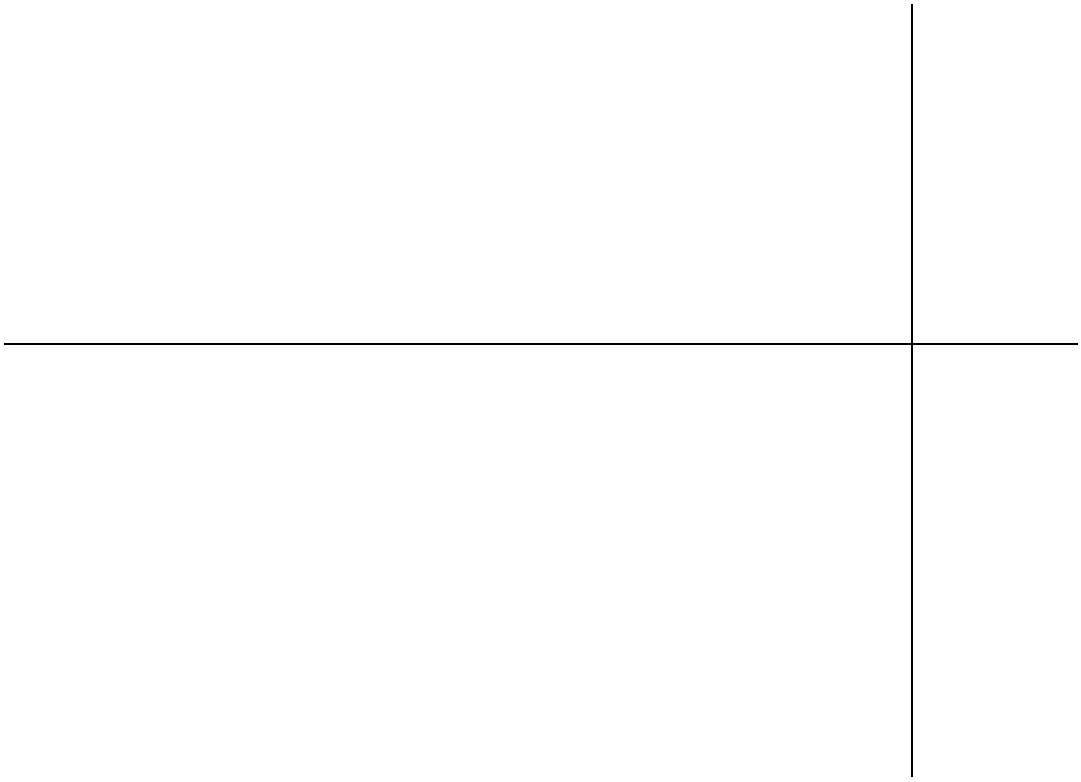
SNOMED Clinical Terms® Reference Set Specification

EXTERNAL DRAFT FOR COMMENT
January 2008



IMPORTANT NOTE

THIS DOCUMENT IS AN *EXTERNAL DRAFT FOR DISCUSSION*. IT HAS BEEN APPROVED FOR CIRCULATION FOR COMMENT ONLY.

THIS DOCUMENT WILL BE AMENDED FOLLOWING CONSULTATION AND SHOULD NOT BE USED AFTER THE CLOSE OF THE CONSULTATION PERIOD.

Table of Contents

Document History

The SNOMED CT Reference Set specification document would be released as a draft specification for external comment in October 2005.

Version	Notes
21-May-04	Original draft based on TWG proposals
13-Jan-05	Revision of original draft with additional notes drawn from TWG minutes
24-Apr-05	Changes from the TWG meeting of 04/11/2005
24-May-05	Changes from the TWG meeting of 04/28/2005
02-Jun-05	Language RefSet Proposal derived from original generic RefSet proposal
06-Jun-05	Structure overview diagram added, correction of RefSetVersionId field name, inclusion of a new field (priority)
14-Jun-05	Changes from TWG meeting of 06/06/2005 and subsequent discussion
24-Aug-05	Changes from TSC meeting of 06/15/2005 to Language Refsets
06-Sep-05	Added chapters for simple and navigation RefSets.
26-Sep-05	Changes from TWG meeting of 09/07/2005
05-Oct-05	Changes for Editorial Board Meeting
15-Mar-06	Changes after Technical Design Team meeting - Revision of history mechanism proposal - Transference of some enumerated DesignationType values to the new field Degree of Synonymy in the Language Refset member file. - Updated figures
01-Jun-06	Changes after April Technical Design Team meeting - History tables for RefSet and Members table adopt the new approach for generic history tracking using state-valid log tables - Removal of RefSetVersion table since it is not necessary with the new history tracking and component versioning system.
July 2007 (31-Jul-07)	- Changes to make the draft consistent with the RefSet/History mechanism Technology Preview documentation - RefSet history tracking content has been removed, since RefSets would use the generic, state-valid enhanced history mechanism proposal as documented elsewhere. - Document updated to reflect the transfer of SNOMED CT to the International Health Terminology Standards Development Organisation (IHTSDO)
January 2008	- Copyright statements updated to 2008 - No other changes are included in this version

1 Introduction

1.1 Purpose

The purpose of this document is to inform wider discussion about the proposed Reference set mechanism, an evolution of the original SNOMED CT subset specification.

1.2 Who should read this guide?

The intended audience for this document is any individual or organization that wishes to develop or apply systems that will use SNOMED Clinical Terms. Comments are especially welcome from users/developers developers who have already implemented SNOMED CT into systems.
This document is mainly for:

- Clinical software developers - who have worked with SNOMED Clinical Terms and others who may be planning to build clinical systems.
- Clinicians – with an interest in health informatics.
- Health informatics specialists – who may be asked to advise on the implementation, and migration towards a new terminology, within new and existing systems.
- Planners, managers and information specialists – who will be involved in both local (and wider) implementations.
- Government or private sector organizations with a need for a controlled healthcare terminology.

1.3 Status

This document is regarded as an External Draft and is subject to revisions and extensions without notice. However, all revisions will be documented to help those already evaluating the specification

1.4 Feedback

Further information about SNOMED CT is available on the Internet at:

www.ihtsdo.org

Please send feedback by email to:

support@ihtsdo.org

or contact the International Health Terminology Standards Organisation at:

IHTSDO
Rued Langgaards Vej 7, 5te
DK-2300 Copenhagen S
Denmark
Tel: +45 36 44 87 36
Fax: +45 44 44 87 36

2 Introducing Reference Sets

2.1 Overview

The SNOMED CT subset mechanism was provisionally adopted by the SNOMED International Editorial Board in late 2001. As originally planned, a revision of the mechanism has been done based on comments and implementation experience. The original requirements were refined to address the increasing usage of SNOMED in a range of diverse applications and the need to advance the internationalization of the core SNOMED CT design. The current Reference Set specification is an evolution of the original subset mechanism that enhances the ability to localize SNOMED CT to accommodate diverse user preferences and use cases.

The Reference Set proposal also includes recommendations for the evolution of other SNOMED CT elements, notably the Descriptions table, that are either necessary for the Reference Set proposal or provide improvements to SNOMED CT that would be made possible by its adoption. In the rest of this document the abbreviated form “RefSet” may be used to make references to Reference Sets.

2.2 Basic principles

The effective usage of SNOMED CT requires a way to refer to sets of components that are appropriate for a specific use case. The subset mechanism may be used to filter and arrange the concepts for display, and also for selecting the descriptions most suitable for a particular scenario.

The Reference Set specification incorporates changes and additions to the subset mechanism. The enhancements are based on the following set of principles:

2.2.1 Improve the internationalization of the SNOMED CT design

The distinction between internationalization and localization is important. Internationalization is the adaptation of products for potential use globally, while localization is the addition of special features for use in a specific region or realm.

2.2.2 Increase the support for SNOMED CT localization

SNOMED CT localization efforts would be facilitated by the concentration of language sensitive components in a separate layer, and by providing a mechanism to adapt navigation and aggregation of SNOMED CT content to fit diverse user needs while preserving the integrity of the core.

2.2.3 Enhance change management support

SNOMED content is distributed as snapshots of the current component status at a particular release date. The original SNOMED CT history mechanism does not include change tracking support for subsets and their membership. This Reference Set specification incorporates enhancements to the change tracking mechanism that are aligned with the enhancements recommended to the overall SNOMED CT history mechanism. These overall enhancements support component-level versioning by enabling the release of valid states and effective times for components. The status of each RefSet member and its attributes can be tracked over time at the maximum level of granularity. The enhancements will enable incremental updates of

SNOMED content since last synchronization, and facilitate time-sensitive queries for point in time retrieval of the status of each component.

The release content can be delivered using state-valid tables that will include all valid (released) versions of each component. Applications would then extract snapshots of that content to fit the purposes of their implementation.

2.2.4 Facilitate the use of Refset groups

Certain use cases require the combination of different kinds of RefSets to accomplish the desired effect. For example a set of concepts that are frequently used in a particular specialty would be more usable if they are arranged in an intuitive navigational hierarchy, and eventually privileging the specialty preferences over the default language RefSet.

2.3 Background

Experience with earlier versions of SNOMED and with NHS Clinical Terms has indicated that while comprehensive terminologies are valuable, their size and breadth of scope can pose a challenge for users and implementers. There are many situations in which it is useful to limit the set of Concepts and/or Descriptions that are readily available. The multinational, multilingual nature of SNOMED CT adds to the size and complexity of the terminology, increasing the need for limiting access to sets of components appropriate within a particular setting.

Experience since the first release of SNOMED CT suggests that the subset mechanism is an important element in making SNOMED CT implementable. However, there have been several cases where it has seemed that the mechanism could be enhanced and made easier to understand with a few relatively modest changes.

The current mechanisms have been shown to work and deliver significant benefits to SNOMED CT in terms of:

- Language (English, Spanish and German)
- Dialect (US and GB English)
- Realm Concept Subsets (Non-human, UK GP, Pathology)
- Navigational (Top-level and CTV3 hierarchies)

While the subset mechanism has gained broad acceptance by the user community, the original design had some limitations:

- Only active members of the subset were included in the Subset members table. Changes in membership composition were not registered in the Component history table.
- The approach brought together many different requirements and addressing these with a single file structure, tended to overload fields within that structure. The subset members table field MemberStatus was overloaded, having different meanings for each subset type.
- Experience with navigational subsets suggested more details were needed at the node level to support complex navigational use cases.
- Subset groups and the ability to define a subset based on another subset were only supported in the XML distribution format.

2.4 Proposed Reference set types

Types of RefSets proposed in this document:

- Simple RefSet – selects a set of elements from SNOMED CT
- RefSet Group – groups RefSets that are designed to work together.
- Tagged RefSet – facilitates the marking of RefSet members
- Language RefSet – identifies elements and their role for a specific language
- Navigation RefSet – provides an alternate navigational hierarchy for the selected elements
- Aggregation RefSet – provides an alternate aggregation hierarchy for the selected elements

2.5 Relational representation of RefSets – summary

The RefSet mechanism has two tables to store the RefSets and the RefSet members. The structure of the RefSet table is the same for all RefSet types, and this specification includes five variants of the file structure of the RefSet Members table. The basic structure of the RefSetMember table support the distribution of RefSet groups, Simple and Tagged RefSets, while the other RefSet types require a specialization of the Members table structure to support additional attributes. Additional RefSet types might be defined in complementary specifications.

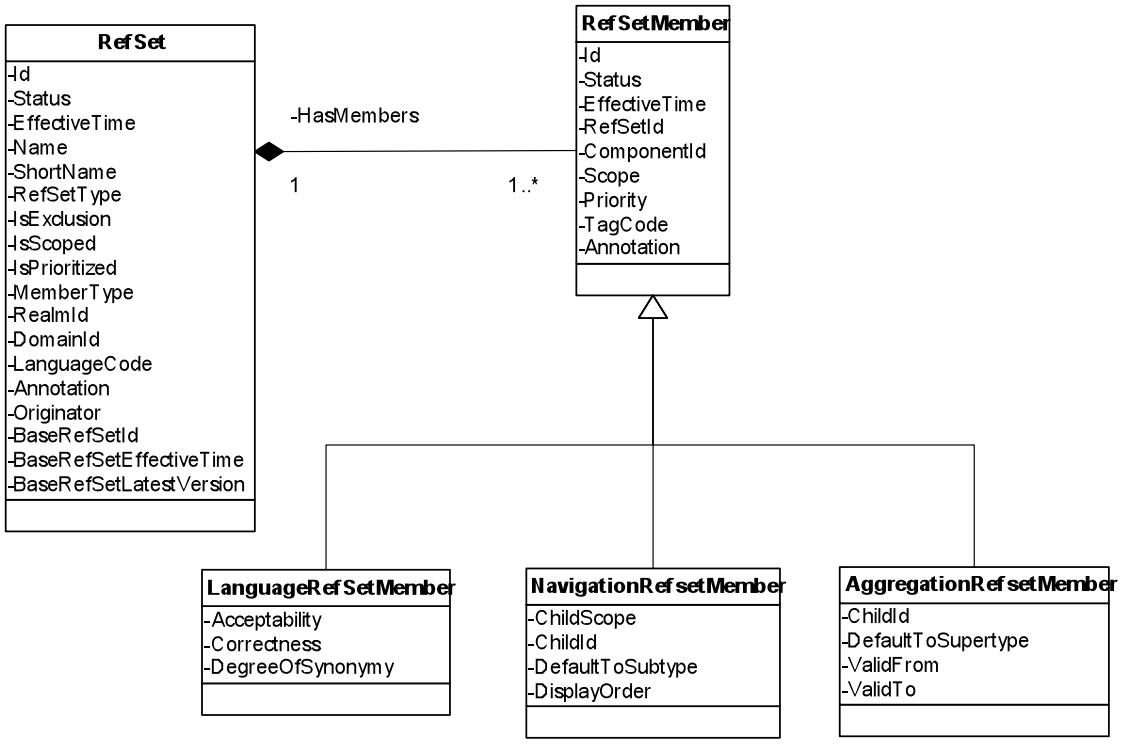


Figure 1 - The Reference Set Mechanism model

3 RefSets

3.1 Overview

The original Subset mechanism included a Subset table to store information about each subset version. A new Subset Id was issued for each new release (version) of the subset, and the field OriginalSubsetId stored the Id of the first version. The subset member table only included active members.

This versioning scheme has been replaced in the RefSet mechanism by log tables that enable the tracking of valid (released) component states over time.

The RefSet table includes a row for each valid state of a RefSet that has been released over time. The field EffectiveTime stores the Universal Time (UTC) at which the component representation has become effective (published). If the attributes of the RefSet row have no changed in a new release, the state remains valid and the value of the EffectiveTime field is maintained. This table is therefore a history table itself for the purpose of tracking RefSet attributes over time.

The RefSetMember table (and the related specializations described later) also includes a row for each valid state of a RefSet member and the corresponding EffectiveTime.

3.2 The RefSet table structure

The RefSet table includes information about each RefSet. The RefSetId is functionally equivalent to the OriginalSubsetId field in the Subset specification. Published subsets that would eventually be released in the new RefSet format will keep the OriginalSubsetId value as the RefSetId. The component identifier of a RefSet would be stable once it has been assigned.

RefSets Table:

Field Name	Type	Notes	
Id	SCTID	Identifies the Reference Set, using the same partitionId allocated originally to SubsetIds (3).	M
EffectiveTime	String[14]	Universal Time (UTC) at which the component representation becomes effective. The component representation remains valid until it is superseded. The date and time should be represented using ISO 8601 conventions with no punctuation and no letter T between date and time, with a granularity up to the level of seconds.	M
Status	Enum	Status of the RefSet record	M
Name	String[255]	Name of the RefSet	M
ShortName	String[20]	Short name for the RefSet	R
RefSetType	Enum	1= Simple; 2=Prioritized; 3=Navigation; 4=Tagged; 5=Language; 6=RefSet Group 8=Aggregation **NOTE: It has been suggested that this set of enumerated values should be changed to make them consistent with the allowed values for Subset Type in the original specification. If this suggestion is accepted, the enumerated values would change to:**	M

Field Name	Type	Notes	
		1= Language; 7=Navigation; 9=Simple; 10=Tagged; 11=RefSet Group 12=Aggregation	
IsExclusion	Boolean	0=RefSet members are included 1=RefSet members are excluded	0
IsScoped	Boolean	0=Not Scoped 1=Scoped (members must have the Scope property)	0|1
IsPrioritized	Boolean	0=RefSet member rows are not prioritized 1=RefSet member rows are prioritized	0|1
MemberType	Enum	0=Concept, 1=Description, 2=Relationship, 3=RefSet (only applicable to RefSet groups), 6=CrossMap 7=RefSetMembers (only applicable to Tagged RefSets)	1
RealmId	SCTID	Country or other jurisdiction for which this RefSet applies (null=applies for all). It is a SNOMED CT concept.	R
DomainId	SCTID	Discipline, specialty or business area for which this RefSet applies (null=applies for all). It is a SNOMED CT concept.	R
LanguageCode	String[12]	Language for which this RefSet applies	M
Annotation	String[255]	Text to associate with the RefSet	R
Originator	SCTID	Namespace concept for the creator of the RefSet	R
BaseRefSetId	SCTID	Foundation RefsetId on which this Refset Version is based	R
BaseRefSetIdEffectiveTime		Effective time of the foundation Refset state on which the Refset is based.	R
BaseRefSetLatestVersion	Boolean	0=Use the specified BaseRefSetIdEffectiveTime of the Base RefSet members 1=Use the latest version of the base RefSet members if a more recent version is available	0|1

Notes:

RefSetId – each RefSet is given a unique RefSetId. For RefSets that include content previously released in Subset format, the SubsetOriginalId (the SubsetId of the first release of the Subset) is used as RefSetId.

RealmId, DomainId – it is proposed that SNOMED CT concepts will be used as the values for these fields.

LanguageCode – consists of a code and optional sub-code, separated by a dash. Code=ISO639-1 language code; sub-code=ISO3166 country code (or IANA code for countries not listed in ISO3166). If the LanguageCode value=0, the RefSet applies to all languages – however, this is not a valid value for Language RefSets.

BaseRefSetId – identifies a foundation RefSet on which the RefSet is based. The original Subset Definition File in XML format allowed the specification of such a base subset to facilitate the definition of new subsets by only describing the differences or exceptions. This feature is now included in the relational RefSet distribution. All Base RefSets will need to be considered at run-time to determine members. When creating a RefSet quality assurance should be done to ensure it does not contain cyclical references. Membership in a child RefSet overrides membership in a base RefSet.

BaseRefSetEffectiveTime – Effective time of the foundation RefSet state on which the RefSet is based

Scope - Some exclusions, inclusions and prioritizations may be fixed for a given installation, configurable by user login or user selection. Others may apply to specific data entry fields, templates or protocols. In these cases, the context in which each set of components is included, excluded or prioritized needs to be represented. Implementations need to support dynamic real time context-sensitive switches in these constraints. In the current subset mechanism, context subsets are intended to fulfill this role. In the proposed RefSet mechanism, this role is fulfilled by scoped RefSets.

Priority - Prioritized Reference Sets would allow the prioritization of certain components over others according to the intended usage. For example, a user may prefer to see the ten most common procedures they perform at the top of a pick list.

3.3 The RefSet members table structure

The members table includes a row for each valid state of a member of a RefSet released at a specific time. In addition to creating new functionality, the design goal was to remove the ambiguity created by having multiple uses for the MemberStatus and LinkedId fields in the Subset Members table. The structure of RefSet members table is specified for each type of RefSet in the forthcoming sections. Several RefSet types use the basic RefSetMember table, while other require specialization of that structure to accommodate additional fields.

- For more details see *RefSetMembers_SV*

Column	Description
id	The unique *SNOMED CT* Identifier (*SCTID*) assigned to this *RefSetMember*.
effectiveTime	The date and time from which this representation of the identified *RefSetMember* applied.
status	The *status* of the *RefSetMember* from the specified *effectiveTime*.
refSetId	The unique identifier of the *RefSet* to which this member belongs.
componentId	The unique identifier of the *Component* referenced by this member.
tagCode	A coded attribute associated with this member.
annotation	A free-text string annotation associated with this member.
scope	A string identifying the *scope* within which this member applies.
priority	The relative *priority* of the member in the set.
acceptability	Indicates whether the *term* of the referenced *Description* is preferred or accepted by users as a description of the associated *Concept*. The *Preferred Term* for a *Concept* value is indicated by *acceptability* = 1(Preferred). Recognized synonyms are indicated by *acceptability* = 2 (Acceptable).
correctness	Indicates whether the *term* of the referenced *Description* is a linguistically correct description of the associated *Concept* according to the official Naming Authority for the language, dialect or realm.
degreeOfSynonymy	Indicates the degree to which the *term* of the referenced *Description* covers the same concept as the *fullySpecifiedName* attribute of the associated *Concept* (i.e. the primary, original *Fully Specified Name*).
childScope	The scope within which the *Component* referred to by the *childId* attribute is to be interpreted.
childId	Refers to a *Component* that is intended to behave as a hierarchical child of the *Component* referred to by the *componentId*.
defaultToSubtype	Indicates whether the logical subtypes of the *Component* referred to by the *componentId* attribute should be treated as Navigational children of that *Component*.
displayOrder	Determines the order in which to display the *Component* referenced by *childId* when showing children of the *Component* referenced the *componentId* attribute. The items with the lowest *displayOrder* are shown first.
defaultToSupertype	Indicates whether the *Concept* referred to by the *child* attribute should be aggregated under its logical supertypes if it has no specified aggregation parents.
validFrom	Specifies a date and time from which this aggregation is valid.
validTo	Specifies a date and time after which this aggregation is no longer valid.

The columns required to be present or used in a RefSetMembers distribution file vary according to the type of RefSet to which the members in that file apply.

The next table shows the relationships between the complete set of columns permitted in the RefSetMembers file and the type of RefSets to which the members apply. Note that even when the use of a column is "prohibited" is may be present. Thus it is possible for all RefSetMembers to be included in a single file. However, "prohibited" columns may be omitted to simplify the distribution of particular types of RefSetMember. It is also possible (though not recommended) to omit columns that are "optional". However, in this case omission is only possible if the row is not applicable to any of the members.

Table. Applicability of RefSetMember columns to different RefSet types

	Applicability based on refSetType			
Column	**Simple**	**Language**	**Navigation**	**Aggregation**
id	M	M	M	M
effectiveTime	M	M	M	M
status	M	M	M	M
refSetId	M	M	M	M
componentId	M	M	M	M
tagCode	O	O	O	O
annotation	O	O	O	O
scope	{M:isScoped=1}{P}	{M:isScoped=1}{P}	{M:isScoped=1}{P}	P
priority	{M:isPrioritized=1}{P}	P	P	P
acceptability	P	M	P	P
correctness	P	O	P	P
degreeOfSynonymy	P	O	P	P
childScope	P	P	{M:isScoped=1}{P}	P
childId	P	P	{M:defaultToSubtype=Null}{O}	M
defaultToSubtype	P	P	M	P
displayOrder	P	P	M	P
defaultToSupertype	P	P	P	M
validFrom	P	P	P	O
validTo	P	P	P	O

Applicability Key

M=Mandatory – the column SHALL be present and SHALL contain a value.

O=Optional – the column SHOULD be present and MAY contain a value.

P=Prohibited – the column MAY be present but SHALL NOT contain a value.

{*applicable*:*criterion*}=Conditional – The *applicable* value applies if the *criterion* is satisfied.

Example {M:isPrioritized=1}{P} Means if *RefSet.isPrioritized* = 1 then Mandatory but otherwise Prohibited.

4 Simple Reference Sets

4.1 Overview

The requirements of the following subset types, described in the original subset specification could be addressed with the simple Reference Set table structure:

- Realm concept subset (the concepts applicable for a particular Realm)
  - A realm concept subset refers specifically to concepts, but it also affects the availability of descriptions and relationships. Only descriptions and relationships associated with the included concepts are allowed. Descriptions and relationships not associated with concepts in the subset are not referenced.
- Realm relationship subset (the relationships applicable for a particular Realm)
- Context concept subset (now a Scoped Simple RefSet)

Please note that functionality provided by realm description subset and context description subsets is now intended to be provided through Language Refsets.

When a Simple RefSet refers specifically to concepts, it also affects the availability of descriptions and relationships. Navigational RefSets (described later) could also serve the purpose of limiting availability of concepts and descriptions while providing alternate navigational links between concepts.

4.2 Relational representation of Simple Reference set members

Each valid status of a Simple Reference Set member is represented by a row in the RefSetMember table.

Field Name	Type	Notes	
Id	SCTID	Unique SCT identifier for the RefSet member row. It uses the partitionId 7, that identifies RefSet members.	M
EffectiveTime	String[14]	Universal Time (UTC) at which the component representation becomes effective. The component representation remains valid until it is superseded. The date and time should be represented using ISO 8601 conventions with no punctuation and no letter T between date and time, with a granularity up to the level of seconds.	M
Status	Enum	0=Current, 1=Non Current	M
RefSetId	SCTID	Refset to which the component reference belongs	M
ComponentId	SCTID	Unique SCT identifier of the referenced component	M
Scope	String[20]	Context in which this member row applies	C
Priority	Integer	Relative priority of the member in the set	M
TagCode	String[20]	Coded attribute to associate with this member	R
Annotation	String[255]	Text to associate with this version of the RefSet	R

5 RefSet Groups

5.1 Overview

Experience gathered during the development of subsets for specific specialty domains demonstrated that a suite of coordinated subsets may be needed to provide the desired functionality.

- For example, a specialty Simple Refset would facilitate the filtering of the content to concepts relevant to the domain. A Language RefSet could be used to store the preferences for the display of descriptions and to provide intuitive navigation of the content, a companion Navigational RefSet may be convenient.
- The requirements for efficient browsing of the content of a Simple RefSet may include the ability to change the "view" depending on the user role or task. This could be accomplished with a group of Navigational and Language RefSets working together.

5.2 Relational representation of RefSet groups

RefSet groups could be represented using the Simple RefSet members table structure. In this case, the referenced component is a RefSet identifier.

6 Tagged RefSets

6.1 Overview

- In some scenarios, it is useful to flag members of a RefSet with user-defined tag codes. This can be accomplished using the Simple RefSet Member table structure. The referenced members are then Ids

6.2 Relational representation of Tagged RefSets

Tagged RefSets could be represented using the Simple RefSet table structure, but the member types could also include RefSet members in addition to other SNOMED CT component types; a non null value is mandatory for the field TagCode.

Note that the TagCode field is a free text field. Alternatives for the provision of a more structured definition of allowable codes for this field were examined. Since Tagged RefSets could be useful in very diverse scenarios, additional constraints were avoided. Content of this field is thus implementation specific. This decision may need revision according to user feedback.

A potential use of this RefSet type is the definition of value, using the TagCode field to store the value set identifier. This usage would require further constraints and/or explicitly enumerated value set identifiers to facilitate data set exchange.

7 Language Reference Sets

7.1 Overview

SNOMED CT is designed as a multilingual terminology. The Descriptions Table will contain Descriptions for every Concept in all supported Languages and Dialects.

A user or groups of users normally only require Descriptions in their own Language. The preferences for particular descriptions within the Language would depend on many factors e.g. dialects, regional and cultural aspects, user specialty, local variants, and intended use.

The RefSet mechanism extends the SNOMED CT potential for localization by clearly separating language (descriptions) from language usage (local preferences for particular descriptions). To achieve this, preferences would be stored in the Language RefSet. To resolve the potential overlap with features previously provided by the Descriptions table, slight modifications of the table are proposed for the distribution format and the enumerated values in certain fields. Those modifications would be backward compatible with the original distribution format.

7.2 Requirements

7.2.1 Requirement summary

- Some Descriptions may be preferred in one dialect or setting and but not acceptable in another dialect of the same language.

- A Language RefSet should refer to each Description that is used in a given language, dialect or local variant and state the appropriate preferences for language usage in that setting.

- User preferences may favor the usage of terms that are discouraged or considered incorrect by a Language Authority. Terms borrowed from other languages for example anglicisms, galicisms and Latinisms are used in everyday practice. These user preferences (or actual language usage) should be distinguished from the language Authority recommendations.

- While synonymy is usually relative, it is expected that terms that are true synonyms could be exchanged without affecting the meaning of the phrase. In restricted implementation contexts, there is a need to support terms that are beyond the specific context of use, which would be ambiguous or more general.

- In certain implementation scenarios that involve Navigational RefSets, there is a need to support context specific descriptions that otherwise could not be considered synonyms of the associated concept. Those descriptions should not be used in search indexes, and should only be displayed when referenced in conjunction with a navigational RefSet within an explicit scope.

- A Language RefSet should keep references to Descriptions that have been previously published but become non active members of the RefSet.

- The same mechanism used for the distribution of Language Editions should be available to implementations that wish to localize the linguistic preferences at the regional, institutional or even user level.

7.2.2 Current approach

- The Descriptions table stores terms with three description types: Fully Specified Names (FSN), Preferred Terms, and Synonyms. The LanguageCode field stores Language and Dialect information.

- Language subsets define what descriptions are included in a particular Language or dialect distribution and the description type specific to the subset, which might eventually override the "Preferred term" defined in the Descriptions table.

- Changes in the Descriptions table are recorded in the Component History table. But changes in the membership status of a subset are not tracked since the subset identification does not include unique Ids for member rows or membership status.

7.2.3 Proposed approach

The proposed approach involves a combination of the following:

- The *RefSet* mechanism
- The revised approach to component history tracking to enable dialect preferences to be effectively maintained.
- Changes to the *Descriptions Table* to disentangle dialect and preference specific information from this core table.

Omitting any of these elements would significantly impair the effectiveness of the proposed approach.

7.3 Language Reference set member table structure – summary

Field Name	Type	Notes	
Id	SCTID	Unique SCT identifier for the RefSet member row. It uses the partitionId 7, that identifies RefSet members.	M
EffectiveTime	String[14]	Universal Time (UTC) at which the component representation becomes effective. The component representation remains valid until it is superseded. The date and time should be represented using ISO 8601 conventions with no punctuation and no letter T between date and time, with a granularity up to the level of seconds.	M
Status	Enum	0=Current, 1=Non Current	M
RefSetId	SCTID	Refset to which the component reference belongs	M
ComponentId	SCTID	Unique SCT identifier of the referenced component	M
Scope	String[20]	Context in which this member row applies	C
Priority	Integer	Relative priority of the member in the set	M
TagCode	String[20]	Coded attribute to associate with this member	R

Field Name	Type	Notes	
Annotation	String[255]	Text to associate with this version of the RefSet	R
Acceptability	Enum	Acceptability of the description to users: 0=Not specified,1=Preferred, 2=Acceptable, 3=Not recommended, 4=Not acceptable, 5=Invalid	M
Correctness	Enum	Correctness of the member in the language: 0=Not specified, 1=Recommended, 2=Acceptable, 3=Not recommended, 4=Incorrect, 5=Invalid	R
DegreeOfSynonymy	Enum	Degree to which a term covers the same concept covered by the original, primary Fully Specified Name: 0=Not specified, 2=Synonymous, 3=Near synonymous (depending of context of use), 4=Non synonymous, 5=Invalid	R

Id – a SCTID that uniquely identifies the component as a member of the RefSet; the Id is retained across versions of the RefSet. This is not the component's own unique SCTID within SNOMED CT, which is indicated in the ComponentId field.

Status – once a component has been released as a member of a Language RefSet, it persists as a member of every subsequent version to enable interoperability among versions. If the reference is retired (the description is no longer considered as an active reference) then the RefSetMemberStatus would record the new status of the reference. The status of the referenced component is tracked in the corresponding component table. Descriptions with LanguageAcceptability of "Invalid" will always be "Not Current".

Scope – this property defines the context in which this member-row is valid as part of a scoped RefSet. Within a scoped RefSet, a component may appear as a member more than once. Each instance will have a unique Id and must have a unique Scope value.

ComponentId – the SCTID that uniquely identifies the component within SNOMED CT.

Acceptability – indicates the user preferences (within the dialect/realm) for descriptions of a concept.
Allowed values for Acceptability are:

0 = **Not specified** – the acceptability of the referenced term is not specified in the RefSet or is not considered relevant.
1 = **Preferred** – indicates the most common or accepted designation for a concept (within the realm). There may be only one Preferred designation of each DescriptionType (see *Changes to the Descriptions Table*, below)
2 = **Acceptable** – accepted by most users as a valid representation of the concept. Equivalent to a synonym in the original descriptions table.
3 = **Not recommended** – understood as a valid representation of the concept, but not frequently used and possibly carrying undesired connotations
4 = **Not acceptable** – not a valid designation in this realm or dialect, although it may be valid in other, related realms that share the same Language.
5 = **Invalid** – not a valid designation for the concept; it may have previously been published as a valid designation. Descriptions that are retired at the Language

level (Descriptions table) would become invalid in all Language RefSets referencing that description.

Please note that traditionally only positive preferences have been considered ("Preferred terms"), and those preferences were usually perceived as an attribute of the description. This proposal considers that preferences depend on the usage of the language rather than on the language itself. In addition, negative preferences are also recorded in this field. For example, in a certain region, a term that is acknowledged as a valid designation for the concept in the language may be considered not acceptable at the local level.

Correctness – indicates the status of the description as determined by the official Naming Authority for the language/realm, which may or may not correspond with user preferences (Acceptability). If no Authority exists for the language/realm, Correctness=0 (not specified). Allowed values for Correctness are:

0 = **Not specified** – the correctness of the referenced term is not specified in the RefSet or is not considered relevant.
1 = **Recommended** – indicates the preferred designation for a concept (according to the Authority).
2 = **Acceptable** – accepted by the Authority as a valid designation of the concept (agreed upon as an alternative)
3 = **Not recommended** – Discouraged by the Naming Authority, however it may be acceptable to users.
4 = **Incorrect** – the linguistic expression is not accepted by the Authority, however it may be acceptable to, and understood by, users
5 = **Invalid** – not a valid designation for the concept; it may have previously been published as a valid designation. Descriptions that are retired at the Language level (Descriptions table) would become invalid in all Language RefSets referencing that description.

Degree of Synonymy – Degree to which a term covers the same concept as the primary, original Fully Specified Name. There is a tension between the need to accommodate expressions common in natural language and a more restrictive policy to only include actual synonyms. A central role in this dilemma is played by the representation of context. In natural language, the context in which an expression is used might modify the intended meaning.

Allowed values for Degree of Synonymy are:

0 = **Not specified** – the Degree of Synonymy of the referenced term is not specified in the RefSet.
2 = **Synonym** – a term that represents the same concept as the original Fully Specified Name in the American English Edition.

Two terms are considered synonyms if they represent a single concept, within a specific semantic category as the only additional context. Ideally, synonyms should unequivocally represent a concept regardless of the context; however, it is not uncommon to find a term representing two different concepts. For the purposes of this specification, two terms will be considered synonyms provided they represent the same concept in a specific semantic category (e.g. <dressing> in <devices>, <procedures>, <findings>, etc.).

Two terms are considered synonyms provided the concept represented by one of them in a statement does not vary if the terms are interchanged (i.e. the first term is replaced by the second one). This interchangeability property should be assessed in most use cases.

Two terms are considered synonyms provided they share the same level of granularity for a concept. In the case where one term should be more specific than the other, they might be representing different concepts.

Two terms are considered synonyms provided they represent the same concept in all or most contexts.

3 = **Near synonym** – a term that represents the same or a similar concept as the original Fully Specified Name in the American English Edition, depending of context of use. The interchangeability is limited. Implementers should decide whether to consider the referenced term as a valid concept designation in the context on which the users would interact with the application. It is included as a relative synonym since users frequently use it to reference the concept.

4 = **Non synonymous** – a term that cannot be considered a synonym of the concept designated by the Fully Specified Name in the American English Edition, but which is included to support implementation specific scenarios. The Scope attribute of the corresponding member row should be other than the default scope, since this term should only be displayed when referenced by a scoped navigation RefSet. The term should not be included in retrieval scenarios that are not restricted to the scope specified. The referenced descriptions are included as a particular case of near synonyms when the phrase is commonly used to reference the associated concept. While the release of this kind of description is discouraged, the enumerated value is provided for its potential use in extensions that require the inclusion of colloquial terms, jargon, and clipped forms.

5 = **Invalid** – not a valid designation for the concept; it may have previously been published as a valid designation. Descriptions that are retired at the Language level (Descriptions table) would become invalid in all Language RefSets referencing that description.

7.4 Proposed changes to the Descriptions table

The inclusion of the Acceptability, Correctness and DegreeOfSynonymy attributes in the RefSetMembers table removes dialect-dependent information from the Descriptions table, improving support for international editions. This proposal also includes changes to the enumerated values allowed in three fields, to avoid overloading in their meaning.

- For more details see *Descriptions_SV*

Column	Description
id	The unique *SNOMED CT* Identifier (*SCTID*) assigned to this *Description*.
effectiveTime	The date and time from which this representation of the identified *Description* applied.
status	The *status* of the *Description* from the specified *effectiveTime*.
conceptId	The unique *SNOMED CT* Identifier of the associated *Concept*.
term	The text of a *term* used to describe the associated *Concept*.
initialCapitalStatus	Indicates whether the capitalization status of the first character of the *term* is significant.
descriptionType	Specifies the nature of the *term* in relation to the associated *Concept*.
languageCode	Specifies the language in which the *term* is valid.
descriptionForm	An indication of whether the *term* is a plural, abbreviated or symbolic form.

7.4.1 Additional attribute: descriptionForm

An indication of whether the *term* is a plural, abbreviated or symbolic form.

History

This is a new attribute that enables enhanced language support. Previously, there was no way to indicate the presence of plurals, abbreviations or symbolic forms. This additional information allows recognition of:

- abbreviated forms (e.g. "LVF")
- plural forms (e.g. "investigations" as a *term* applied to the *Concept* "evaluation procedure")
- symbolic forms (e.g. UCUM representation of units).

7.4.2 Changes to existing attribute: status

The *status* of the *Description* from the specified *effectiveTime*.

Notes

- The *status* values applicable to *Descriptions* include only three values. Current (active), Retired (inactive), and Referred to another concept (inappropriate).
- The *status* of a *Description* does not change when a *Concept* is made inactive. Thus in the *Descriptions_SV* an inactive *Concept* will have active *Descriptions*.

History

- Previously called: **DescriptionStatus**

In the original release format some values for the *status* of a *Description* were entirely dependent on the *status* of the *Concept* (i.e. active *Descriptions* of *Concepts* with Limited *status* also had "Limited" *status* and active *Descriptions* of Inactive *Concepts* had the *status* "Inactive

Concept"). This redundant representation of the *status* of the *Concept* is no longer supported in the *Descriptions_SV* file. However, it can still be derived from the *status* of the associated *Concept*.

These values are not used in the proposed new release format because:

- they do not add information
- if they were included, additional redundant rows would be added to the *Descriptions_SV* table every for every *Description* of each *Concept* that changed its *status*.

The reason for including these *DescriptionStatus* values in the original release format was to optimize filtering when searching the *Descriptions* table. Like many other potentially useful optimizations this denormalized form can be created by applying a simple SQL query.

7.4.3 Changes to existing attribute: languageCode

Specifies the language in which the *term* is valid.

Notes

- This attribute represents the primary language in which this *Description* is managed.
- Dialect tags are not included in the *languageCode*.
    - All *terms* that are valid in any dialect of a language have the same *languageCode* in the *Descriptions_SV* file;
    - Dialect support is dependent on use of the relevant *Language RefSet*. *
- If a lexically identical *term* is coincidently the same in two languages, these are supported by separate *Descriptions* with their own unique identifiers.
    - This minimizes the risk of errors introduced by changes made in one language that are inappropriate in another language. *
- International *terms*, such as those derived from classical Greek and Latin may be supported with a different *languageCode* and referenced by different *Language RefSets* for several languages.

History

- Previously called: **LanguageCode**

In the past, the *languageCode* included the dialect tag and the same *Description* was released with a different *languageCode* in different *SNOMED CT Editions*. This approach duplicated the information in the *Language Subset*. The revised distribution format does not include this redundant information because it would undermine:

- The component history tracking approach by allowing the same *Description* to have two different *languageCode* values at the same time.
- The enhanced support for multi-lingual development and maintenance provided by *Language RefSets*.

It is possible to generate a dialect specific *Description* snapshot that matches the current distribution format by applying a simple SQL query to the new distribution files. This may be useful for implementation optimization but is not an authoritative representation of the *SNOMED CT Description*.

7.4.4 Changes to existing attribute: descriptionType

Specifies the nature of the *term* in relation to the associated *Concept*. Indicates whether the term is a Designation, a FullySpecifiedName, a Definition or an external reference.

Notes

- The *descriptionType* attribute only distinguishes between *Descriptions* that are of a fundamentally different type. It is not used to specify preference or acceptability of a *term* in a particular language.
- Dialect or other preferences are specified by the relevant *Language RefSet*.

History

- Previously called: **DescriptionType**

Dialect and usage preference, including allocation of a *Preferred Term* is no longer represented in the *Descriptions Table*. The *Language RefSet* provides a more expressive mechanism for denoting preferred, acceptable and deprecated *terms* in a particular dialect, locality or specialty.

In the past, the *descriptionType* was used to distinguish between *Preferred Terms*, *Synonyms* and *terms* not used in a given dialect. As a result the same *Description* was released with a different *descriptionType* in different *SNOMED CT Editions*. This approach duplicated information in the *Language Subset*. The revised distribution format does not include this redundant information because it would undermine:

- The component history tracking approach by allowing the same *Description* to have two different *descriptionType* values at the same time.
- The enhanced support for multi-lingual development and maintenance provided by *Language RefSets*.

It is possible to generate a dialect specific *Description* snapshot that matches the current distribution format by applying a simple SQL query to the new distribution files. This may be useful for implementation optimization but is not an authoritative representation of the *SNOMED CT Description*.

8 Navigation Reference Sets

8.1 Overview

There is a requirement to be able to organize concepts into hierarchies suitable for navigation. This requirement arises because the subtype hierarchy is often not intuitive for navigation due to its depth, size and lack of a definable order. The introduction of support for negation and context will make some subtype hierarchies more accurate but even less intuitive to a human reader.

8.2 Requirements

An enhanced version of this requirement includes the ability to determine the navigational children of a concept, based on the manner in which that concept was reached. For example, if diabetes mellitus is found by navigation down a hierarchy of disorders then one set of refinements might be presented as its navigational children. However, if the same concept is reached as a cause of cataract then a much more limited selection of options might be available.

8.3 Navigation Reference set member table structure – summary

Field Name	Type	Notes	
Id	SCTID	Unique SCT identifier for the RefSet member row. It uses the partitionId 7, that identifies RefSet members.	M
EffectiveTime	String[14]	Universal Time (UTC) at which the component representation becomes effective. The component representation remains valid until it is superseded. The date and time should be represented using ISO 8601 conventions with no punctuation and no letter T between date and time, with a granularity up to the level of seconds.	M
Status	Enum	0=Current, 1=Non Current	M
RefSetId	SCTID	Refset to which the component reference belongs	M
ComponentId	SCTID	Unique SCT identifier of the referenced component	M
Scope	String[20]	Context in which this member row applies	C
Priority	Integer	Relative priority of the member in the set	M
TagCode	String[20]	Coded attribute to associate with this member	R
Annotation	String[255]	Text to associate with this version of the RefSet	R
ChildScope	String[20]	Target scope to be considered to select the Child	C
ChildId	SCTID	SCTID of the referenced child component	
DefaultToSubtype	Boolean	Indicates whether drill-down navigation should continue using the subtype hierarchy if the concept referenced by the ChildId has no explicit navigational children in the subset.	C
DisplayOrder	Integer	Indicates the default order for member rows	R

Notes

The root members of the navigational hierarchy can be identified by comparing the fields ComponentId and ChildId for the default scope. If these fields match, then it is a root member.

8.4 Navigation Reference Set example

- Works by direct reference to a component.
- Scope added to modulate the children referenced by a parent according to situation.
- ChildScope added to modulate the components referenced by after reaching a Child.
- This approach mean simple cases (i.e. with same value for Scope and ChildScope) work exactly like a simple navigations Refset (or current subset).

C1: Diagnoses
- C2: Asthma
  - C5: Investigation
    - C7: Peak flow
    - C8: Chest X-ray
  - C6: Treatment
    - C9: Salbutamol
    - C10: Beclomethasone
- C3: Pneumonia
  - C5: Investigation
    - C8: Chest X-ray
    - C11: Sputum Culture
  - C6: Treatment
    - C12: Amoxil
    - C13: Clindamicin
- C4: Appendicitis
  - C6: Treatment
    - C14: Appendectomy
    - C15: Antibiotics
      - C16 Amoxyclav
      - C12: Amoxil
      - C17: Metronidazole

Note the parents and children of concepts change with scope A, B, C, D so that C6: Investigation has different children in each of the three scopes and C8: Chest X-reay and C12: Amoxil have different parents according to scope.

See below the table that delivers the scope based navigation hierarchy on previous page.

Scoped Navigation example (partial view, simplified)

ComponentId	Scope	ChildScope	ChildId	DefaultToSubtype	DisplayOrder
C1	A	A	C2	0	1
C1	A	A	C3	0	2
C1	A	A	C4	0	3
C2	A	B	C5	0	1
C2	A	B	C6	0	2
C3	A	C	C5	0	1
C3	A	C	C6	0	2
C4	A	D	C6	0	1
C5	B	B	C7	0	1
C5	B	B	C8	0	2
C5	B	B	C8	0	2
C6	B	B	C9	0	1
C6	B	B	C10	0	2
C5	C	C	C8	0	1
C5	C	C	C11	0	2
C6	C	C	C12	0	1
C6	C	C	C13	0	2
C6	D	D	C14	0	1
C6	D	D	C15	0	2
C15	D	D	C16	0	1
C15	D	D	C12	0	2
C15	D	D	C17	0	3

9 Aggregation Reference Sets

9.1 Overview

Aggregation Reference Sets would allow the distribution of alternative aggregation hierarchies for SNOMED Concepts. The preliminary table structure is included to stimulate discussion at this stage of the Draft document while ongoing refinement work is underway.

10 Appendix: Reference Set file structure details

Note: The content in this chapter was originally produced as part of the RefSet and History Mechanism Technology Preview documentation under development at the time this draft version is published. It is reproduced here to ensure consistency, although some features like hyperlinks are limited or absent. Table numbers were intentionally removed to avoid conflicts with the data dictionary that would eventually unify documentation related to SNOMED CT data structures. This content would then be replaced by a reference to the appropriate documents once they become available.

Table *RefSets_SV* file details

Table: RefSets_SV		Partition Identifier
Component: RefSet		Core: **03**
		Extension: **13**
Each row in this table represents the view of the metadata associated with a *SNOMED CT RefSet* in the state in which it was released at a specified time. A *SNOMED CT RefSet* is a set of references to other *SNOMED CT Components*. Notes • Several different types of *RefSet* are supported to meet a range of different requirements including language selection, content filtering, content prioritization and alternative hierarchies for navigation and aggregation. • Each reference within the set is represented by a *RefSetMember* which refers to the component and may also include additional data specific to the type of *RefSet* and its intended purpose.		
Column	**Data Type**	**Value Set**
id	*sctid*	
The unique *SNOMED CT* Identifier (*SCTID*) assigned to this *RefSet*. Note • Multiple views of the same *RefSet* may exist in the *RefSets_SV Table* and these will share the same *id*. The unique key for these "state valid" (SV) tables is the *id* + *effectiveTime*.		
effectiveTime	*timestamp (14)*	
The date and time from which this representation of the identified *RefSet* applied.		
status	*int (8) (Enum)*	*StatusSimple*
The *status* of the *RefSet* from the specified *effectiveTime*.		
name	*nvarchar (255)*	
A name that describes the purpose or usage of this *RefSet*.		
shortName	*varchar (20)*	
An abbreviated name of the *RefSet*. This is useful for display in menu options of other places where the *RefSet.name* may be too long.		
refSetType	*int (8) (Enum)*	*RefSetType*
Indicates the nature of the *RefSet*.		
isExclusion	*boolean*	*IsExclusion*
Indicates where the members of the *RefSet* are intended to be excluded (rather than included) in searches.		

isScoped	*boolean*	*IsScoped*
Indicates whether any members of the *RefSet* have the *scope* attribute. *Note* • A scoped *RefSet* allows the membership to be varied dependent on the *scope* in which it is used.		
isPrioritized	*boolean*	*IsPrioritized*
Indicates whether any members of the *RefSet* have the *priority* attribute. *Note* • A prioritized *RefSet* allows members to be given different priority levels within the set.		
memberType	*int (8) (Enum)*	*ComponentType*
Indicates the type of *SNOMED CT Component* that is referenced by members of this *RefSet*.		
realmId	*sctid*	
Country, or other jurisdiction, within which this *RefSet* applies. *Note* • A blank or null value implies that there is no specified realm restriction.		
domainId	*sctid*	
Discipline, specialty or business area for which this *RefSet* applies. *Note* • A blank or null value implies that there is no specified domain restriction.		
languageCode	*lang (12)*	
Language and, optionally dialect, to which this *RefSet* applies.		
annotation	*nvarchar (255)*	
A text annotation that is associated with this *RefSet* and applies to all its members.		
originator	*sctid*	
Namespace concept for the creator of this version of the *RefSet*.		
baseRefSetId	*sctid*	
Refers to the *RefSet.id* of another *RefSet* on which this *RefSet* is based. *Note* • The explicitly specified members of this *RefSet* add to or override members of the *Base RefSet*.		
baseRefSetEffectiveTime	*timestamp (14)*	
Effective time of the *Base RefSet* membership on which the *RefSet* is based. *Note* • See also *baseRefSetLatestVersion*.		
baseRefSetLatestVersion	*boolean*	*BaseRefSetLatestVersion*
Indicated whether the membership of the *Base RefSet* should be based on the latest available version of that *RefSet* rather than a specified *baseRefSetEffectiveTime*.		

Table *RefSetMembers_SV* file details

Table: RefSetMembers_SV		Partition Identifier
Component: RefSetMember		Core: **07**
		Extension: **17**
Each row in this table represents the view of a *SNOMED CT RefSetMember* in the state in which it was released at a specified time. A *SNOMED CT RefSetMember* is a reference to a *SNOMED CT Component* that indicates its *status* within a specified *RefSet*. *Note* • A *RefSetMember* may also include additional data specific to the type of *RefSet* and its intended purpose. For example in the case of a Language *RefSet* it indicates whether a referenced description is the *Preferred Term*.		
Column	**Data Type**	**Value Set**
id	*sctid*	
The unique *SNOMED CT* Identifier (*SCTID*) assigned to this *RefSetMember*. *Note* • Multiple views of the same *RefSetMember* may exist in the *RefSetMembers_SV Table* and these will share the same *id*. The unique key for these "state valid" (SV) tables is the *id* + *effectiveTime*.		
effectiveTime	*timestamp (14)*	
The date and time from which this representation of the identified *RefSetMember* applied.		
status	*int (8) (Enum)*	*StatusSimple*
The *status* of the *RefSetMember* from the specified *effectiveTime*.		
refSetId	*sctid*	
The unique identifier of the *RefSet* to which this member belongs.		
componentId	*sctid*	
The unique identifier of the main *SNOMED CT Component* referenced by this member.		
tagCode	*varchar (20)*	
A coded attribute associated with this member. *Note* • The *tagCode* may be used to attach codes that flag or categorize components that are members of the *RefSet*.		
annotation	*nvarchar (255)*	
A free-text string annotation associated with this member. *Note* • The *annotation* may be used to attach explanations or other information to components that are members of the *RefSet*.		
scope	*varchar (20)*	
A string identifying the *scope* within which this member applies. *Notes* • This property is only used in Scoped RefSets. The membership of these sets varies according to the *scope* in which they are used. • The *scope* field defines the context in which this member is a part of a scoped *RefSet*. • Outside the specified *scope* a *RefSetMember* is ignored. However, a scoped *RefSet* may contain several		

RefSetMembers that reference the same component in different scopes.

- There are no rules about the identifiers other than that they fit within the specified datatype.

Example

- A Scoped *RefSet* might contain members applicable to different fields in a data entry form, protocol or communication specification. Each of these scopes is assigned an identifier

priority	*int (16)*	

The relative *priority* of the member in the set.

Note

- The highest *priority* is represented by the *priority* =1.

acceptability	*int (8) (Enum)*	*Acceptability*

Indicates whether the *term* of the referenced *Description* is preferred or accepted by users as a description of the associated *Concept*.

The *Preferred Term* for a *Concept* value is indicated by the *acceptability* value "Preferred". Recognized synonyms are inidicated by the *acceptability* value "Acceptable".

Note

- Preferences depend on the usage of the language rather than on the language or dialect itself. Thus acceptability may vary both between dialects and within a dialect based on the conventions of a particular discipline, specialty or organization.

correctness	*int (8) (Enum)*	*Correctness*

Indicates whether the *term* of the referenced *Description* is a linguistically correct description of the associated *Concept* according to the official Naming Authority for the language, dialect or realm.

Note

- If no Authority exists for the language, dialect or realm, the *correctness* value "Unspecified" is used.

degreeOfSynonymy	*int (8) (Enum)*	*DegreeOfSynonymy*

Indicates the degree to which the *term* of the referenced *Description* covers the same concept as the *fullySpecifiedName* attribute of the associated *Concept* (i.e. the primary, original *Fully Specified Name*).

Notes

- There is a tension between the need to accommodate expressions common in natural language and a more restrictive policy to only include actual synonyms.
- In natural language, the context in which a phrase is used might refine its meaning and make it seem synonymous with most specific phrase. However, the phrase on its own may have a much less specific meaning.

Example

- The *term* "fundus" is often used to mean "fundus of the eye" or "fundus of the uterus". However it is not a complete synonym for either. Its assumed synonymy is dependent on the context in which it is used.

childScope	*varchar (20)*	

The scope within which the *Component* referred to by the *childId* attribute is to be interpreted.

Note

- Within a *Navigation Hierarchy* the hierarchical parents and hierarchical children of a *Component* may differ according to the scope that it is interpreted in.

Examples

- In a hierarchy the children of a *Concept* called "investigation" could differ according to whether it was reached by navigation from a particular disease.
- If a particular procedure is reaached by navigation from a particular disorder, that disorder should appear as a navigational parent. However, this disorder might not appear as a navigational parent if the same

procedure was reached by a different navigational route.		
childId	*sctid*	
Refers to a *Component* that is intended to behave as a hierarchical child of the *Component* referred to by the *componentId*.		
defaultToSubtype	*boolean*	*DefaultToSubtype*
Indicates whether the logical subtypes of the *Component* referred to by the *componentId* attribute should be treated as Navigational children of that *Component*.		
displayOrder	*int (16)*	
Determines the order in which the child *Component* should be should be displayed when showing the set of children of the *Component* referenced the the *componentId* attribute. The items with the lowest *displayOrder* are shown first. *Notes* • *displayOrder* values need not be unique. Items with identical order numbers may be ordered alphabetically or in an arbitrary manner). • *displayOrder* values need not be consecutive. Gaps in the order should be ignored. o This means there is no need to change the *displayOrder* of all children in the list simply because one child has been removed. o Deliberately leaving gaps in the *displayOrder* sequence also allows new children to be added without renumbering other children.		
defaultToSupertype	*boolean*	*DefaultToSupertype*
Indicates whether the *Concept* referred to by the *child* attribute should be aggregated under its logical supertypes if it has no specified aggregation parents.		
validFrom	*timestamp (14)*	
Specifies a date and time from which this aggregation is valid. *Notes* • Unlike the *effectiveTime*, the *validFrom* value does not influence the *status* of the *RefSetMember*.		
validTo	*timestamp (14)*	
Specifies a date and time after which this aggregation is no longer valid. *Notes* • Unlike the *effectiveTime*, the *validTo* value does not influence the *status* of the *RefSetMember*.		

Table Simple *RefSetMembers_SV* file details

Table: Simple RefSetMembers_SV		Partition Identifier
Component: RefSetMember		Core: **07**
		Extension: **17**
Each row in this table represents the view of a *SNOMED CT RefSetMember* in the state in which it was released at a specified time. A *SNOMED CT RefSetMember* is a reference to a *SNOMED CT Component* that indicates its *status* within a specified *RefSet*. *Note* • A *RefSetMember* may also include additional data specific to the type of *RefSet* and its intended purpose. For example in the case of a Language *RefSet* it indicates whether a referenced description is the *Preferred Term*.		
Column	**Data Type**	**Value Set**
id	*sctid*	
The unique *SNOMED CT* Identifier (*SCTID*) assigned to this *RefSetMember*. *Note* • Multiple views of the same *RefSetMember* may exist in the *Simple RefSetMembers_SV Table* and these will share the same *id*. The unique key for these "state valid" (SV) tables is the *id* + *effectiveTime*.		
effectiveTime	*timestamp (14)*	
The date and time from which this representation of the identified *RefSetMember* applied.		
status	*int (8) (Enum)*	*StatusSimple*
The *status* of the *RefSetMember* from the specified *effectiveTime*.		
refSetId	*sctid*	
The unique identifier of the *RefSet* to which this member belongs.		
componentId	*sctid*	
The unique identifier of the main *SNOMED CT Component* referenced by this member.		
tagCode	*varchar (20)*	
A coded attribute associated with this member. *Note* • The *tagCode* may be used to attach codes that flag or categorize components that are members of the *RefSet*.		
annotation	*nvarchar (255)*	
A free-text string annotation associated with this member. *Note* • The *annotation* may be used to attach explanations or other information to components that are members of the *RefSet*.		
scope	*varchar (20)*	
A string identifying the *scope* within which this member applies. *Notes* • This property is only used in Scoped RefSets. The membership of these sets varies according to the *scope* in which they are used. • The *scope* field defines the context in which this member is a part of a scoped *RefSet*. • Outside the specified *scope* a *RefSetMember* is ignored. However, a scoped *RefSet* may contain several		

RefSetMembers that reference the same component in different scopes. • There are no rules about the identifiers other than that they fit within the specified datatype. *Example* • A Scoped *RefSet* might contain members applicable to different fields in a data entry form, protocol or communication specification. Each of these scopes is assigned an identifier		
priority	*int (16)*	
The relative *priority* of the member in the set. *Note* • The highest *priority* is represented by the *priority* =1.		

Table Language *RefSetMembers_SV* file details

Table: Language RefSetMembers_SV		Partition Identifier
Component: RefSetMember		Core: **07**
		Extension: **17**
Each row in this table represents the view of a *SNOMED CT RefSetMember* in the state in which it was released at a specified time. A *SNOMED CT RefSetMember* is a reference to a *SNOMED CT Component* that indicates its *status* within a specified *RefSet*. *Note* • A *RefSetMember* may also include additional data specific to the type of *RefSet* and its intended purpose. For example in the case of a Language *RefSet* it indicates whether a referenced description is the *Preferred Term*.		
Column	**Data Type**	**Value Set**
id	*sctid*	
The unique *SNOMED CT* Identifier (*SCTID*) assigned to this *RefSetMember*. *Note* • Multiple views of the same *RefSetMember* may exist in the *Language RefSetMembers_SV Table* and these will share the same *id*. The unique key for these "state valid" (SV) tables is the *id + effectiveTime*.		
effectiveTime	*timestamp (14)*	
The date and time from which this representation of the identified *RefSetMember* applied.		
status	*int (8) (Enum)*	*StatusSimple*
The *status* of the *RefSetMember* from the specified *effectiveTime*.		
refSetId	*sctid*	
The unique identifier of the *RefSet* to which this member belongs.		
componentId	*sctid*	
The unique identifier of the main *SNOMED CT Component* referenced by this member.		
tagCode	*varchar (20)*	
A coded attribute associated with this member. *Note* • The *tagCode* may be used to attach codes that flag or categorize components that are members of the *RefSet*.		
annotation	*nvarchar (255)*	
A free-text string annotation associated with this member. *Note* • The *annotation* may be used to attach explanations or other information to components that are members of the *RefSet*.		
scope	*varchar (20)*	
A string identifying the *scope* within which this member applies. *Notes* • This property is only used in Scoped RefSets. The membership of these sets varies according to the *scope* in which they are used. • The *scope* field defines the context in which this member is a part of a scoped *RefSet*. • Outside the specified *scope* a *RefSetMember* is ignored. However, a scoped *RefSet* may contain several		

RefSetMembers that reference the same component in different scopes. • There are no rules about the identifiers other than that they fit within the specified datatype. *Example* • A Scoped *RefSet* might contain members applicable to different fields in a data entry form, protocol or communication specification. Each of these scopes is assigned an identifier		
acceptability	*int (8) (Enum)*	*Acceptability*
Indicates whether the *term* of the referenced *Description* is preferred or accepted by users as a description of the associated *Concept*. The *Preferred Term* for a *Concept* value is indicated by the *acceptability* value "Preferred". Recognized synonyms are inidicated by the *acceptability* value "Acceptable". *Note* • Preferences depend on the usage of the language rather than on the language or dialect itself. Thus acceptability may vary both between dialects and within a dialect based on the conventions of a particular discipline, specialty or organization.		
correctness	*int (8) (Enum)*	*Correctness*
Indicates whether the *term* of the referenced *Description* is a linguistically correct description of the associated *Concept* according to the official Naming Authority for the language, dialect or realm. *Note* • If no Authority exists for the language, dialect or realm, the *correctness* value "Unspecified" is used.		
degreeOfSynonymy	*int (8) (Enum)*	*DegreeOfSynonymy*
Indicates the degree to which the *term* of the referenced *Description* covers the same concept as the *fullySpecifiedName* attribute of the associated *Concept* (i.e. the primary, original *Fully Specified Name*). *Notes* • There is a tension between the need to accommodate expressions common in natural language and a more restrictive policy to only include actual synonyms. • In natural language, the context in which a phrase is used might refine its meaning and make it seem synonymous with most specific phrase. However, the phrase on its own may have a much less specific meaning. *Example* • The *term* "fundus" is often used to mean "fundus of the eye" or "fundus of the uterus". However it is not a complete synonym for either. Its assumed synonymy is dependent on the context in which it is used.		

Table *Navigation RefSet*Members_SV file details

Table: Navigation RefSetMembers_SV	Partition Identifier
Component: RefSetMember	Core: **07**
	Extension: **17**
Each row in this table represents the view of a *SNOMED CT RefSetMember* in the state in which it was released at a specified time. A *SNOMED CT RefSetMember* is a reference to a *SNOMED CT Component* that indicates its *status* within a specified *RefSet*. *Note* • A *RefSetMember* may also include additional data specific to the type of *RefSet* and its intended purpose. For example in the case of a Language *RefSet* it indicates whether a referenced description is the *Preferred Term*.	
Column	**Data Type**	**Value Set**

id	*sctid*	
The unique *SNOMED CT* Identifier (*SCTID*) assigned to this *RefSetMember*. *Note* • Multiple views of the same *RefSetMember* may exist in the *Navigation RefSetMembers_SV Table* and these will share the same *id*. The unique key for these "state valid" (SV) tables is the *id + effectiveTime*.		
effectiveTime	*timestamp (14)*	
The date and time from which this representation of the identified *RefSetMember* applied.		
status	*int (8) (Enum)*	*StatusSimple*
The *status* of the *RefSetMember* from the specified *effectiveTime*.		
refSetId	*sctid*	
The unique identifier of the *RefSet* to which this member belongs.		
componentId	*sctid*	
The unique identifier of the main *SNOMED CT Component* referenced by this member.		
tagCode	*varchar (20)*	
A coded attribute associated with this member. *Note* • The *tagCode* may be used to attach codes that flag or categorize components that are members of the *RefSet*.		
annotation	*nvarchar (255)*	
A free-text string annotation associated with this member. *Note* • The *annotation* may be used to attach explanations or other information to components that are members of the *RefSet*.		
scope	*varchar (20)*	
A string identifying the *scope* within which this member applies. *Notes* • This property is only used in Scoped RefSets. The membership of these sets varies according to the *scope* in which they are used. • The *scope* field defines the context in which this member is a part of a scoped *RefSet*. • Outside the specified *scope* a *RefSetMember* is ignored. However, a scoped *RefSet* may contain several RefSetMembers that reference the same component in different scopes. • There are no rules about the identifiers other than that they fit within the specified datatype. *Example* • A Scoped *RefSet* might contain members applicable to different fields in a data entry form, protocol or communication specification. Each of these scopes is assigned an identifier		
childScope	*varchar (20)*	
The scope within which the *Component* referred to by the *childId* attribute is to be interpreted. *Note* • Within a *Navigation Hierarchy* the hierarchical parents and hierarchical children of a *Component* may differ according to the scope that it is interpreted in. *Examples* • In a hierarchy the children of a *Concept* called "investigation" could differ according to whether it was reached by navigation from a particular disease.		

• If a particular procedure is reaached by navigation from a particular disorder, that disorder should appear as a navigational parent. However, this disorder might not appear as a navigational parent if the same procedure was reached by a different navigational route.		
childId	*sctid*	
Refers to a *Component* that is intended to behave as a hierarchical child of the *Component* referred to by the *componentId*.		
defaultToSubtype	*boolean*	*DefaultToSubtype*
Indicates whether the logical subtypes of the *Component* referred to by the *componentId* attribute should be treated as Navigational children of that *Component*.		
displayOrder	*int (16)*	
Determines the order in which the child *Component* should be should be displayed when showing the set of children of the *Component* referenced the the *componentId* attribute. The items with the lowest *displayOrder* are shown first. *Notes* • *displayOrder* values need not be unique. Items with identical order numbers may be ordered alphabetically or in an arbitrary manner). • *displayOrder* values need not be consecutive. Gaps in the order should be ignored. o This means there is no need to change the *displayOrder* of all children in the list simply because one child has been removed. o Deliberately leaving gaps in the *displayOrder* sequence also allows new children to be added without renumbering other children.		

Table *Aggregation RefSet*Members_SV file details

Table: Aggregation RefSetMembers_SV		Partition Identifier
Component: RefSetMember		Core: **07**
		Extension: **17**
Each row in this table represents the view of a *SNOMED CT RefSetMember* in the state in which it was released at a specified time. A *SNOMED CT RefSetMember* is a reference to a *SNOMED CT Component* that indicates its *status* within a specified *RefSet*. *Note* • A *RefSetMember* may also include additional data specific to the type of *RefSet* and its intended purpose. For example in the case of a Language *RefSet* it indicates whether a referenced description is the *Preferred Term*.		
Column	**Data Type**	**Value Set**
id	*sctid*	
The unique *SNOMED CT* Identifier (*SCTID*) assigned to this *RefSetMember*. *Note* • Multiple views of the same *RefSetMember* may exist in the *Aggregation RefSetMembers_SV Table* and these will share the same *id*. The unique key for these "state valid" (SV) tables is the *id + effectiveTime*.		
effectiveTime	*timestamp (14)*	
The date and time from which this representation of the identified *RefSetMember* applied.		
status	*int (8) (Enum)*	*StatusSimple*
The *status* of the *RefSetMember* from the specified *effectiveTime*.		
refSetId	*sctid*	
The unique identifier of the *RefSet* to which this member belongs.		
componentId	*sctid*	
The unique identifier of the main *SNOMED CT Component* referenced by this member.		
tagCode	*varchar (20)*	
A coded attribute associated with this member. *Note* • The *tagCode* may be used to attach codes that flag or categorize components that are members of the *RefSet*.		
annotation	*nvarchar (255)*	
A free-text string annotation associated with this member. *Note* • The *annotation* may be used to attach explanations or other information to components that are members of the *RefSet*.		
scope	*varchar (20)*	
A string identifying the *scope* within which this member applies. *Notes* • This property is only used in Scoped RefSets. The membership of these sets varies according to the *scope* in which they are used. • The *scope* field defines the context in which this member is a part of a scoped *RefSet*. • Outside the specified *scope* a *RefSetMember* is ignored. However, a scoped *RefSet* may contain several		

RefSetMembers that reference the same component in different scopes. • There are no rules about the identifiers other than that they fit within the specified datatype. *Example* • A Scoped *RefSet* might contain members applicable to different fields in a data entry form, protocol or communication specification. Each of these scopes is assigned an identifier		
priority	*int (16)*	
The relative *priority* of the member in the set. *Note* • The highest *priority* is represented by the *priority* =1.		
childId	*sctid*	
Refers to a *Component* that is intended to behave as a hierarchical child of the *Component* referred to by the *componentId*.		
defaultToSupertype	*boolean*	*DefaultToSupertype*
Indicates whether the *Concept* referred to by the *child* attribute should be aggregated under its logical supertypes if it has no specified aggregation parents.		
validFrom	*timestamp (14)*	
Specifies a date and time from which this aggregation is valid. *Notes* • Unlike the *effectiveTime*, the *validFrom* value does not influence the *status* of the *RefSetMember*.		
validTo	*timestamp (14)*	
Specifies a date and time after which this aggregation is no longer valid. *Notes* • Unlike the *effectiveTime*, the *validTo* value does not influence the *status* of the *RefSetMember*.		

Table *Descriptions_SV* file details

Table: Descriptions_SV		Partition Identifier
Component: Description		Core: **01**
		Extension: **11**
Each row in this table represents the view of a *SNOMED CT Description* in the state in which it was released at a specified time. A *SNOMED CT Description* associates a *term* with a *SNOMED CT Concept*, which it can be used to represent. *Note* • Each *Description* has its own unique identifier (descriptionId) and also contains the text of a *term* and the identifier of the *Concept* it may represent. Where the same *term* can be used to express several different *Concepts*, each distinct use of the *term* is represented by a separate uniquely identified *Description*.		
Column	**Data Type**	**Value Set**
id	*sctid*	
The unique *SNOMED CT* Identifier (*SCTID*) assigned to this *Description*. *Note* • Multiple views of the same *Description* may exist in the *Descriptions_SV Table* and these will share the same *id*. The unique key for these "state valid" (SV) tables is the *id* + *effectiveTime*.		
effectiveTime	*timestamp (14)*	
The date and time from which this representation of the identified *Description* applied.		
status	*int (8) (Enum)*	*StatusDescription*
The *status* of the *Description* from the specified *effectiveTime*. *Notes* • The *status* values applicable to *Descriptions* include only three values. Current (active), Retired (inactive), and Referred to another concept (inappropriate). • The *status* of a *Description* does not change when a *Concept* is made inactive. Thus in the *Descriptions_SV* an inactive *Concept* will have active *Descriptions*.		
conceptId	*sctid*	
The unique *SNOMED CT* Identifier of the associated *Concept*. *Note* • This field provides the association between the *Descriptions* and *Concepts* tables.		
term	*nvarchar (255)*	
The text of a *term* used to describe the associated *Concept*. *Notes* • There may be many *terms* associated with a single *Concept*. In each language or dialect there must be one *term* that fulfills the role of a *Fully Specified Name* for the *Concept* and there may be many other synonymous or partially synonymous *terms*. • In any given language or dialect the *Preferred Term* and the *acceptability* of other *terms* are specified by a Language *RefSet*. *Language RefSets* can be applied or refined by national, local or specialty groups to meet specific requirements. • The limitation in the length of this field is 255 bytes. All ASCII characters in the range 32 to 127 can be encoded in a single byte using UTF-8. However, accented and special characters require two or three byte encoding. Therefore, the maximum length for a *term* that contains these characters is less than 255 characters. • The *FullySpecifiedName* is also present in the *Concepts Table*. It is included in the *Descriptions Table* to		

<table>
<tr><td colspan="3">allow language and dialect variations to be represented</td></tr>
<tr><td>initialCapitalStatus</td><td>boolean</td><td>InitialCapitalStatus</td></tr>
<tr><td colspan="3">Indicates whether the capitalization status of the first character of the term is significant.
<u>Note</u>
• Capitalization of characters other than the first character in the term is always regarded as significant.</td></tr>
<tr><td>descriptionType</td><td>int (8) (Enum)</td><td>DescriptionType</td></tr>
<tr><td colspan="3">Specifies the nature of the term in relation to the associated Concept.
<u>Notes</u>
• The descriptionType attribute only distinguishes between Descriptions that are of a fundamentally different type. It is not used to specify preference or acceptability of a term in a particular language.
• Dialect or other preferences are specified by the relevant Language RefSet.</td></tr>
<tr><td>languageCode</td><td>lang (4)</td><td></td></tr>
<tr><td colspan="3">Specifies the language in which the term is valid.
<u>Notes</u>
• This attribute represents the primary language in which this Description is managed.
• Dialect tags are not included in the languageCode.
o All terms that are valid in any dialect of a language have the same languageCode in the Descriptions_SV file;
o Dialect support is dependent on use of the relevant Language RefSet. *
• If a lexically identical term is coincidently the same in two languages, these are supported by separate Descriptions with their own unique identifiers.
o This minimizes the risk of errors introduced by changes made in one language that are inappropriate in another language. *
• International terms, such as those derived from classical Greek and Latin may be supported with a different languageCode and referenced by different Language RefSets for several languages.</td></tr>
<tr><td>descriptionForm</td><td>int (8) (Enum)</td><td>DescriptionForm</td></tr>
<tr><td colspan="3">An indication of whether the term is a plural, abbreviated or symbolic form.</td></tr>
</table>

11 Appendix: Datatypes used in the Distribution Files

Note: The content in this chapter was originally produced as part of the RefSet and History Mechanism Technology Preview documentation under development at the time this draft version is published. It is reproduced here to ensure consistency, although some features like hyperlinks are limited or absent. Table numbers were intentionally removed to avoid conflicts with the data dictionary that would eventually unify documentation related to SNOMED CT data structures. This content would then be replaced by a reference to the appropriate documents once they become available.

.

Table Datatype details

Datatype	Description
boolean	A Boolean value representing either *true* or *false*. In the distribution files a boolean value is represented as a single digit string: • **0** = *False* • **1** = *True*
char (1)	A single ASCII character string *Note* • Only ASCII values in range 32-127 are permitted this datatype can be read as simple ASCII or as UTF-8.
int (8) (Enum)	An eight bit unsigned integer representing an enumerated value. See associated value-set In the distribution files an integer value is represented as a string of decimal digits.
int (16)	A sixteen bit signed integer In the distribution files an integer value is represented as a string of decimal digits.
lang (4)	A language code represented as a string of up to 4 ASCII characters. The language code conforms to ISO639-1 and in this short form does not include a dialect sub-code.
lang (12)	A language code (optionally including dialect) represented as a string of up to 12 ASCII characters Consists of two parts separated by a dash. <li class="TableText">a language code conforming to ISO639-1 <li class="TableText">an optional dialect (or country) sub-code, confirming to the ISO3166 country code (or IANA code for countries not listed in ISO3166). If the languageCode value= 0, the RefSet applies to all languages – however, this is not a valid value for Language RefSets.
nvarchar (255)	A Unicode string of up to 255 characters represented in the distribution file using UTF-8 encoding.
oid	A string representing an ISO Object Identifier (OID)
sctid	*SNOMED CT* Identifier as an unsigned 64 bit integer or a string of up to 18 digits.
timestamp (14)	In the distribution files a timestamp is represented as a string of 8 or 14 decimal digits encoded as ASCII characters. The order of the digits is as follows

	YYYYMMDDhhmmss. This confirms with the separator free format specified in ISO 8601. *Notes* • All timestamps are based on Universal Time which is equivalent to the GMT time zone. This convention applies in all releases irrespective of local time zone or daylight saving time. • If time is omitted the assumed time is 00:00:00 UTC on the specified date. *Examples* • 20070131 o Means 00:00:00 UTC (midnight GMT) on 31st January 2007 • 20070131190000 o Means 17:00:00 UTC (5PM GMT) on 31st January 2007
varchar (5)	A string of up to 5 ASCII characters (ASCII values in range 32-127) *Note* • Only ASCII values in range 32-127 are permitted so these strings can be read either as simple ASCII or as UTF-8.
varchar (8)	A string of up to 8 ASCII characters (ASCII values in range 32-127) *Note* • Only ASCII values in range 32-127 are permitted so these strings can be read either as simple ASCII or as UTF-8.
varchar (12)	A string of up to 12 ASCII characters (ASCII values in range 32-127) *Note* • Only ASCII values in range 32-127 are permitted so these strings can be read either as simple ASCII or as UTF-8.
varchar (20)	A string of up to 20 ASCII characters *Note* • Only ASCII values in range 32-127 are permitted so these strings can be read either as simple ASCII or as UTF-8.
varchar (255)	A string of up to 255 ASCII characters (ASCII values in range 32-127) *Note* • Only ASCII values in range 32-127 are permitted so these strings can be read either as simple ASCII or as UTF-8.

12 Appendix: Value-Sets used in the Distribution Files

Note: The content in this chapter was originally produced as part of the RefSet and History Mechanism Technology Preview documentation under development at the time this draft version is published. It is reproduced here to ensure consistency, although some features like hyperlinks are limited or absent. Table numbers were intentionally removed to avoid conflicts with the data dictionary that would eventually unify documentation related to SNOMED CT data structures. This content would then be replaced by a reference to the appropriate documents once they become available.

Table *Acceptability* Value-Set details

Value-Set: Acceptability		
Acceptability of the *term* to users		
Value	**Name**	**Description**
0	Unspecified	The *acceptability* of the *term* is not specified in this *RefSet*. This is the default when a *Description* is not referenced by the *RefSet*.
1	Preferred	The referenced *Description* contains the *Preferred Term* for the associated *Concept* in the language and *scope* of this *RefSet*.
2	Acceptable	The referenced *Description* contains a *term* which, in the language and *scope* of this *RefSet*, is considered an acceptable representation of the associated *Concept*.
3	Not recommended	The referenced *Description* contains a *term* which, in the language and *scope* of this *RefSet*, is regarded as a deprecated representation of the associated *Concept*.
4	Not acceptable	The referenced *Description* contains a *term* which, in the language and *scope* of this *RefSet*, is not acceptable as a representation of the associated *Concept*.
5	Invalid	The referenced *Description* contains a *term* which, in the language and *scope* of this *RefSet*, not valid as a representation of the associated *Concept*.

Table BaseRefSetLatestVersion Value-Set details

Value-Set: BaseRefSetLatestVersion		
Value	**Name**	**Description**
0	Use specified version	The membership of this *RefSet* is specified as differences from the membership of the *RefSet* referenced by *baseRefSetId* as it was at the specified *baseRefSetEffectiveTime*.
1	Use latest version	The membership of this *RefSet* is specified as differences from the membership of the most recent version of the *RefSet* referenced by *baseRefSetId*.

Table *ComponentType* Value-Set details

Value-Set: ComponentType		
Type of *SNOMED CT Component*.		
Value	**Name**	**Description**
0	Concept	
1	Description	
2	Relationship	
3	RefSet	
4	CrossMapSet	
5	CrossMapTarget	
6	CrossMap	
7	RefSetMember	

Table *Correctness* Value-Set details

Value-Set: Correctness		
Formal correctness of the *term* in the language		
Value	**Name**	**Description**
0	Unspecified	The formal linguistic correctness of the *term* is not specified in this *RefSet*. This is the default when a *Description* is not referenced by the *RefSet*.
1	Recommended	The referenced *Description* contains a *term* is recommended as formally correct in the language and *scope* of this *RefSet*.
2	Acceptable	The referenced *Description* contains a *term* which, while not recommended, is regarded as linguistically acceptable in the language of this *RefSet*.
4	Incorrect	The referenced *Description* contains a *term* which is regarded as linguistically incorrect in the language of this *RefSet*.
5	Invalid	The referenced *Description* contains a *term* which is invalid in the language of this *RefSet*.

Table *DefaultToSubtype* Value-Set details

Value-Set: DefaultToSubtype		
Value	**Name**	**Description**
0	Do not default to subtype	The defined subtype *__Concept__(s)* are not to be treated as navigational children of the member *Concept*.
1	Default to subtype	The defined subtype *__Concept__(s)* are to be treated as navigational children of the member *Concept*.

Table *DefaultToSupertype* Value-Set details

Value-Set: DefaultToSupertype		
Value	**Name**	**Description**
0	Do not default to supertype	An aggregation parent *Concept* is specified and the defined supertype *__Concept__(s)* are not to be treated as aggregators for the member *Concept*.
1	Default to supertype	The defined supertype *__Concept__(s)* are to be treated as aggregators for the member *Concept*.

Table *DegreeOfSynonymy* Value-Set details

Value-Set: DegreeOfSynonymy		
Degree to which a *term* covers the same concept covered by the original *Fully Specified Name* in the US English Edition.		
Value	**Name**	**Description**
0	Unspecified	The degree of synonymy between this *term* and the *FullySpecifiedName* of the associated *Concept* has not been specified. This is the default when a *Description* is not referenced by the *RefSet*.
2	Synonymous	The *term* is a valid synonym for the associated concept.
3	Near synonymous	The degree of synonymy depends on context.
4	Non-synonymous	The *term* is not synonymous with the associated concept.
5	Invalid	The *term* is not a valid synonym for the associated concept.

Table *DescriptionForm* Value-Set details

Value-Set: DescriptionForm		
Value	**Name**	**Description**
0	Unspecified	The *Description* Form has not been specified.
1	Unabbreviated singular	The *term* has not been marked as a plural, abbreviation or symbolic form. It may therefore be regarded as an unabbreviated, singular form.
2	Plural form	The *term* expresses a plural form of the associated *Concept*. In general *SNOMED CT Concepts* are singular but some plural *terms* exist either as a legacy from earlier source terminologies or because they are clinically appropriate.
3	Incomplete or abbreviated form	The *term* expresses the associated *Concept* using abbreviation or acronyms. It may incompletely express the meaning of the associated concept. Note: Incomplete or abbreviated forms may be misinterpreted outside their intended *scope*. Therefore, they should usually be used only as entry *terms* within a *scope* specified by an associated *RefSet*. For example: The *term* "fundus" is an incomplete form common used in the ophthalmology to mean "fundus of eye". However, since the same *term* is also used in other environments to mean "fundus of stomach" or "fundus of uterus" it is essential that it is used in a way that ensure the correct concept is selected for the *scope* in which it is used.
4	Symbolic form	The *term* provides a symbolic representation of the associated concept. For example, UCUM (Unified Code for Units of Measure) provides a symbolic textual rendering of units using a computer processable syntax. *SNOMED CT Concepts* representing units can thus include the symbolic UCUM rendering as *Description* marked as "symbolic form". All UCUM based *Descriptions* would also be referenced by a UCUM *RefSet* enabling them to be selected where required. Note: Symbolic forms should not be subject of general purpose searches.

Table *DescriptionType* Value-Set details

Value-Set: DescriptionType		
Value	**Name**	**Description**
2	Synonym	A *Description* that is not the *Fully Specified Name* but is used as a synonym for the *Concept* specified by a *Fully Specified Name*.
3	FullySpecifiedName	The *FullySpecifiedName* for the associated *Concept*.

Table *InitialCapitalStatus* Value-Set details

Value-Set: InitialCapitalStatus		
Value	**Name**	**Description**
0	Initial capital not significant	The first character of the *term* may be presented in upper or lower case according to its position in a sentence without changing its meaning.
1	Initial capital is significant	The capitalization of the first character of the Term must not be changed. If the first character of the This setting is used to indicate that the term must retain an initial capital (e.g. "Down syndrome") or must not have its initial letter capitalized (e.g. "ml").

Table *IsExclusion* Value-Set details

Value-Set: IsExclusion		
Value	**Name**	**Description**
0	Not an exclusion RefSet	The *RefSetMembers* are intended to be included (rather than excluded) from searches.
1	Exclusion RefSet	The *RefSetMembers* are intended to be included (rather than excluded) from searches.

Table *IsPrioritized* Value-Set details

Value-Set: IsPrioritized		
Value	**Name**	**Description**
0	Not a priortized RefSet	The *RefSetMembers* do not have specified Priority values.
1	Prioritized RefSet	The *RefSetMembers* are intended to be processed based on their specified Priority values.

Table *IsScoped* Value-Set details

Value-Set: IsScoped		
Value	**Name**	**Description**
0	Not a scoped RefSet	The *RefSetMembers* do not include a *scope* attribute indicating the specific scope in which they apply.
1	Scoped RefSet	The *RefSetMembers* include a *scope* attribute indicating the specific scope in which they apply.

Table *RefSetType* Value-Set details (NOTE: A change request for this value set has been submitted during the feedback period. If the request is approved this enumeration might change to 1= Language; 7=Navigation; 9=Simple; 10=Tagged; 11=RefSet Group 12=Aggregation)

Value-Set: RefSetType		
Value	**Name**	**Description**
1	Simple	A *RefSet* used to specify a subset of *SNOMED CT Components* by specifying the components to be included in or excluded from the set.
2	Prioritized	A *RefSet* used to assign priorities to particular *Descriptions* of *Concepts* so that these appear first or are highlighted in a search list.
3	Navigation	A *RefSet* used to represent an alternative hierarchy used for navigation.
4	Tagged	A *RefSet* used to tag additional codes or annotations to a set of *SNOMED CT Components*.
5	Language	A *RefSet* used to specify the *acceptability* and *correctness* of particular *Descriptions* in a particular language or dialect.
6	RefSet Group	A *RefSet* used to group together a set of other RefSets.
8	Aggregation	A *RefSet* representing an alternative aggregation hierarchy.

Table *StatusDescription* Value-Set details

Value-Set: StatusDescription		
Value	**Name**	**Description**
0	Current (active)	Active *Description* – available for current use.
1	Inactive (no reason given)	Inactive *Description* – withdrawn from use without a specified reason.
7	Refers to a different concept	Inactive *Description* – withdrawn from use as the *term* is not an appropriate description of the associated *Concept*. Another *Description* containing the same *term* associated with an appropriate *Concept* may already exist or may have been added.

Table *StatusSimple* Value-Set details

Value-Set: StatusSimple		
Value	**Name**	**Description**
0	Active (current)	Active *SNOMED CT Component* – available for current use.
1	Inactive	Inactive *SNOMED CT Component* – withdrawn from use.

13 Appendix: Components and Component History

Note: The content in this chapter was originally produced as part of the RefSet and History Mechanism Technology Preview documentation under development at the time this draft version is published. It is reproduced here to ensure consistency, although some features like hyperlinks are limited or absent. Table numbers were intentionally removed to avoid conflicts with the data dictionary that would eventually unify documentation related to SNOMED CT data structures. This content would then be replaced by a reference to the appropriate documents once they become available.

13.1 Background

13.1.1 SNOMED CT Components

SNOMED CT consists of a set of uniquely identified components. The primary content of *SNOMED CT* is represented using three component types - *Concepts*, *Descriptions* and *Relationships*. In addition there are a range of supporting component types that enable standard mechanisms for referencing, subsetting, organizing and cross-mapping the primary content.

All *SNOMED CT Components* have:

- A unique identifier that conforms to the *SNOMED CT* Identifier (*SCTID*) specification.
- A status assigned from an enumerated set of *SNOMED CT* status values. This indicates whether the component is intended for active use and, in the case of some component types, provides more specific status information.

13.1.2 Releases, maintenance, and history

SNOMED CT Components published in release files that conform to a published specification.

The content of *SNOMED CT* is subject to a continuous process of maintenance and enhancement which leads to regular publication of revised releases. Between each release and the next release some *Components* are added, some are subject to minor changes and others are withdrawn from active use.

Release files need to enable implementers to access a "snapshot view" of the content of *SNOMED CT* after the release. The "snapshot view" must also include information about *Concepts* and *Descriptions* which are no longer in active use because these may be referenced by patient records or other artifacts created using a previous release.

The "snapshot view" of *SNOMED CT* meets many day-to-day usage requirements. However, some more advanced processing requires additional information about the release in which particular additions or changes were made.

13.1.3 Component representation in current release format

The current release format was designed to provide a simple snapshot view of the release data required by users at a point in time. As a result, there are differences between the representations of different types of *Components* and changes to these *Components* in the release files.

Each release supports the "snapshot view" requirements by including:

- All *Concepts* and *Descriptions* ever released.
    - Marked with their status at the time of the release.

- All active *Relationships* at the time of the release.
- Any *CrossMapSets* intended for use at the time of the release
    - With their constituent *CrossMaps* and *CrossMapTargets*.
- Any *Subset* intended for use the time of the release
    - With their constituent *SubsetMembers*.

Each release also meets the main requirements for change tracking by including:

- *ComponentHistory* indicating the release in which each *Concept* and *Description* was first added and subsequently changed.
- Historical *Relationships* linking inactive *Concepts* to relevant active *Concepts*.

However, there are some limitations to the current approach:

- To make full use of the historical information in the current release format it is essential to have access to the data provided in previous releases;
- The way that history applies to different components differs.

13.1.4 Requirements for a more consistent approach to history

Since the first release of *SNOMED CT* in 2002, various implementers have requested a more complete and consistent approach to representation of *Component* and tracking of changes. In particular concerns have been expressed about the difficulty in identifying significant changes to *Relationships* and about the inconsistent approaches to *CrossMapSets* and *Subsets*.

In addition to improved access to information about changes, some implementers have asked for an incremental release option. Rather than wholesale replacement of one edition by the next, they would prefer to be able to receive and process only the changes between releases. It has been argued that this would be particularly useful to implementers working with multiple *Extensions* and that it would also pave the way for a more frequent content updates in the future.

During 2006 an initial specification was prepared for consideration by the Technical Steering Committee (TSC) of the *SNOMED International Standards Board*. In October 2006 it was agreed that work should proceed to develop a "Technology Preview Release" that would allow implementers to consider the propose enhancements.

13.2 Revised approach to component history

13.2.1 Outline of the changes

In the new release files every row representing a *Component* includes:

- A component identifier (*id*)
- An indication of the date (and optionally time) at which a new *Component* or a change to component became effective (*effectiveTime*)
- An indication of the status of the *Component* at the specified *effectiveTime* (*status*).

13.2.2 Component Identifiers

All *SNOMED CT Components* already have a unique identifier. However, in the case of *CrossMapSets* and *Subsets*, while each set is a uniquely identified *Component*, the individual members of those sets (i.e. *SubsetMembers* and *CrossMaps*) were not separately identified as discrete *Components*.

In order to support history tracking at the level of individual members of a set *CrossMaps* and RefSetMembers (the replacement for *SubsetMembers*) are now recognized as components with a unique identifier. New *SCTID* partitions have been allocated to ensure these identifiers do not clash with the identifiers used for other components.

13.2.3 Component History

Currently the date of the releases in which each *Concept* or *Description* first appeared and was subsequently changed is represented in the *ComponentHistory Table*. The new approach adds a new attribute (*effectiveTime*) to each representation of a component in the release files. As a result there is no longer a requirement for a separate *ComponentHistory Table*.

13.2.4 Component Status

The current distribution form of *Concepts* and *Descriptions* already includes a status attribute but the release forms of other components do not include this attribute. The rationale for omission was that only active components of these other types were released. However, with the new approach this rationale no longer applies and the status attribute has thus been added.

The combined effect of these changes is that all components have three common attributes. In the case of the identifier and status attributes the previous release format named this in a manner that depended on the component type (e.g. *ConceptId*, *DescriptionId*, *RelationshipId*, *ConceptStatus*, *DescriptionStatus*). In the new release format the common attributes are given the same names irrespective of the component type (i.e. Id, *effectiveTime*, Status). The objective of this change is to simplify documentation and implementation, while recognizing common characteristics and behavior of this set of essential attributes.

13.2.5 Unique keys

A significant consequence of the revised distribution form is that the primary key of a row in the distribution file is *id*+*effectiveTime*, rather than *id* alone.

At any given point in time only the most recent row with a given *id* is relevant. Therefore, in normal use the *id* remains the unique key for the *Component*.

Advice on strategies for managing this change is provided later in this document.

13.2.6 Benefits of the revised approach

The proposed new release file format brings together the current state of the component and the history information into a single representation. Furthermore, it provides this information in a manner that is consistent for all types of *Components*.

The new format offers implementers the following benefits:

- Full audit trail of all states of all components over time since the first release of *SNOMED CT*.
- Allows data to be analyzed based on historical or current *Relationships* between components.
- Makes it easier to identify likely impact of changed *Relationships* on existing queries and protocols.
- Avoids the need to process component history and current *Component* data to track changes made.
- Enables new releases to be applied by adding new and changed rows as updates rather than completely replacing a previous release.

13.3 Naming convention changes

13.3.1 Reduction of inconsistent prefix repetition

Review of the current naming conventions applied to attributes reveals considerable inconsistency in relation to whether the name includes or excludes a prefix repeating or abbreviating the component name.

For example:

- *Concept.FullySpecifiedName* – does not repeat the component name "*Concept*".
- *Subset.SubsetName* – repeats the component name "*Subset*".
- *CrossMap.MapSetName* – repeats an abbreviated variant of the component name "*CrossMap*".

A more consistent attribute naming convention has been applied as follows:

- The attribute name should not repeat or abbreviate the component name where the purpose of the attribute is clear from the combined name of the component and attribute.
- The attribute name should include the component name (or abbreviation) where this clarifies the purpose of the attribute.