Example 3.1.5.3. *A simple definition of a data item describing a physical quantity.*

```
data_diffrn_ambient_temperature
    _name               '_diffrn_ambient_temperature'
    _category                    diffrn
    _type                        numb
    _type_conditions             esd
    _enumeration_range           0.0:
    _units                       K
    _units_detail                kelvin
    _definition
;      The mean temperature in kelvins at which the
       intensities were measured.
;
```

specified (`0.0:`) indicates that it may be any non-negative real number. The physical units of the quantity are also indicated.

The `_definition` attribute is a concise human-readable documentation of the meaning associated with the data name.

Example 3.1.5.4 is taken from the powder dictionary and illustrates a data item that can have only one of a limited set of values. This data item indicates the geometry of the experiment. The associated data value is of type *char* and may legally take only one of the two possible values listed.

3.1.5.4.2. *Looped data*

Many of the attributes of looped data items, such as their physical units or valid numerical values, may be defined in exactly the same way as for non-looped data. However, more care needs to be taken to describe the relationships between different looped data items.

Consider the following example listing of some three-dimensional atom-site coordinates and displacement parameters.

```
loop_
     _atom_site_label
     _atom_site_fract_x
     _atom_site_fract_y
     _atom_site_fract_z
     _atom_site_U_iso_or_equiv
     _atom_site_thermal_displace_type
    O1 .4154(4) .56990(10) .3026000   .0600(10) Uani
    C2 .5630(5) .5087(2)   .32460(10) .060(2)   Uani
    C3 .5350(5) .4920(2)   .39970(10) .0480(10) Uani
    N4 .3570(3) .55580(10) .4167000   .0390(10) Uani
    C5 .3000(5) .6122(2)   .35810(10) .0450(10) Uani

loop_
     _atom_site_aniso_label
     _atom_site_aniso_U_11
     _atom_site_aniso_U_22
     _atom_site_aniso_U_33
     _atom_site_aniso_U_12
     _atom_site_aniso_U_13
     _atom_site_aniso_U_23
O1 .071(1) .076(1) .0342(9) .008(1)  .0051(9) -.0030(9)
C2 .060(2) .072(2) .047(1)  .002(2)  .013(1)  -.009(1)
C3 .038(1) .060(2) .044(1)  .007(1)  .001(1)  -.005(1)
N4 .037(1) .048(1) .0325(9) .0025(9) .0011(9) -.0011(9)
C5 .043(1) .060(1) .032(1)  .001(1) -.001(1)   .001(1)

loop_
     _geom_bond_atom_site_label_1
     _geom_bond_atom_site_label_2
     _geom_bond_distance
      O1  C2   1.342(4)
      O1  C5   1.439(3)
      C2  C3   1.512(4)
      C2  O21  1.199(4)
```

These loops, or tables of values, are properties of atom sites, each identified by a label such as O1. The definition of a data name such as `_atom_site_U_iso_or_equiv` expresses this by using the DDL1 `_list_reference` attribute (Example 3.1.5.5).

Example 3.1.5.4. *A data item that can take only one of a discrete set of allowed values.*

```
data_pd_spec_mount_mode
    _name                         '_pd_spec_mount_mode'
    _category                      pd_spec
    _type                          char
    loop_ _enumeration             reflection
                                   transmission
    _definition
;      A code describing the beam path through
       the specimen.
;
```

Example 3.1.5.5. *Definition relating a looped data item to the item used to identify a 'loop packet', or row of entries in a table.*

```
data_atom_site_U_iso_or_equiv
    _name               '_atom_site_U_iso_or_equiv'
    _category                      atom_site
    _type                          numb
    _type_conditions               esd
    _list                          yes
    _list_reference               '_atom_site_label'
```

Example 3.1.5.6. *Definition of a mandatory item within a loop.*

```
data_atom_site_label
    _name                         '_atom_site_label'
    _category                      atom_site
    _type                          char
    _list                          yes
    _list_mandatory                yes
    loop_ _list_link_child
          '_atom_site_aniso_label'
          '_geom_bond_atom_site_label_1'
          '_geom_bond_atom_site_label_2'
```

For an entry in the table to make sense, the site identifier must be present, so the definition for `_atom_site_label` declares it a mandatory item within its list (Example 3.1.5.6).

It is common for an atom-site identifier to be used in several related tabulations in a particular crystal structure description, and in a CIF description this means that it may occur in several different looped lists. The dictionary definition gives a formal account of this by listing the data names in other looped lists which are just different manifestations of this same item. This is done using the `_list_link_child` attribute, which identifies the data names to which the one being currently defined is 'parent'. In Example 3.1.5.6 (which is a subset of the full list in the core dictionary), `_atom_site_aniso_label`, `_geom_bond_atom_site_label_1` and `_geom_bond_atom_site_label_2` are identified as children of `_atom_site_label`.

It can been seen immediately that `_atom_site_aniso_label` is the atom-site identification label appearing in the second table in the example listing above, and the `_geom_bond_` items are clearly atom-site labels in a table of bonding properties between specified sites. There is, however, a difference between the two secondary tables: the bond-properties table is described by data items in the GEOM_BOND category, but the table of anisotropic displacement parameters includes data names that have the same `_category` attribute as the coordinate data items, namely ATOM_SITE. The latter is an example of multiple lists or tables belonging to the same category, a feature permitted only in DDL1-based data files.

3.1.5.4.3. *Units*

The physical units in which a quantitative data item must be expressed are identified by the DDL1 attributes `_units` and `_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