

## 3.3. SNOMED CT storage options for effective retrieval

The form in which records are represented may have a substantial impact on the efficiency, accuracy and completeness of retrieval. The forms that best suit retrieval may differ from the forms that are required to meet the principles of clinically safe and legally valid [electronic health records](#) .

### Storing information as entered

This option leaves information in the form entered in the [electronic health record](#) with no additions to assist future retrieval. The application must do all the work needed to locate the required records and compute subsumption and [equivalence](#) when a request is made to retrieve data.

### Using an Expression Repository

An innovative approach to the issues raised by literal storage of [postcoordinated expressions](#) is to implement an [expression repository](#). Each unique [expression](#) used in the system is stored in a referenced database table and assigned an internal unique [Identifier](#) (e.g. a [UUID](#)). When an [expression](#) is used in a clinical record entry the unique [expression](#) id is used to reference the [expression](#) in the repository.

The key advantages of this approach of this approach are:

- The [expression Identifier](#) can have a fixed size whereas a [postcoordinated expression](#) is of variable and indeterminate size. This significantly improves storage and index efficiency.
- The [expression repository](#) can also be used to store [normal form](#) representations of each [expression](#) and to relate these to the original [expression](#). This optimizes performance for [expression normalization](#) during retrieval.
- The [expression repository](#) could also be processed by a [Description Logic Classifier](#) and a [transitive closure](#) table of all the [expression](#) used in the application could then be generated. [postcoordinated](#) retrieval would then be highly optimized by using the [transitive closure](#) to test a single join between each predicate and the candidate [expressions](#) .

Related Links

- [7 Testing and Traversing Subtype Relationships](#)
- [Optimisation of normalization and expression subsumption testing](#)
- [7.5 Optimizing concept subsumption testing](#)
- [Testing subsumption and equivalence between expressions](#)

### Minimizing postcoordination

One possible approach to optimization of retrieval is to [transform](#) the original stored information into an equivalent representation with the minimum number of [postcoordinated](#) components.

The objective of this approach is to allow the generation of simple indices for the [precoordinated](#) representation. It is then possible to undertake most retrievals using the [116680003](#) [is a] [subtype hierarchy](#) to compute whether [Concepts](#) in the record are [subtypes](#) of the [Concepts](#) used to specify retrieval. Where [postcoordination](#) is required, the minimum number of additional tests are required to confirm that a [Concept](#) in the record meets the specified retrieval criteria.

One difficulty with this approach is that there may be more than one representation that requires the same degree of [postcoordination](#). This is discussed in more detail and illustrated in [Transforming expressions to normal forms](#) .

If this approach is adopted additional rules need to be applied to determine the choice between alternatives with a similar number of [postcoordinated](#) components.

Example:

In the hypothetical example illustrated in , the [Concept](#) "red steel pedal bicycle", for which no [precoordinated](#) representation exists, could be represented as:

"red pedal bicycle" + |make of| = |steel|

or

"steel pedal bicycle" + " color " = "red"

Both are equally close to the objective of minimizing [postcoordination](#). A rule is needed to determine which of these is preferred. There is no obvious right or wrong solution to this but a simple rule that places the attributes in an [order](#) will, if applied consistently, allow all [postcoordinated](#) representations to be reduced to a single minimized form.

### Maximizing postcoordination

An alternative approach is to expand any [precoordinated concepts](#) in the record to their fullest possible [postcoordinated](#) forms. This general type of [transformation](#) is illustrated in [Transforming expressions to normal forms](#) .

This approach requires a richer record structure but has the advantage that there are three possible end-points to [postcoordination](#), each of which ensures that any computably equivalent representations of [Concepts](#) will expand to an identical [postcoordinated](#) form. The three end-points are summarized here:

- Short [canonical form](#):
  - This is the most parsimonious of the three options.
  - A [concept](#) is represented as the combination of:
    - [Subtype relationships](#) with its most proximate [primitive](#) supertypes;
    - The recorded [qualifier](#) values and/or [defining characteristics](#) that distinguish it from its most proximate [primitive](#) supertypes.
- Long [canonical form](#):
  - This option is more verbose as it includes some redundancy.
  - A [concept](#) is represented as the combination of:
    - [Subtype relationships](#) with its most proximate [primitive](#) supertypes;
    - All of its recorded [qualifier](#) values and/or [defining characteristics](#), irrespective of whether they are shared by its most proximate [primitive](#) supertypes.
- Exhaustive [postcoordinated](#) form:
  - This option is extremely verbose.
  - A [Concept](#) is represented as a combination of:
    - [Subtype relationships](#) with all of its [supertype ancestors](#)
    - All of its recorded [qualifier](#) values and/or [defining characteristics](#), irrespective of whether they are shared by its most proximate [primitive](#) supertypes.

If the retrieval criteria are expressed in a similar form, a relatively simple [query](#) can interrogate the record for all entries with a matching set of [primitive Concepts](#) and specified characteristics.