Universal Terminology

Chapter 3: Identifiers

The $\mathbf{T}_{logy}$is made up of elementary building blocks called anatomical entities. These entities must be uniquely identified in order to organize the database according to the needs of its users.

The objective of this document is to present the unique identifier of entities in the Terminologia Anatomica Humana (TAH). A number of constraints and properties are required for a unique identifier in a database. Although this aspect is not specific to anatomy, it is important that all stakeholders share a common understanding. Strict rules are defined and must be continuously applied throughout the life of the database.

This document constitutes Chapter 3 of the book Universal Terminology, which presents comprehensive documentation on $\mathbf{T}_{logy}$.

Contents

3.1 The global situation

The current number of entities in the TAH is approximately 50,000 and should be able to accommodate substantial growth, depending on future developments, without requiring a redesign of the database structure. Therefore, the design must be ready for a million entities, without any issues or performance degradation. This figure can be easily accommodated, without any particular effort, using today's available systems.

An entity identifier Def:**identifier** is not just an internal pointer; it can be made visible to users, who may need to copy or transfer this identifier occasionally. This is a first prerequisite for using a unique identifier: it must be easy to remember for a short period of time without difficulty. For example, it is common to read the identifier and type it in another window. The solution is to use five-digit positive integers, expandable to six digits. It would be counterproductive to use an ID longer than six digits, due to the limitations of human memory.

For external use of the identifier, outside of the TAH database, if you wish to point to an entity in a scientific communication, it is recommended to prefix the unique identifier with the three letters TAH. The identifier TAH3993 means: access the TAH database and request the internal identifier 3993, which corresponds to the sphenoidal portion of the middle cerebral artery.

It is important to understand that unique TAH identifiers exist independently of a strict database implementation. This is because the IFAA is an international organization whose specific objective is to share knowledge with all members of the scientific community, without economic or political restrictions. As soon as we begin to share information, we immediately lose some ownership, including the freedom to arbitrarily modify critical content. This is exactly what happens with entity identifiers. In the example above, assigning an identifier to an entity and sharing its value implies a commitment by the TAH authors not to modify this identifier in the future. However, if this were to happen anyway, it would need to be governed by restrictive rules and additional information.

This document sets out the rules to be applied by TAH database managers and users in all circumstances. Since it must be approved by FIPAT and therefore by the IFAA, this document is not subject to the control of any individual. Furthermore, it should be accessible to the public at all times, which is the best way to ensure compliance.

3.2 Rules Applicable to Identifiers

The rules applicable to the creation and management of an Def:**identifier** are as follows.

They will be detailed in this chapter.

1	An identifier is an integer between 1 and 999,999.
2	An identifier is issued in ascending order within the authorized range.
3	The existence of an identifier cannot be inferred from the existence of another identifier.
4	No speculation about the current value of the next issued identifier is allowed.
5	The link between an entity's unique core and its identifier is guaranteed to be constant.
6	No one can create an identifier alone. Only a process in the $\mathbf{T}_{logy}$database is authorized to create an identifier.
7	Never reuse an ID that has already been publicly assigned.
8	Maintain a pool of unused IDs and use it at any time.
9	Move deleted entities to their graveyard, but never delete them.
10	Never reactivate an entity unless a deletion error is obvious.
11	List and document deleted entities.
12	Allow resetting to the last provided value.
13	Prepare a list of IDs by type.

Table 3.1: 13 rules applicable to identifiers.

3.3 Meaning of identifiers

Although it is highly preferable for an identifier to have no particular meaning, it is necessary to examine what can and cannot be guaranteed by instances of these identifiers. The arguments presented here may seem trivial, but explaining the important points is never a waste of time.

Why must an identifier be meaningless, that is, be completely anonymous with respect to the entity it represents? The reason is simple: an identifier must be permanent and of almost unlimited duration, while the quality and properties of entities are likely to change over time.

We will discuss below in turn all the rules applicable to identifiers.

3.3.1 Short Positive Integer

This aspect has already been discussed above.

Entity Identifier
`An identifier (definition) is an integer value between 1 and 999,999.`

3.3.2 Ascending Sequential Delivery

There are at least two possible solutions: delivering the identifier as a sequence or delivering it randomly within the specified range. Both solutions are generally available, and each has advantages. Random delivery is more indiscriminate and secure than sequential delivery, but, from a pragmatic point of view, the latter allows for greater human control. This is not the place to discuss two different techniques, each perfectly valid. The original authors of the TAH database

opted for sequential delivery. In fact, this is the approach most commonly used by database developers.

We will see later that during database development, some identifiers are created by mistake, and we don't want to lose them. If they haven't yet been officially published, they can be recycled and reused without any problem. A pool of unused identifiers will then be created. Retrieving an identifier from this pool is a legitimate operation, equivalent to delivering a new identifier.

The sequential delivery of identifiers implies a time interval between their successive deliveries. It is therefore possible to infer a time interval from the numerical difference between two identifiers. However, this exercise is rather approximate for two reasons: the time scale can change without warning; the existence of a pool of unused identifiers always makes a break in the sequence possible. Consequently, such inferences are considered unreliable and therefore inadequate. Despite this, in practice this method is sometimes used by $\mathbf{T}_{logy}$developers.

Sequential Delivery
`An identifier (definition) is delivered in an increasing sequence within the permitted range.`

3.3.3 Existence of an Identifier

The exercise of deducing the existence of an identifier from the existence of a higher-level identifier is strictly prohibited. As we will see later, we will take care to keep the sequence of identifiers delivered as compact as possible. However, this does not mean that all previously delivered identifiers are always significant in the $\mathbf{T}_{logy}$. Database administrators are free to organize themselves as they wish and without notice.

When using database procedures, an error message is generated each time a user uses a non-existent identifier. This is a way to determine the existence of an identifier. Other procedures external to the database, but based on the actual contents of the database, are subject to the same conditions.

Existence of an Identifier
`The existence of an identifier (definition) cannot be inferred from the existence of another identifier.`

3.3.4 Position of the Last Delivered Identifier

The database remembers the last delivered identifier in order to deliver the next one. However, this value is generally not directly accessible to users. Therefore, it should be considered inaccessible to users and their applications. However, database administrators can access and even modify it. This is not a recommended exercise. Furthermore, this type of operation is implementation-dependent. Under no circumstances may users speculate on its availability.

No Speculation on Identifier
`No speculation (definition) on the value of the next delivered identifier is allowed.`

3.4 Permanence of identifiers

3.4.1 Permanence of the Entity-Identifier Link

An identifier is automatically created each time a new entity is created in the entity table and its value is assigned to it. However, the entity remains empty and can be populated with any content. In other words, the database ignores the fields defining an entity and its identifier. Furthermore, an unexpected procedure can swap the contents of two entities, resulting in both entities still being present, but their identifiers having been swapped. In other words, the database offers no guarantees regarding the content of entities to which an identifier has been assigned.

It is the responsibility of database managers to maintain a consistent database. In particular, they guarantee the permanent link between an identifier and the core of an entity, which are the attributes of its uniqueness.

However, responsibility for database content may be shared among multiple users, and ensuring the stability of entity-identifier relationships is a difficult task.

Identifier Permanence
`The permanence of an identifier (definition) is the link between the unique core of an entity and its identifier, whose consistency and durability are guaranteed over time.`

As a result of this definition, an identifier must never be reused. Thus, when an entity is no longer used or has been replaced by other entities, it must be given the status *deleted*, which is interpreted as an absence in the terminology. However, this entity still exists in the database, but is not visible to users of the $\mathbf{T}_{logy}$.

Many examples are available in TA98, where a few dozen entities were deleted during the update of this version of the terminology. For example, EN:*thigh nutrient artery* has been replaced by EN:*proximal thigh nutrient artery* and EN:*distal thigh nutrient artery*

3.4.2 No one can create an identifier

Identifiers are created by a unique process internal to the database, in the reference version of that database and regardless of distributed copies of the database. This is strictly the only way to create a new identifier.

Centralized Identifier Generation
`A centralized database (definition) is the only way to create new identifiers, and no one can create an identifier alone.`

3.4.3 Prohibition on Reusing a Deleted Identifier

According to the principle of permanence of the entity-identifier link, it is essential to never reuse an identifier after deletion, at least not if it has been published. If an identifier is reused, an external user could inadvertently assume that it is still linked to the previous entity, even if the link has disappeared.

Take the example of Wikipedia, which indexes articles in the TA as well as other sources. The index used is based on the unique identifier. If the current rule is not applied, Wikipedia will point to the wrong entity, forcing it to revise all its applications. The existence of TAHs in the scientific community strongly depends on these criteria. FIPAT has no choice, and we must follow this common-sense rule.

Reuse prohibited
`Reuse of an identifier (definition) is prohibited if this identifier is part of a publicly validated version of the terminology.`

3.5 Managing Unused Identifiers

As part of the daily operations of a dynamic database that accepts updates and upgrades, it is unfortunately possible to create a new entity by mistake. For example, an entity that already exists is sometimes created elsewhere. When the error is discovered, the newly created entity is deleted and its identifier is made available for another entity. This identifier is available because the new temporary entity is unknown to the rest of the world. In fact, hundreds of such entities can be inadvertently created in a single work week. A management strategy must be implemented.

Any identifier assigned and released in error must be recycled for future use as soon as the error is discovered. A pool of these identifiers must be created. This is essentially a list of empty entities. When a false identifier is found, the entity is emptied and moved elsewhere in the database. When a new identifier is needed, instead of creating a new one, it is possible to take one from the pool.

Management of unused identifiers
`Management of unused identifiers (definition) is implemented in the form of a pool of unused identifiers available upon request.`

3.6 Entity Deletion

Regardless of how the database evolves, sooner or later, some entities are no longer used and must be deleted.

This means that these entities become inaccessible to ordinary users and their applications. This does not mean that they must be physically deleted.

3.6.1 Physical Deletion Prohibited

In the case of TAH, entity deletion should be a rare exception: what was once present in a human body is not destined to disappear overnight. Therefore, for security reasons and to maintain a deletion history, it is necessary to decide whether to move deleted entities to a pool or graveyard. This is a one-way transfer, as these entities have been published and their identifiers are permanently prohibited from reuse.

Deleted Entity Graveyard
`A deleted entity graveyard (definition) is a database pool where all deleted entities are stored.`

A list of deleted entities is available. Indeed, these entities have officially existed for some time and can therefore never be lost. A casual user can ask questions about an entity at any time: a response is expected, even for deleted entities.

It should be noted that some 7,500 identifiers were made public in January 2013 in the online version of TA98. Some of these identifiers have been removed during the current revision.

3.6.2 Reanimation in case of error only

Unless necessary to reanimate a deleted entity, such an operation is strictly prohibited. However, it may happen that an entity has been deleted by mistake, and correcting this error constitutes a reanimation. It must be proven that the new entity and the old one are indeed the same entity.

Reanimation
`A reanimation (definition) is the recovery of an entity deleted by mistake.`

In practice, reanimation is more common than expected. In a database, new entities are created with new identifiers. But deleted identifiers are lost unless a reanimation process is scheduled. Without such a process, one risks encountering unexplained gaps in the list of identifiers. Although revival is not strictly mandatory, the authors of the $\mathbf{T}_{logy}$database preferred to revive unexpectedly deleted entities. This way, the list of identifiers is continuous from zero to the last generated entity (nearly 55,000 in July 2025). However, it may contain unused identifiers.

3.7 Minimum Database Requirements

The database system must meet a set of minimum requirements for the proper administration of the TAH database. These requirements are explained below.

3.7.1 Next Delivered Identifier Control

This is clearly a mandatory feature, which remains relatively poorly documented in some database systems. In theory, such a need should never arise, but it does occur in practice. For example, entities may be accidentally deleted, regardless of their origin: user, system, etc. In this case, the database manager must reset the last delivered identifier value to any value, and then create new entities from there, until the last value is restored to its initial value.

ID Reset
`An ID reset (definition) is a special process that allows the generation of IDs to be restarted at any value.`

This process allows for the necessary resuscitation operation.

3.7.2 List Assigned IDs by Type

It can be useful to be able to provide statistics on ID usage. In particular, evidence of the compactness of IDs within their value range may be necessary for optimization purposes.

ID List
`The ID List (definition) is a list by entity type for assigned IDs and a list of available free IDs for generic entities (less than 25,000) or other entities (greater than 25,000).`

These lists are mandatory for proper database management. They are only available internally in the database.

3.8 Identifiers and Granularity

The simple identifier, as defined above, applies to anatomical entities. How can other atoms of the granularity be identified from entity identifiers? How can units, words, and lists be identified? This is the subject of this section.

3.8.1 Unit Identification

Units are composed of one to five entities depending on their type. Therefore, the unit identifier is always chosen from one of these entities. Of course, when the unit consists of a single entity, the entity identifier becomes the unit identifier. This is true for simple entities, taxonomic entities, vocabulary entities, and others.

For the set entity, the mixed set entity, the pair entity, and the pair set entity, the rule is to select the generic entity identifier as the unit identifier.

Unit ID
`A unit ID (definition) is the ID of its generic entity, between 1 and 25,000.`

For example, consider the unit LA:*931 humerus (par)*: this unit is an even unit and is therefore composed of four entities: the generic entity LA:*931 humerus*, the specific even entity LA:*32729 humerus (pair)*, the specific left entity LA:*32730 left humerus* and the specific right entity LA:*32731 right humerus*. The ID of the generic entity is 931 and becomes the ID of the unit. Since generic entities always have an ID lower than 25,000, this characteristic is also true for unit IDs.

3.8.2 List Identification

Lists are generally hierarchical lists according to one of the active hierarchies of the $\mathbf{T}_{logy}$: partonomy, taxonomy, or the TA98 hierarchy. Lists are identified by the identifier of the superior entity that defines them. Formally, the identifier must also contain the hierarchy type to avoid duplicates. But this aspect has not yet been implemented!

List Identifier
`A list identifier (definition) is the identifier of its hierarchically superior entity.`

For example, the entity *cochlea* of the inner ear is the subject of a list of more than 20 entities in the partonomic hierarchy. This list is at level P4. This list is cochlea.

It should be noted here that the hierarchically superior entity is the specific even entity 26108 and not the generic entity 7271. The identifier for this list is therefore composed of 26108 and P4. The same number 26108 can theoretically appear in other hierarchies.

3.8.3 Word Identification

Words are represented by simple entities and are therefore identified by the corresponding entity identifier.

Vocabulary Identifier
`A vocabulary identifier (definition) is the identifier of its simple entity.`

For example, the word la:*arche*, which occurs more than a hundred times in the $\mathbf{T}_{logy}$, can be viewed on demand.

3.9 TA98 Identifiers

This TAH terminology is derived from an earlier version, Terminologia Anatomica, published in 1998, known as TA98. The core of the terminology is therefore this relatively recent source, from which all entities have been taken, subject to modifications due to the evolution of medical science, particularly anatomy.

In addition, many entities have been added for the sake of precision and to be as explicit as possible. For example, TA98 only cites a single entity, the *thoracic vertebrae*, while the TAH explicitly creates 12 distinct entities, each with its own identifier. By providing explicit identifiers for each vertebra, some users can use them as pointers in a medical record, since these identifiers are guaranteed to be permanent over time without limit. Hundreds of new entities have been created in the TAH, in the name of this principle of explicitation.

There was therefore a significant difference in the presentation details between TA98 and TAH. Such an initiative disrupted the TA98 identifier system, to the point of making them unusable in the new perspective. Reading this chapter clearly shows that the TA98 identifiers had major flaws.

The TA98 identifiers were 12-character codes, such as *A02.2.02.101 Atlas*. Characters 1, 4, 6, and 9 are constant and therefore serve no purpose in identification and could be deleted without harm. Furthermore, these codes partially mimic the partonomic hierarchy. But this hierarchy is explicit in the database, so the redundancy from the code serves no purpose, and moreover, it poses a risk of error in the event of modification.

This is, in fact, the main flaw of the TA98 code. If an entity is assigned a code dependent on its hierarchical position, if this position is changed, the code must be changed, so its permanence is not guaranteed. For example, the

en:*extraocular muscles* were part of the chapter on sense organs in TA98, but were moved to the chapter on the muscular system in the TAH. Such a move was impossible in TA98 without modifying the identifiers.

Another flaw is the explicit sequential nature of the TA98 codes. Indeed, the last three characters are in ascending sequential order. It is possible to delete an entity from a sequence without causing harm, but it is not possible to add one in the middle of a sequence without breaking the sequence. However, for many lists in the $\mathbf{T}_{logy}$, there is a natural order of presentation. For example, for the stomach, it is natural to present the en:*cardia*irst and the en:*pyloric portion*ast. Such an arrangement was impossible in TA98 without modifying the identifiers.

As a result of the arguments put forward above, the TA98 identifiers were abandoned in favor of a solution consistent with the criteria set out in this chapter and with undeniable scientific foundations.

3.10 Log of updates

26 Jul 2025 Full revision of the chapter. From now on, the source text for this chapter is the French version.

30 Mar 2022 Standardisation of the file as a chapter.

21 Oct 2021 Creation of the file.

3.11 Credentials

This document is part of the work *Universal Terminology*, which accompanies the website Terminologia Anatomica. It expresses the vision of the authors of the $\mathbf{T}_{logy}$on the foundations of the science of ontology, supporting the terminology presented here. Although it is as accurate as possible, close to the reality of the terminology database and the software that supports it, approximations, errors, and ambiguities are possible and must be considered independent of their will and intentions.

Any comments regarding the content of the website and its presentation are welcome. An appropriate response will be provided if necessary.

Authentic URL of this file:
https://ifaa.unifr.ch/Public/TNAEntryPage/help/Chap03FR.pdf
providing access to the latest update of the document.