY KEY (including_concept_system, member_concept)
  );

CREATE TABLE ass_concept_system_reference
/* a many-to-many reflexive association */
  (
    /* columns and foreign keys for roles of the association */
    referenced_concept_system                        INTEGER
                                REFERENCES cls_concept_system (concept_system_id)  ,
    referencing_concept_system                       INTEGER
                                REFERENCES cls_concept_system (concept_system_id)  ,
    PRIMARY KEY (referenced_concept_system, referencing_concept_system)
  );

CREATE TABLE ass_data_element_concept_domain
/* a many-to-many association */
  (
    /* columns and foreign keys for roles of the association */
    usage                                            INTEGER
                    REFERENCES cls_data_element_concept (data_element_concept_id)  ,
   dommain                                           INTEGER
                    REFERENCES cls_conceptual_domain (conceptual_domain_id)  ,
     PRIMARY KEY (usage, domain)
  );

CREATE TABLE ass_data_element_concept_object_class
/* a one-to-many association */
  (
    /* columns and foreign keys for roles of the association */
    object_class                                     INTEGER           PRIMARY KEY
                                  REFERENCES cls_object_class (object_class_id)  ,
    data_element_concept                             INTEGER
                    REFERENCES cls_data_element_concept (data_element_concept_id)
  );

CREATE TABLE ass_data_element_concept_property
/* a one-to-many association */
  (
    /* columns and foreign keys for roles of the association */
    property                                         INTEGER           PRIMARY KEY
                                REFERENCES cls_property (property_id)  ,
    data_element_concept                             INTEGER
                    REFERENCES cls_data_element_concept (data_element_concept_id)
  );
```

```
CREATE TABLE ass_data_element_meaning
/* a one-to-many association */
  (
    /* columns and foreign keys for roles of the association */
    representation                                  INTEGER           PRIMARY KEY
                                  REFERENCES cls_data_element (data_element_id)  ,
    meaning                                         INTEGER
                      REFERENCES cls_data_element_concept (data_element_concept_id)
  );

CREATE TABLE ass_derivation_input
/* a many-to-many association */
  (
    /* columns and foreign keys for roles of the association */
    input                                           INTEGER
                                  REFERENCES cls_data_element (data_element_id)  ,
    inputter                                        INTEGER

                REFERENCES cls_data_element_derivation (data_element_derivation_id)  ,
    PRIMARY KEY (input, inputter)
  );

CREATE TABLE ass_derivation_output
/* a many-to-many association */
  (
    /* columns and foreign keys for roles of the association */
    output                                          INTEGER
                                  REFERENCES cls_data_element (data_element_id)  ,
    derivation                                      INTEGER

                REFERENCES cls_data_element_derivation (data_element_derivation_id)  ,
    PRIMARY KEY (output, derivation)
  );

CREATE TABLE ass_derivation_rule_application
/* a one-to-many association */
  (
    /* columns and foreign keys for roles of the association */
    application                                     INTEGER           PRIMARY KEY
                                  REFERENCES cls_data_element (data_element_id)  ,
    rule                                            INTEGER
                                REFERENCES cls_derivation_rule (derivation_rule_id)

  );

CREATE TABLE ass_designation_definition_pairing
/* a one-to-one association */
  (
    /* columns and foreign keys for roles of the association */
    definition_heading                  INTEGER             UNIQUE
                           REFERENCES cls_designation (designation_id)  ,
    specific_definition                 INTEGER             UNIQUE
                             REFERENCES cls_definition (definition_id)  ,
    PRIMARY KEY (definition_heading, specific_definition)
  );

CREATE TABLE ass_designation_namespace
/* a many-to-many association */
  (
    /* columns and foreign keys for roles of the association */
    included_designation                INTEGER
                           REFERENCES cls_designation (designation_id)  ,
    namespace                           INTEGER
                               REFERENCES cls_namespace (namespace_id)  ,
    PRIMARY KEY (included_designation, namespace)
  );
```

```
CREATE TABLE ass_dimensionality_measure_class
/* a many-to-many association */
  (
    /* columns and foreign keys for roles of the association */
    applicable_unit                                INTEGER
                                 REFERENCES cls_measure_class (measure_class_id)  ,
    dimensionality                                 INTEGER
                               REFERENCES cls_dimensionality (dimensionality_id)  ,
    PRIMARY KEY (applicable_unit, dimensionality)
  );

CREATE TABLE ass_exemplification
/* a many-to-many association */
  (
    /* columns and foreign keys for roles of the association */
    exhibitor                                      INTEGER
                                  REFERENCES cls_data_element (data_element_id)  ,
    example                                        INTEGER
                       REFERENCES cls_data_element_example (data_element_example_id)  ,
    PRIMARY KEY (exhibitor, example)
  );

CREATE TABLE ass_identification
/* a one-to-many association */
  (
    /* columns and foreign keys for roles of the association */
    identified_item                  INTEGER          PRIMARY KEY
                     REFERENCES typ_identified_item (identified_item_id)  ,
    identifier                       INTEGER
                     REFERENCES cls_scoped_identifier
                                                (scoped_identifier_id)
  );

CREATE TABLE ass_identifier_scope
/* a one-to-many association */
  (
    /* columns and foreign keys for roles of the association */
    contained_identifier             INTEGER            PRIMARY KEY
                                  REFERENCES cls_scoped_identifier
                                                (scoped_identifier_id)  ,
    scope                            INTEGER
                                REFERENCES cls_namespace (namespece_id)
  );

CREATE TABLE ass_item_definition
/* a one-to-many association */
  (
    /* columns and foreign keys for roles of the association */
    definition                       INTEGER            PRIMARY KEY
                             REFERENCES cls_definition (definition_id)  ,
    item                             INTEGER
                     REFERENCES typ_designatable_item
                                                (designatable_item_id)
  );

CREATE TABLE ass_item_designation
/* a one-to-many association */
  (
    /* columns and foreign keys for roles of the association */
    designation                      INTEGER            PRIMARY KEY
                           REFERENCES cls_designation (designation_id)  ,
    item                             INTEGER
                              REFERENCES typ_designatable_item
                                                (designatable_item_id)
  );
```

```
CREATE TABLE ass_item_slot
/* a one-to-many association */
  (
    /* columns and foreign keys for roles of the association */
    slot                                INTEGER            PRIMARY KEY            ,
                                         REFERENCES cls_slot (slot_id)  ,
    item                                INTEGER
                    REFERENCES typ_identified_item (identified_item_id)
  );

CREATE TABLE ass_link_end_concept
/* a one-to-many association */
  (
    /* columns and foreign keys for roles of the association */
    link_end                                          INTEGER            PRIMARY KEY
                                                REFERENCES cls_link_end (link_end_id)  ,
    concept                                           INTEGER
                                                  REFERENCES cls_concept (concept_id)
  );

CREATE TABLE ass_link_end_role
/* a one-to-many association */
  (
    /* columns and foreign keys for roles of the association */
    link_end                                          INTEGER            PRIMARY KEY
                                                REFERENCES cls_link_end (link_end_id)  ,
    role                                              INTEGER
                                      REFERENCES cls_relation_role (relation_role_id)
  );

CREATE TABLE ass_link_has_link_end
/* a one-to-many association */
  (
    /* columns and foreign keys for roles of the association */
    link_end                                          INTEGER            PRIMARY KEY
                                                REFERENCES cls_link_end (link_end_id)  ,
    link                                              INTEGER
                                                   REFERENCES cls_link (link_id)
  );

CREATE TABLE ass_naming_convention_conformance
/* a many-to-many association */
  (
    /* columns and foreign keys for roles of the association */
    convention                          INTEGER
                  REFERENCES cls_naming_convention (naming_convention_id)  ,
    conformant_designation              INTEGER
                               REFERENCES cls_designation (designation_id)  ,
    PRIMARY KEY (convention, conformant_designation)
  );

CREATE TABLE ass_naming_convention_utilization
/* a many-to-many association */
  (
    /* columns and foreign keys for roles of the association */
    acceptable_convention               INTEGER

                 REFERENCES cls_naming_convention (naming_concention_id)  ,
    utilization                         INTEGER
                                 REFERENCES cls_namespace (namespace_id)  ,
    PRIMARY KEY (acceptable_convention, utilization)
  );
```

```
CREATE TABLE ass_permissible_value_meaning
/* a one-to-many association */
  (
    /* columns and foreign keys for roles of the association */
    representation                                    INTEGER            PRIMARY KEY
                            REFERENCES cls_permissible_value (permissible_value_id)  ,
    meaning                                           INTEGER
                                    REFERENCES cls_value_meaning (value_meaning_id)
  );

CREATE TABLE ass_permissible_value_set
/* a many-to-many association */
  (
    /* columns and foreign keys for roles of the association */
    permissible_value                                 INTEGER
                            REFERENCES cls_permissible_value (permissible_value_id)  ,
    containing_domain                                 INTEGER

                REFERENCES cls_enumerated_value_domain (value_domain_id)             ,
    PRIMARY KEY (permissible_value, containing_domain)
  );

CREATE TABLE ass_registration_authority_namespace
/* a one-to-one association */
  (
    /* columns and foreign keys for roles of the association */
    registration_namespace                            INTEGER                 UNIQUE
                                          REFERENCES cls_namespace (namespace_id)  ,
    maintainer                                        INTEGER                 UNIQUE
                  REFERENCES cls_registration_authority (registration_authority_id)  ,
    PRIMARY KEY (registration_namespace, maintainer)
  );

CREATE TABLE ass_registration_authority_namespace
/* a one-to-many association */
  (
    /* columns and foreign keys for roles of the association */
    usage                                               INTEGER           PRIMARY KEY
                                      REFERENCES cls_data_element (data_element_id)  ,
    domain                                            INTEGER
                                      REFERENCES cls_value_domain (value_domain_id)
  );

CREATE TABLE ass_registration_authority_namespace
/* a one-to-many association */
  (
    /* columns and foreign keys for roles of the association */
    representation                                    INTEGER             PRIMARY KEY
                             REFERENCES cls_described_value_domain (value_domain_id)  ,
    meaning                                           INTEGER
                   REFERENCES cls_described_conceptual_domain (conceptual_domain_id)
  );

CREATE TABLE ass_registration_authority_namespace
/* a many-to-many reflexive association */
  (
    /* columns and foreign keys for roles of the association */
    superdomain                                       INTEGER
                                      REFERENCES cls_value_domain (value_domain_id)  ,
    subdomain                                         INTEGER
                                      REFERENCES cls_value_domain (value_domain_id)  ,
    PRIMARY KEY (superdomain, subdomain)
  );
```

```
CREATE TABLE ass_registration_authority_registrar
/* a one-to-many association */
  (
    /* columns and foreign keys for roles of the association */
    registrar                                       INTEGER            PRIMARY KEY
                                                REFERENCES cls_registrar (registrar_id)  ,
    authority                                       INTEGER
                  REFERENCES cls_registration_authority (registration_authority_id)
  );

CREATE TABLE ass_relation_link
/* a one-to-many association */
  (
    /* columns and foreign keys for roles of the association */
    link                                            INTEGER            PRIMARY KEY
                                                     REFERENCES cls_link (link_id)  ,
    relation                                        INTEGER
                                               REFERENCES cls_relation (relation_id)
  );

CREATE TABLE ass_relation_role_set
/* a one-to-many association */
  (
    /* columns and foreign keys for roles of the association */
    role                                            INTEGER            PRIMARY KEY
                                    REFERENCES cls_relation_role (relation_role_id)  ,
    source                                          INTEGER
                                               REFERENCES cls_relation (relation_id)
  );

CREATE TABLE ass_stewardship
/* a one-to-many association */
  (
    /* columns and foreign keys for roles of the association */
    stewarded_item                                  INTEGER            PRIMARY KEY
                            REFERENCES typ_administered_item (administered_item_id)  ,
    stewardship_record                              INTEGER
                          REFERENCES cls_stewardship_record (stewardship_record_id)
  );

CREATE TABLE ass_submission
/* a many-to-many association */
  (
    /* columns and foreign keys for roles of the association */
    submission_record                               INTEGER
                                  REFERENCES cls_submitted_record (submitted_record_id)  ,
    submitted_item                                  INTEGER
                                REFERENCES typ_registered_item (registered_item_id)  ,
    PRIMARY KEY (submission_record, submitted_item)
  );

CREATE TABLE ass_unit_of_measure_class
/* a many-to-many association */
  (
    /* columns and foreign keys for roles of the association */
    member_unit                                     INTEGER
                                REFERENCES cls_unit_of_measure (unit_of_measure_id)  ,
    measure_class                                   INTEGER
                                    REFERENCES cls_measure_class (measure_class_id)  ,
    PRIMARY KEY (member_unit, measure_class)
  );
```