# Container-Template Subject Structure

## Container/Template Subject Structure (CTSS)

***Please note that this page and related pages are being developed as part of the CIM Courses Project and are subject to change.***

The Container/Template Subject Structure was built as part of the CIM Courses Project. This article describes the structure and functionality of the Container/Template Structure.

**Table of Contents:**


- Container/Template Subject Structure (CTSS)
  - Purpose of the Container/Template Subject Structure
  - Content of the Container/Template Subject Structure
  - Foundational Tables
    - Subject Container ( `SUBJECT_CONTAINER`)
      - Status and Last Active Term
    - Subject Template ( `SUBJECT_TEMPLATE`)
    - Subject Registerable Instance ( `SUBJECT_TMPL_REG_INSTANCE`)
  - Foundational Concepts
    - Foundational Table Relationships
    - Cross-Listed Subject Numbers
    - Prerequisites and Corequisites, aka Requisites ( `SUBJECT_TMPL_REQUISITE`)
      - Example requisite trees
    - Enrollment Limitations ( `SUBJECT_TMPL_ENROLL_LIMIT`)
    - Other Subject Template Tables
    - Equivalencies (EQs) and Scheduling Relationships (SRs)
  - Basic Functionality Flow
      - Subject Creation
      - Subject Editing
      - Subject Deactivation
      - Subject Reactivation
      - Complex Use Cases/Test Cases
  - Data Models
    - Subject Container Data Model
    - Subject Template Data Model
  - System Rules
  - Other Documentation
    - Expected Behavior When A New Subject is Approved and Deactivated in the Same Proposal Term
  - Subject Management Documentation Index


## Purpose of the Container/Template Subject Structure

The Container/Template Subject Structure was developed to be the new system of record for subjects at MIT. It was designed to make several improvements over legacy systems which are further discussed here.

## Content of the Container/Template Subject Structure

The CTSS contains all subject data from 2013FA onward. It also contains the most recent data for some subjects that were deactivated between 2008FA and 2012SU if a subject that was active in 2013FA had an equivalency with that subject.

## Foundational Tables

The Container/Template Subject Structure is built upon three foundational tables:

- Subject Container (`SUBJECT_CONTAINER`) - the subject over time
- Subject Template (`SUBJECT_TEMPLATE`)- details about the subject during a particular period of time
- Subject Registerable Instance (`SUBJECT_TMPL_REG_INSTANCE`)- the subject numbers under which the subject is offered.

### Subject Container (`SUBJECT_CONTAINER`)

The Subject Container can be thought of as "the subject over time", "the conceptual subject", or more simply, just "the subject".

| | COLUMN_NAME | DATA_TYPE | NULLABLE | DATA_DEFAULT | COLUMN_ID | COMMENTS |
|---|---|---|---|---|---|---|
| 1 | SUBJECT_CONTAINER_ID | NUMBER(10,0) | No | (null) | 1 | Unique identifier number of a subject container |
| 2 | SUBJECT_TYPE | VARCHAR2(20 CHAR) | No | (null) | 2 | Type of subject - Administrative, Crossreg, Standard |
| 3 | LAST_ACTIVE_TERM | VARCHAR2(6 CHAR) | Yes | (null) | 3 | Term Code (eg 2015FA) in which an inactive subject was last active. Null for active subjects |
| 4 | STATUS | VARCHAR2(20 BYTE) | No | (null) | 4 | Status of subject container - Active, Inactive |
| 5 | CREATE_BY | VARCHAR2(20 CHAR) | No | (null) | 5 | Identifier of individual or system that created record. Individuals are identified by kerb id. |
| 6 | CREATE_DATE | TIMESTAMP(6) | No | (null) | 6 | Date and time when record was created |
| 7 | MODIFY_BY | VARCHAR2(20 CHAR) | Yes | (null) | 7 | Identifier of individual or system that last modified record. Individuals are identified by kerb id. |
| 8 | MODIFY_DATE | TIMESTAMP(6) | Yes | (null) | 8 | Date and time when record was last modified |

(Screenshot up-to-date as of 2/21/2019)

When a subject is created, a Subject Container record is inserted into the Subject Container table. Since most data about a subject is stored in the Subject Template table, there are only a few fields that pertain to the Subject Container.

### Status and Last Active Term

`SUBJECT_CONTAINER` contains a `STATUS` column. This is the status (Active/Inactive) of the subject as of the current academic term (SBGPG_TERM where application_code = 'AC' and curr_fut_ind = 'C'). Deactivations and reactivations are usually recorded in advance. The value in the `STATUS` column does not change when the action is recorded. It only changes when the action takes effect. `SUBJECT_CONTAINER` also contains a `LAST_ACTIVE_TERM` column which is populated for deactivated containers, unless the container has been reactivated. The `LAST_ACTIVE_TERM` column is cleared when a subject is reactivated. The `LAST_ACTIVE_TERM` is updated when a deactivation or reactivation is recorded.

The data in the `STATUS` and `LAST_ACTIVE_TERM` columns can be derived from the `EFFECTIVE_FROM_TERM` and `EFFECTIVE_THRU_TERM` of a container's templates (See next section for more details). The `SUBJECT_CONTAINER` columns are intended to facilitate certain queries, but are not suited for all use cases. For example, changes to subjects are usually proposed an academic year ahead of the actual year, so the subject proposal use cases typically need the status as of a future term, not the current academic term. A container is active in a given term if the term falls between the `EFFECTIVE_FROM_TERM` and `EFFECTIVE_THRU_TERM` of any of its templates.

## Subject Template (`SUBJECT_TEMPLATE`)

The Subject Template contains the details of a subject during a given period of time. The majority of fields that describe a subject are stored in this table, e.g. Main Subject Number, Title, Description, etc.

Each Subject Template maps back to one and only one Subject Container. As such, a Subject Container can have multiple Subject Templates associated with it.

| | COLUMN_NAME | DATA_TYPE | NULLABLE | DATA_DEFAULT | COLUMN_ID | COMMENTS |
|---|---|---|---|---|---|---|
| 1 | SUBJECT_TEMPLATE_ID | VARCHAR2(32 BYTE) | No | (null) | 1 | Unique identifier GUID of a subject template |
| 2 | SUBJECT_CONTAINER_ID | NUMBER(10,0) | No | (null) | 2 | Unique id GUID of a container in subject_container table |
| 3 | EFFECTIVE_FROM_TERM | VARCHAR2(6 CHAR) | No | (null) | 3 | The term from which a subject template is valid from |
| 4 | EFFECTIVE_THRU_TERM | VARCHAR2(6 CHAR) | No | (null) | 4 | The term upto which a subject template is valid |
| 5 | MAIN_SUBJECT_CODE | VARCHAR2(4 CHAR) | No | (null) | 5 | Primary subject code |
| 6 | MAIN_SUBJECT_NUMBER | VARCHAR2(6 CHAR) | No | (null) | 6 | Primary subject number |
| 7 | DESCRIPTION | VARCHAR2(4000 CHAR) | Yes | (null) | 7 | Detailed description of a subject |
| 8 | SORT_KEY | VARCHAR2(30 CHAR) | Yes | (null) | 8 | Key field used to sort subject in a meaningful way |
| 9 | FULL_TITLE | VARCHAR2(255 CHAR) | Yes | (null) | 9 | Full title of the approved subject |
| 10 | FULL_TITLE_SEARCHABLE | VARCHAR2(255 CHAR) | Yes | (null) | 10 | Field used to sort by title |
| 11 | SHORT_TITLE | VARCHAR2(30 CHAR) | Yes | (null) | 11 | Short title of the approved subject displayed on catalog etc. |
| 12 | ALTERNATE_TITLE | VARCHAR2(255 CHAR) | Yes | (null) | 12 | Alternate title of the subject |
| 13 | ALTERNATE_DESCRIPTION | VARCHAR2(4000 CHAR) | Yes | (null) | 13 | Alternate descriotion which gives more information about the subject |
| 14 | SUBJECT_LEVEL | VARCHAR2(2 CHAR) | Yes | (null) | 14 | Subject level whether it is U - Undergraduate or G - Graduate subject |
| 15 | LECTURE_UNITS | NUMBER(4,2) | Yes | (null) | 15 | Lecture credit units assigned for the subject |
| 16 | VARIABLE_UNITS | VARCHAR2(1 CHAR) | Yes | (null) | 16 | Whether variable unit subject or not Y /N |
| 17 | LAB_UNITS | NUMBER(4,2) | Yes | (null) | 17 | Lab credit units assigned for the subject |
| 18 | PREP_UNITS | NUMBER(4,2) | Yes | (null) | 18 | Prep credit units assigned for the subject |
| 19 | BILLING_UNITS | NUMBER(4,2) | Yes | (null) | 19 | Billing units assigned for the subject |
| 20 | TOTAL_LOAD_UNITS | NUMBER(4,2) | Yes | (null) | 20 | Total load units assigned for the subject |
| 21 | TOTAL_CREDIT_LIMIT_UNITS | NUMBER(4,2) | Yes | (null) | 21 | Total credits assigned for the subject |
| 22 | CATALOG_FACULTY | VARCHAR2(300 CHAR) | Yes | (null) | 22 | Catalog faculty assigned for the subject |
| 23 | CATALOG_FACULTY_PREFACE | VARCHAR2(10 CHAR) | Yes | (null) | 23 | Catalog faculty preface |
| 24 | IS_OFFERED_FALL | VARCHAR2(1 CHAR) | Yes | (null) | 24 | Is the subject offered in fall Y/N |
| 25 | IS_OFFERED_IAP | VARCHAR2(1 CHAR) | Yes | (null) | 25 | Is the subject offered for IAP Y/N |
| 26 | IS_OFFERED_SPRING | VARCHAR2(1 CHAR) | Yes | (null) | 26 | Is the subject offered in Spring Y/N |
| 27 | IS_OFFERED_SUMMER | VARCHAR2(1 CHAR) | Yes | (null) | 27 | Is the subject offered in Summer Y/N |
| 28 | TERM_DURATION | VARCHAR2(60 CHAR) | Yes | (null) | 28 | Duration of term whether full term etc |
| 29 | FIRST_WEEK | NUMBER(2,0) | Yes | (null) | 29 | First week term plan |
| 30 | LAST_WEEK | NUMBER(2,0) | Yes | (null) | 30 | Last week term plan |
| 31 | FREQUENCY | VARCHAR2(60 CHAR) | Yes | (null) | 31 | Term plan frequency |
| 32 | NEXT_OFFERED_YEAR | NUMBER(4,0) | Yes | (null) | 32 | (null) |
| 33 | ROLL_FACULTY_IND | VARCHAR2(1 BYTE) | Yes | (null) | 33 | When generating instructor assign records for new year, should value be taken from P(revious) or L(ike) term |
| 34 | GRAD_SCHED_TYPE_CODE | VARCHAR2(1 BYTE) | Yes | (null) | 34 | Gradable Schedule Type Code |
| 35 | UG_DEGREE_AUDIT_LAB_UNITS | NUMBER(2,0) | Yes | (null) | 35 | Number of units that can be applied to fulfilling the UG Laboratory Requirement |
| 36 | ADDITIONAL_LISTING_CONTENT | VARCHAR2(500 CHAR) | Yes | (null) | 36 | Additional content that should be displayed in subject listings, but is not appropriate for description, enrollment limitation etc |
| 37 | CREATE_BY | VARCHAR2(20 CHAR) | Yes | (null) | 37 | Identifier of individual or system that created record. Individuals are identified by kerb id. |
| 38 | CREATE_DATE | TIMESTAMP(6) | Yes | (null) | 38 | Date and time when record was created |
| 39 | MODIFY_BY | VARCHAR2(20 BYTE) | Yes | (null) | 39 | Identifier of individual or system that last modified record. Individuals are identified by kerb id. |
| 40 | MODIFY_DATE | TIMESTAMP(6) | Yes | (null) | 40 | Date and time when record was last modified |
| 41 | VERSION_NUM | NUMBER(5,0) | No | (null) | 41 | version number of record for concurrency control |

(Screenshot up-to-date as of 2/21/2019)

The life of a Subject Template is determined by the `Effective_From_Term` and the `Effective_Thru_Term` fields. Templates will be contiguous (e.g. 2013FA-2015SU, 2016FA-2017SU, 2018FA-999999) unless a subject has been deactivated. A container only has templates when it is active. There are no templates covering a period of inactivity. If a subject is reactivated, there will be a gap in the templates. For example, if a subject is deactivated at the end of AY2017 then reactivated for AY2020, it will have one template with an effective thru term of 2017SU and another with an effective from term of 2020FA. There will be no template for 2018FA-2019SU.

## Subject Registerable Instance (`SUBJECT_TMPL_REG_INSTANCE`)

Each Subject Template has a Registerable Instance for each subject number under which the subject is offered.

| | COLUMN_NAME | DATA_TYPE | NULLABLE | DATA_DEFAULT | COLUMN_ID | COMMENTS |
|---|---|---|---|---|---|---|
| 1 | SUBJECT_TMPL_REG_INSTANCE_ID | VARCHAR2(32 BYTE) | No | (null) | 1 | Unique identifier GUID of a subject registerable instance |
| 2 | SUBJECT_TEMPLATE_ID | VARCHAR2(32 BYTE) | No | (null) | 2 | Unique identifier GUID of a subject template in subject_template table |
| 3 | SUBJECT_CODE | VARCHAR2(4 CHAR) | No | (null) | 3 | Subject code of the registerable instance associated with a subject |
| 4 | SUBJECT_NUMBER | VARCHAR2(6 CHAR) | No | (null) | 4 | Subject number of the registerable instance a subject |
| 5 | IS_PRIMARY | VARCHAR2(1 CHAR) | No | (null) | 5 | Whather this registerable instance is primary or not Y/N |
| 6 | DEPT_CODE | VARCHAR2(4 CHAR) | Yes | (null) | 6 | Department which owns this registerable instance of the subject |
| 7 | PERCENT_CONTRIBUTION | NUMBER(5,2) | Yes | (null) | 7 | Percentage contribution of the dept for the subject |
| 8 | SORT_KEY | VARCHAR2(30 CHAR) | Yes | (null) | 8 | Field which has the key to sort the registerable instance |
| 9 | CREATE_BY | VARCHAR2(20 CHAR) | No | (null) | 9 | Identifier of individual or system that created record. Individuals are identified by kerb id. |
| 10 | CREATE_DATE | TIMESTAMP(6) | No | (null) | 10 | Date and time when record was created |
| 11 | MODIFY_BY | VARCHAR2(20 CHAR) | Yes | (null) | 11 | Identifier of individual or system that last modified record. Individuals are identified by kerb id. |
| 12 | MODIFY_DATE | TIMESTAMP(6) | Yes | (null) | 12 | Date and time when record was last modified |

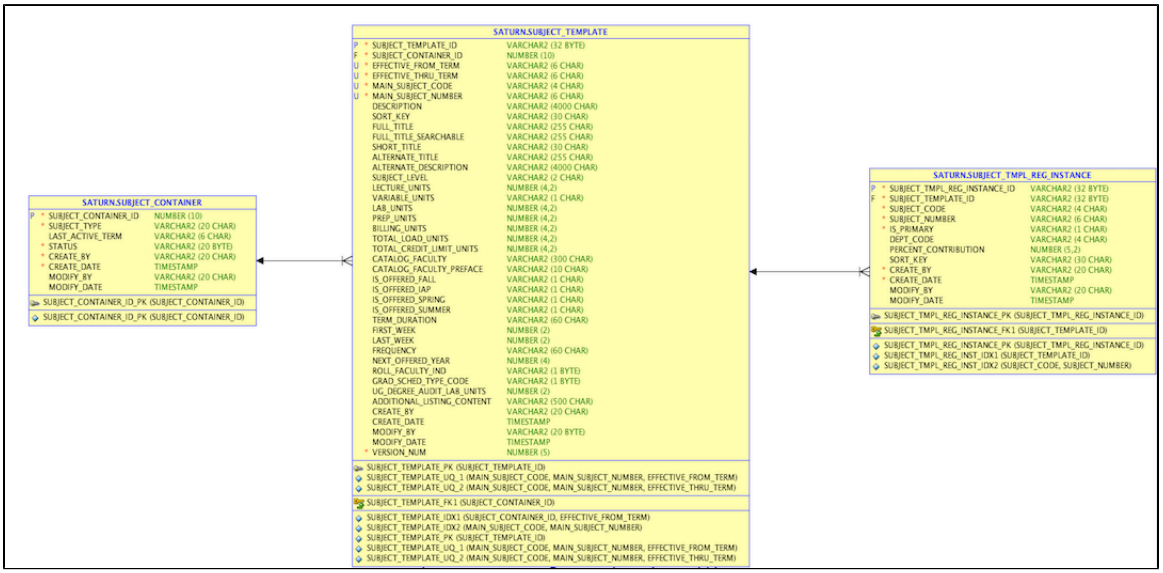
(Screenshot up-to-date as of 2/21/2019)

# Foundational Concepts

Before understanding the functionality of the Container/Template Subject Structure, it is important to understand a few concepts:

## Foundational Table Relationships

The foundational tables are related as seen here:




## Cross-Listed Subject Numbers

A Subject is considered to be cross-listed when it is offered under two or more different Subject Numbers. One of the numbers will be designated the primary or main number.

Although the subject numbers are conceptually just different labels for the same subject, the current registration model has students registering for a particular number. The numbers are therefore treated as a full-fledged domain object, the Registerable Instance. Each template will have one Registerable Instance for each number under which it is offered, including the primary number. The primary number also appears directly in the subject template as the main subject code/number. Registerable Instances are recorded in `SUBJECT_TMPL_REG_INSTANCE`.

## Prerequisites and Corequisites, aka Requisites (`SUBJECT_TMPL_REQUISITE`)

Prerequisites and corequisites, referred to collectively as requisites, are structured data. The structure was designed to improve the consistency of data entry and enable software systems to interpret and analyze requisites and students' fulfillment of them correctly

Requisites are now represented as a non-binary tree structure. (see https://en.wikipedia.org/wiki/Tree_(data_structure)). Each node in the tree is either specifies a simple requisite – a single subject, a single GIR, etc – or a compound requisite – and "AND" or "OR" relationship between two or more requisites. Requisites are typed. As of 2/2019, the simple requisites types are subject, GIR, permission and free text.

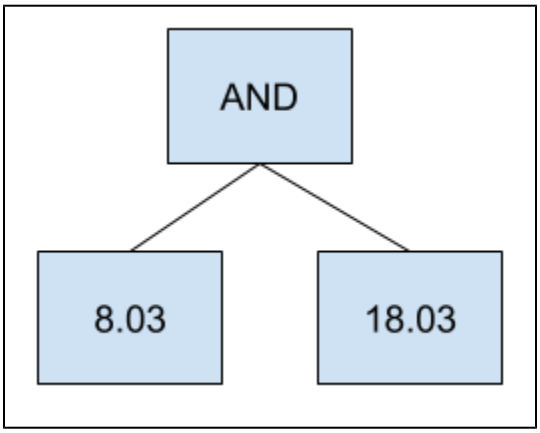
Requisite data is stored is stored in `SUBJECT_TMPL_REQUISITE`. The requisite types can be found in `SUBJECT_REQUISITE_TYPE`.

More details on requisites and how they are handled can be found here.

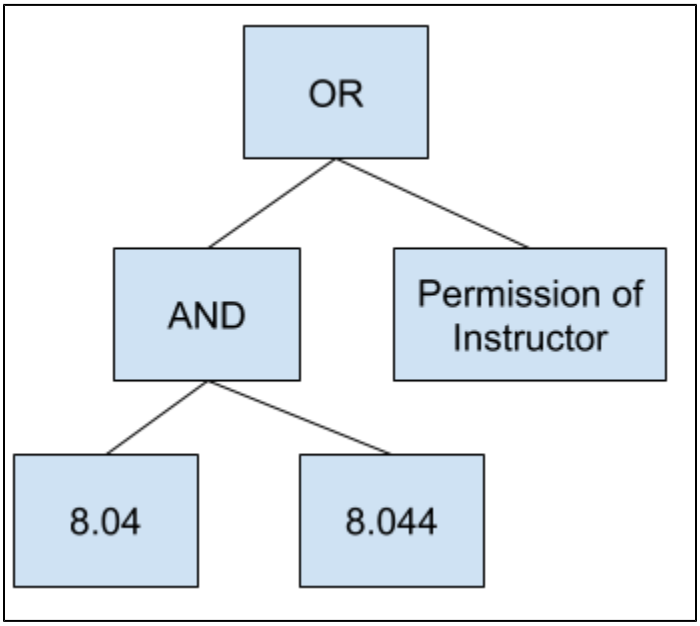
### Example requisite trees

The tree structure that we use in the code to represent the requisites stores the operators and operands in the requisites descriptor as a set of linked tree nodes. From the two examples above, the trees look like this:

8.03 and 18.03




(8.04 and 8.044) or permission of instructor




## Enrollment Limitations (`SUBJECT_TMPL_ENROLL_LIMIT`)

Enrollment limitations, restrictions on the type or number of students who can take a subject, are structured data. The structure was designed to allow automated enforcement of the restrictions.

Enrollment limitations are stored in `SUBJECT_TMPL_ENROLL_LIMIT`. There are various types of enrollment limitations, such as max enrollees or no listeners. Multiple limitations can be combined on a single template. The enrollment limit types can be found in `SUBJECT_ENROLL_LIMIT_TYPE`.

**Note**: After the structured data model was implemented, the business found the proposed CIM interface for entering them too complicated for their users. They decided they wanted the users to enter free text. The CIM Courses project timeline and resources did not allow for coding a process to translate the free text into the appropriate enrollment limit types. Since CIM is the main data entry point for enrollment limtations, only two types of limits are actually being used. All enrollment limits on non-freshman seminars are type 8 (OTHER, free text). Enrollment caps for freshman seminars are type 1 (Enrollment limited to number).

## Other Subject Template Tables

The remaining tables all relate back to the subject_template. They are:

- `SUBJECT_TMPL_ATTRIBUTE`- contains information about the attributes that are associated with a subject template
- `SUBJECT_TMPL_GMOD`- contains information about the grading mode(s) that are associated with a subject template
- `SUBJECT_TMPL_DISPLAY_CTRL`- contains information that control how a subject gets displayed in the Online Subject Listing
- `SUBJECT_TMPL_URL`- contains information about the URLs that are associated with a subject template
- `SUBJECT_TMPL_SEMINAR_FACULTY`- contains information about the faculty that are associated with subject templates for Freshmen Seminars

## Equivalencies (EQs) and Scheduling Relationships (SRs)

Two subjects are considered equivalent if there is substantial overlap in their content such that a student shouldn't receive academic credit for taking one if they have received credit for the other. Two subjects have a scheduling relationship if they are taught together for much or all of their class time.

Equivalencies (EQs) and Scheduling Relationships (SRs) differ from other subject concepts because they are not related to the Subject Template. Instead, EQs and SRs are related to the Subject Container. This is because the equivalencies and relationships are between the subjects themselves, not the particular state of a subject as represented by a subject template. The subject "Chemistry for Physicists" is equivalent to "Chemistry for Computer Scientists" as long as their content overlaps sufficiently. The equivalency or scheduling relationship is not affected by minor title changes, faculty changes, description changes etc. The subjects may no longer be equivalent if a substantial change is made to the content of one of the subjects. Likewise, departments could decide to no longer teach two subjects together. In these cases, the equivalency or scheduling relationship between the subject containers ends.

Equivalencies are recorded in `SUBJECT_CONTAINER_EQUIV`. Scheduling relationships are recorded in `SUBJECT_CONTAINER_RELATION`. Each equivalency or relationship has two records so the equivalency or relationship is represented in both directions (eg Container A is equivalent to Container B and Container B is equivalent to Container A). When an equivalency or scheduling relationship is ongoing, the thru_term of the corresponding record is '999999'. When it ends, the thru_term is set to the appropriate end term.

Both EQs and SRs between two subject must be reflexive, e.g. if Subject A is equivalent to Subject B, then Subject B must be equivalent to Subject A. Similarly, if Subject A has a scheduling relationship with Subject B, then Subject B must have a scheduling relationship with Subject A.

If more than two subjects are involved in an EQ or SR, then those relationships must be transitive. This means that if Subject A is EQ to Subject B and Subject C, then Subject B must be EQ to Subject A and Subject C, and Subject C must be EQ to Subject A and Subject B.

# Basic Functionality Flow

(Note that only the relevant fields are listed in each table)

### Subject Creation

- When a Subject is created (currently accomplished through CIM Courses, SCASUBJI, or the Cross-Registration functionality in AddDrop), a record is inserted into the Subject_Container table to represent the Subject.
- Subject_Container

| Subject_Container_ID | Subject_Type | Status | Last_Active_Term |
|---|---|---|---|
| ABC | Standard | Active | (null) |

- Next, a record is inserted into the Subject_Template table to represent the way the Subject will be offered between the Effective From Term and Effective Thru Term.
- Subject_Template

| Subject_Template_ID | Subject_Container_ID | Main_Subject_Code | Main_Subject_Number | Full Title | Effective_From_Term |
|---|---|---|---|---|---|
| 10002 | ABC | 1 | 100T | Archery | 2017FA |

- Registerable Instance Insertion
  - If the Subject is not cross-listed, a single record will be inserted into the Subject_Tmpl_Reg_Instance table
  - Subject_Tmpl_Reg_Instance

| Subject_Tmpl_Reg_Instance_ID | Subject_Template_ID | Subject_Code | Subject_Number | Is_Primary |
|---|---|---|---|---|
| RI001 | 10002 | 1 | 100T | Y |

  - If the Subject is cross-listed, a record for each Subject Number will be inserted into Subject_Tmpl_Reg_Instance. This includes the main subject number associated with the Subject Template.
  - Subject_Tmpl_Reg_Instance

| Subject_Tmpl_Reg_Instance_ID | Subject_Template_ID | Subject_Code | Subject_Number | Is_Primary |
|---|---|---|---|---|
| RI001 | 10002 | 1 | 100T | Y |
| RI002 | 10002 | 3 | 300T | N |

## Subject Editing

When Subjects are edited, new Subject Template and Registerable Instance records will be inserted and/or existing Subject Template and Registerable Instance records will be updated.

- The following is a common use case for editing a subject. The user makes a change to the "Full Title" field effective 2020FA that result in the creation of a new template. Existing Data:
- Subject_Container

| Subject_Container_ID | Subject_Type | Status | Last_Active_Term |
|---|---|---|---|
| ABC | Standard | Active | (null) |

- Subject_Template

| Subject_Template_ID | Subject_Container_ID | Main_Subject_Code | Main_Subject_Number | Full Title | Effective_From_Term |
|---|---|---|---|---|---|
| 10000 | ABC | 1 | 101T | Basket Weaving | 2013FA |
| 10001 | ABC | 1 | 101T | Basket Weaving | 2015FA |

- **User Edit**: The user changes the Full Title of the subject to "The Weaving of Baskets", effective 2020FA. Note that Subject_Template_ID 10001 gets "bookended" with an effective_thru_term of 2019SU and a new template (10002) gets inserted with the updated Full Title effective 2020FA-999999. Resulting data*:
- Subject_Template

| Subject_Template_ID | Subject_Container_ID | Main_Subject_Code | Main_Subject_Number | Full Title | Effective_From_Term |
|---|---|---|---|---|---|
| 10000 | ABC | 1 | 101T | Basket Weaving | 2013FA |
| 10001 | ABC | 1 | 101T | Basket Weaving | 2015FA |
| 10002 | ABC | 1 | 101T | The Weaving of Baskets | 2020FA |

*Note that Subject_Container is unaffected by this type of change. A new Subject_Tmpl_Reg_Instance is inserted that relates back to subject_template_id 10002.

## Subject Deactivation

When Subjects are deactivated, both the Subject Container and Subject Template tables are updated to reflect the deactivation. Consider the following example:

- Subject are generally deactivated effective in the proposal year - which for this case we will assume is 2020FA. Assume that the deactivation approval occurs when the current term is 2019FA. A Subject exists with the following Container and Template records:
- Subject_Container

| Subject_Container_ID | Subject_Type | Status | Last_Active_Term |
|---|---|---|---|
| ABC | Standard | Active | (null) |

- Subject_Template

| Subject_Template_ID | Subject_Container_ID | Main_Subject_Code | Main_Subject_Number | Effective_From_Term | Effective |
|---|---|---|---|---|---|
| 10000 | ABC | 1 | 101T | 2013FA | 2014SU |

| | | | | | |
|---|---|---|---|---|---|
| 10001 | ABC | 1 | 101T | 2015FA | 999999 |

- **User Edit**: The user deactivates the subject effective 2020FA. The resulting container and templates for the Subject will be as follows:
- Subject_Container

| Subject_Container_ID | Subject_Type | Status | Last_Active_Term |
|---|---|---|---|
| ABC | Standard | Active | 2019SU |

- Subject_Template

| Subject_Template_ID | Subject_Container_ID | Main_Subject_Code | Main_Subject_Number | Effective_From_Term | Effective |
|---|---|---|---|---|---|
| 10000 | ABC | 1 | 100T | 2013FA | 2014SU |
| 10001 | ABC | 1 | 100T | 2015FA | 2019SU |

- Note that the `Status` field retains a value of "Active" even though the deactivation is approved. This is because the *current* term is 2019FA; the subject will remain active through its `last_active_term` of 2019SU. It will be updated with a `Status` of "Inactive" when the current term changes to 2020FA.

## Subject Reactivation

When Subjects are reactivated, both the Subject Container and Subject Template tables are updated to reflect the reactivation. Consider the following example:

- A subject was deactivated effective 2017FA, so it's `last_active_term` was 2016SU:
- Subject_Container

| Subject_Container_ID | Subject_Type | Status | Last_Active_Term |
|---|---|---|---|
| ABC | Standard | Inactive | 2016SU |

- Subject_Template

| Subject_Template_ID | Subject_Container_ID | Main_Subject_Code | Main_Subject_Number | Effective_From_Term | Effective |
|---|---|---|---|---|---|
| 10000 | ABC | 1 | 101T | 2013FA | 2014SU |
| 10001 | ABC | 1 | 101T | 2015FA | 2016SU |

- **User Edit**:The user reactivates the subject effective 2020FA. The resulting container and templates for the Subject will be as follows:
- Subject_Container

| Subject_Container_ID | Subject_Type | Status | Last_Active_Term |
|---|---|---|---|
| ABC | Standard | Inactive | (null) |

- Subject_Template

| Subject_Template_ID | Subject_Container_ID | Main_Subject_Code | Main_Subject_Number | Effective_From_Term | Effective |
|---|---|---|---|---|---|
| 10000 | ABC | 1 | 100T | 2013FA | 2014SU |
| 10001 | ABC | 1 | 100T | 2015FA | 2016SU |
| 10002 | ABC | 1 | 100T | 2020FA | 999999 |

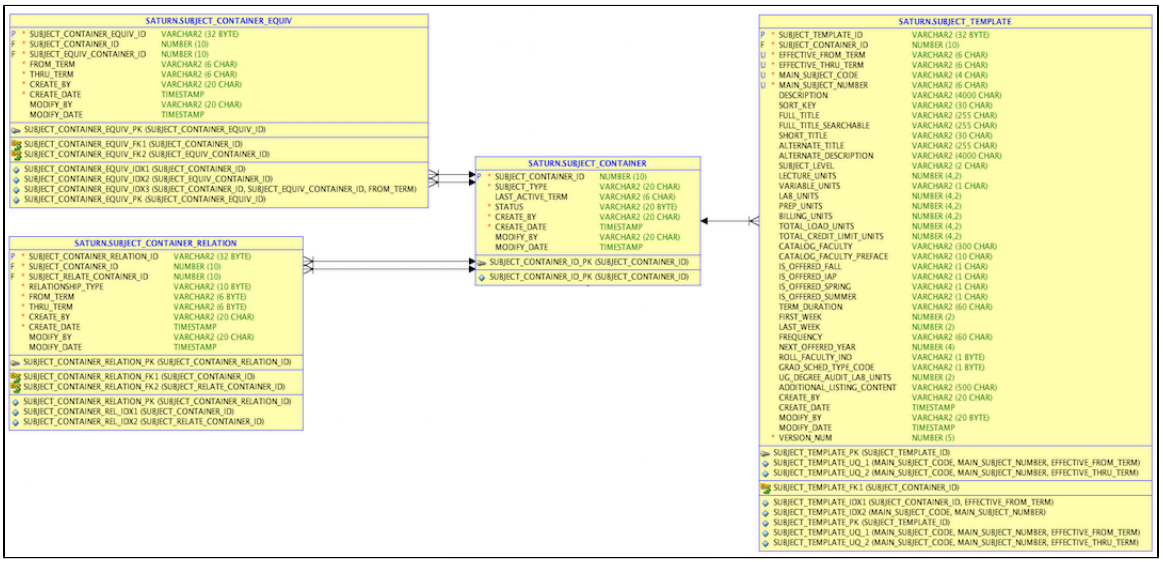
- Note that the `Status` field retains a value of "Inactive" even though the reactivation is approved. This is because the *current* term is 2019FA; the subject will remain inactive until the *current* term is equal to the `effective_from_term` of its newly created template (2020FA). It will be updated with a `Status` of "Active" when the current term changes to 2020FA.

## Complex Use Cases/Test Cases

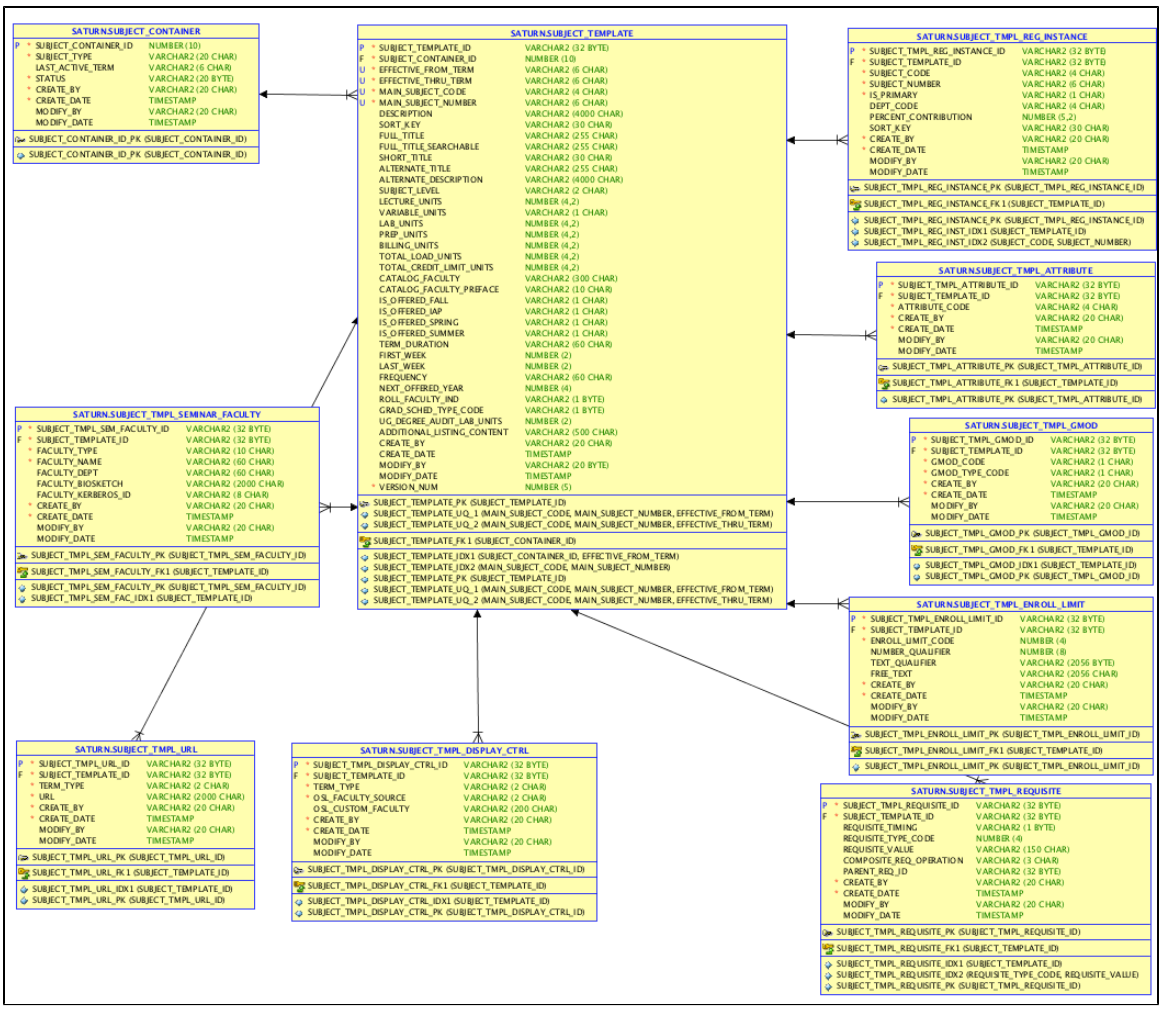
- The purpose of the use cases outlined above is to give the reader a basic understanding of how the CTSS behaves as a result of certain user actions. If you are interested in how the CTSS behaves in more complex scenarios (and also how data is backfilled to legacy systems based on those changes), the automated test cases developed as part the CIM Courses project should serve as a valuable resource: Subject Management Automated Test Case Index.

# Data Models

## Subject Container Data Model




## Subject Template Data Model



# System Rules

- A Subject Container must always have at least one template.
- A Subject Container cannot be created with any status other than "Active".
- The Subject Template associated with a new Subject Container must have an Effective Thru Term of "999999".
- For any Subject Container, the Effective From Term and Effective Thru Term of its associated Subject Templates cannot overlap.
- A Subject Container's Templates will be contiguous (no gaps) unless the Container has been Deactivated or Reactivated.
- Changes to a Subject Template's Effective Thru Term will be handled by the system logic (with the possible exception to Subject Deactivation).
- If a Subject is Inactive, it will not have any templates past its deactivation term.
- A subject cannot be deactivated if any students are registered for the subject.
- Equivalencies between two containers cannot begin before the earliest template for either container.
- Inactive containers can be part of equivalencies.
- Equivalencies are reflexive. If Container A is equivalent to Container B then Container B is equivalent to Container A for the same period of time.
- Equivalencies are transitive. If Container A is equivalent to both Container B and Container C for some period of time, then Container B must be equivalent to Container C and Container C must be equivalent to Container B for the same period of time.
- There can be more than one SUBJECT_CONTAINER_EQUIV record for the same combination of SUBJECT_CONTAINER_ID and SUBJECT_EQUIV_CONTAINER_ID. (This would occur if the two subjects stopped being equivalent a some point and then later became equivalent again.). The effective periods of the records must not overlap.
- Scheduling relationships between two containers cannot begin before the earliest template for either container.
- Inactive containers cannot be part of active scheduling relationships.
- Both containers must have templates for the entire period of the scheduling relationship.
- Scheduling relationships are reflexive. If Container A has a scheduling relationship with Container B then Container B must have a scheduling relationship with Container A for the same period of time.
- Scheduling relationships are transitive. If Container A has a scheduling relationship with both Container B and Container C for some period of time, then Container B must have a scheduling relationship with Container C and Container C must have a scheduling relationship with Container B for the same period of time.
- There can be more than one SUBJECT_CONTAINER_RELATION record for the same combination of SUBJECT_CONTAINER_ID and SUBJECT_RELATE_CONTAINER_ID. (This would occur if the two subjects stopped meeting together a some point and then later began meeting together again.). The effective periods of the records must not overlap.

# Other Documentation

## Expected Behavior When A New Subject is Approved and Deactivated in the Same Proposal Term

When a subject is approved and then deactivated in same proposal term (e.g. 2019FA) the expected behavior in each environment differs. The expected behavior is as follows:

1. CTSS
    a. The container, its templates and related records are deleted
    b. EQs and SRs are deleted
    c. XML for the subject is stored in `SUBJECT_DELETED` so that the XML can be inserted into the CIM Outbound Feed to get the subject to rest
2. CIS
    a. SCRCI_PROPOSAL thru_term is set to one minus the effective term
3. MITSIS
    a. SCRSU_VAR record will be set to inactive

# Subject Management Documentation Index

The Subject Management Documentation Index is the central listing for documentation pertaining to Subject Management.