_units_detail. The latter is a character field describing the units; the **_units** attribute is a code that may be interpreted by machine. In DDL1-based dictionaries, type codes are purely conventional, and there is no mechanism for converting units or relating quantities in different units. Table 3.1.5.1 lists the units codes used in the DDL1-based dictionaries described in this volume. There can be some inconsistencies: two codes ('s' and 'sec') are already in use to indicate the time unit of seconds.

The original CIF paper (Hall *et al.*, 1991) described a convention allowing physical quantities to be listed in a CIF in units other than those specified in the dictionary. Under this convention, a data name representing a value expressed in different units could be constructed by appending one of a series of known 'units extension codes' to the standard data name. Thus **_cell_length_a_pm** would represent a cell length expressed in picometres instead of the default ångströms. This approach is now deprecated, and all quantities must be expressed in the single unit permitted in their definition block. However, to allow the formal validation of old CIFs, a 'compatibility dictionary' is available which defines all data names that could have been constructed under this convention in a properly DDL1.4-compliant form. *This dictionary should only be used for validating old CIFs, and must not be used to construct new data files.* The dictionary is called cif_compat.dic in the IUCr CIF dictionary register (see Section 3.1.8.2).

Table 3.1.5.1. *Units codes and their interpretation in DDL1-based dictionaries*

Unit code (**_units**)	Meaning (**_units_detail**)
A	Ångströms
A^-1^	Reciprocal ångströms
A^2^	Ångströms squared
A^3^	Ångströms cubed
Da	Daltons
K	Kelvins
Kmin^-1^	Kelvins/minute
Mgm^-3^	Megagrams per cubic metre
\ms	Microseconds
deg	Degrees
deg/min	Degrees per minute
eV	Electronvolts
e_A^-3^	Electrons per cubic ångström
fm	Femtometres
kPa	Kilopascals
kV	Kilovolts
kW	Kilowatts
mA	Milliamperes
min	Minutes
mm	Millimetres
mm^-1^	Reciprocal millimetres
s	Seconds
sec	Seconds

3.1.6. Constructing a DDL2 dictionary

The DDL2 dictionary definition language was designed to specify a relational data model and has provision for including within a dictionary tables of relationships between data entries. Like a relational database which contains tables describing the data tables in the database, DDL2-based dictionaries contain definition blocks describing CIF categories, units and relationships as well as data items.

Unlike DDL1 dictionaries, a DDL2 dictionary is presented as a single data block. Within this data block a number of looped lists describe properties of the dictionary as a whole, or properties and relationships shared across the items defined in the dictionary. Typically these are: the dictionary name, version identifiers and revision history; the category groupings that give structure to the items defined by the dictionary; the labels that identify closely related data items; and the physical units employed in the dictionary, their definitions in terms of base units and their interconversion factors.

Definitions of individual data items and categories are contained within save frames. While the save frames are not referenced by name in any dictionary application, they permit multiple occurrences of data definition tags within the scope of a single data block and are therefore suitable for structuring a data dictionary. It is a convention that the name of a save frame defining a category is given in capitals, and the name of a save frame for a definition of a data item is given as lower-case. For example, **save_ATOM_SITE** is the name of the save frame defining the category with the **atom_site** identifier, while **save__atom_site.details** is the name of the save frame holding the definition of the individual data name **_atom_site.details** (note how the initial underscore character of the data name is preserved following the initial **save_** string of the save-frame name).

As with DDL1 dictionaries, the name of the dictionary itself (given by the data name **_dictionary.title**) is usually of the form **cif_*identifier*.dic**, where the *identifier* is a short code for the topic area of the dictionary (*e.g.* 'img' for the image dictionary, 'sym' for the symmetry dictionary).

As is invariable with DDL2 data names, the names themselves are formed from the category name separated by a full stop from the specific descriptor of the item.

Fig. 3.1.6.1 shows the structure of the macromolecular CIF dictionary. The ordering of the various looped lists and save frames is of no significance for machine parsing. The sole data block has the same name as the dictionary title string and the data block is introduced by the dictionary identification data items. The dictionary revision history introduces the file, followed by information about the extended data types and physical units used within the current dictionary. These are followed by the lists of closely related items (corresponding to 'irreducible sets' in DDL1 dictionaries and called 'subcategories' in the terminology of DDL2) and lists of category groupings. The body of the dictionary contains category and item definitions. Each category definition is followed by the definitions of its component data items. The ordering is alphabetic by category and then alphabetic by item name within categories.

3.1.6.1. Dictionary identification

Dictionary files must contain information that unambiguously states their identity and version. In DDL2-based dictionaries this is done using the dictionary attributes described in Section 2.6.6.4. The name of the data block comprising the whole content of a DDL2 dictionary is by convention the same as the dictionary identification string given as **_dictionary.title**. This value is repeated as the value of **_dictionary.datablock_id** (see Example 3.1.6.1) for use in checking the consistency of the dictionary.

The dictionary history is also an important audit record of changes to the dictionary content. Unlike in DDL1-based dictionaries where the history is contained in a single field, DDL2 provides a looped list of version labels, dates and annotations. For convenience, the history records in large DDL2-based dictionaries are sometimes placed at the end of the dictionary file.

3.1.6.2. Subcategory definitions

In the DDL1 formalism, particular relationships between data items may sometimes be stated within a text description or may be implied by the organization of the dictionary (where several data

```
data_mmcif_std.dic

    _dictionary.title           mmcif_std.dic
    _dictionary.version         2.0.09
    _dictionary.datablock_id    mmcif_std.dic
                          (a)

     loop_
         _dictionary_history.version
         _dictionary_history.update
         _dictionary_history.revision   .   .   .
                          (b)

     loop_
    _sub_category.id
    _sub_category.description    .   .

     loop_
    _category_group_list.id
    _category_group_list.parent_id
    _category_group_list.description   .   .   .
                          (c)

    loop_
    _item_type_list.code
    _item_type_list.primitive_code
    _item_type_list.construct
    _item_type_list.detail

    loop_
    _item_units_list.code
    _item_units_list.detail   .   .

     loop_
    _item_units_conversion.from_code
    _item_units_conversion.to_code
    _item_units_conversion.operator
    _item_units_conversion.factor   .   .   .   .
                          (d)


  save_CATEGORY_A   .   .   .   save_
    save__category_a.item_1   .   .   .   save_
    save__category_a.item_2   .   .   .   save_
    save__category_a.item_3   .   .   .   save_

  save_CATEGORY_B   .   .   .   save_
    save__category_b.item_1   .   .   .   save_
    save__category_b.item_2   .   .   .   save_
                          (e)
```

Fig. 3.1.6.1. Schematic structure of the macromolecular CIF dictionary. (*a*) Dictionary identifiers. (*b*) Dictionary history. (*c*) Subcategory and category group listings. (*d*) Data types, units descriptions and conversion tables. (*e*) Multiple category and item definition blocks.

items are defined in the same data block and are understood to share the common attributes itemized in that data block).

Within DDL2, there are mechanisms for more formal and machine-parsable statements of relationships. The `_sub_category.id` attribute is a label shared by several data items within a category that are related in a specific way described by the associated `_sub_category.description` attribute. The relationships may be rather general, such as elements of a matrix; or they may be specific physical properties or attributes, such as the collection of axis lengths of a unit cell. The dictionary should list all such labels that occur within its included data definition blocks. Example 3.1.6.2 is an extract from the macromolecular dictionary.

3.1.6.3. Category groupings

In the DDL2 data model, a *category* of data corresponds to a set of related data items that may be stored in a single relational database table. A number of such tables may collectively describe the complete properties of some physical object. This is expressed formally by assigning the same label (`_category_group.id`) to the relevant categories. While relationships between categories are implied in DDL1 dictionaries by the hierarchical structure of the names of data items, in DDL2 dictionaries the relationships are formally stated.

For subcategories, the category-group relationships present in the dictionary are listed in a separate looped list. Example 3.1.6.3 is an extract from the macromolecular dictionary. The `inclusive_group` entry shows the common parentage of all categories (and ultimately all data items) in the dictionary.

Example 3.1.6.1. *DDL2 dictionary identification entries.*

```
data_mmcif_std.dic

    _dictionary.title           mmcif_std.dic
    _dictionary.version         2.0.09
    _dictionary.datablock_id    mmcif_std.dic

     loop_
    _dictionary_history.version
    _dictionary_history.update
    _dictionary_history.revision
   0.1.1  1993-02-11
;  Highlighted all notes with # %%%%% surrounds.
;
   .  .  .
```

Example 3.1.6.2. *DDL2 subcategories defined in the mmCIF dictionary.*

```
loop_
    _sub_category.id
    _sub_category.description
    'fractional_coordinate'
; The collection of x, y, and z components of a
  position specified with reference to unit cell
  directions.
;
   'matrix'
; The collection of elements of a matrix.
;
   'miller_index'
; The collection of h, k, and l components of the
  Miller index of a reflection.
;
   'cell_length'
; The collection of a, b, and c axis lengths of a
  unit cell.
;
   'mm_atom_site_label'
; The collection of alt id, asym id, atom id, comp id
  and seq id components of the label for a
  macromolecular atom site.
;
```

3.1.6.4. Category definitions

In the DDL2 formalism, a category of data items may be mapped to a relational table. The dictionary entry for a category includes the name of the category (an identifying label which is referenced by the `_item.category_id` attribute of each component data item) and a list of the category groups of which it may be considered a member. The category *key* is explicitly specified – that is, the data item (or group of items) that uniquely identifies an individual row in a table of data of that category.

Where a category encompasses a set of data items that are not normally specified in a looped list, the category may nevertheless be taken to represent a degenerate table with a single row, and therefore there is still a category key. For degenerate categories the key value is often set equal to the name of the parent data block.