

2.6. SPECIFICATION OF A RELATIONAL DICTIONARY DEFINITION LANGUAGE (DDL2)

2.6.6.1.6. *ITEM_ENUMERATION*

The DDL category *ITEM_ENUMERATION* holds lists of permissible values for a data item. Each enumerated value is specified in item *_item_enumeration.value*, each of which may have an associated description item *_item_enumeration.detail*. The combination of items *_item_enumeration.name* and *_item_enumeration.value* form the key for this category. The parent definition of the former item is defined in the category *ITEM*. Multiple unique enumeration values may be specified for each data item.

2.6.6.1.7. *ITEM_EXAMPLES*

The DDL category *ITEM_EXAMPLES* is provided to hold examples associated with individual data items. An example specification consists of the text of the example, *_item_examples.case*, and an optional comment item, *_item_examples.detail*, which can be used to qualify the example. Multiple examples may be provided for each item.

2.6.6.1.8. *ITEM_LINKED*

The *ITEM_LINKED* category defines parent–child relationships between data items. This provides the mechanism for specifying the relationships between data items that may exist in multiple categories. Link relationships are most commonly defined between key items, which form the keys for many different categories.

In the DDL definition, all child relationships are expressed within the parent category.

Because the item *_item_linked.parent_name* has been defined as an implicit item, the child relationships can be specified most economically in the parent category where the parent item name can be automatically inferred. If link relationships are specified in a child category, then both parent and child item names must be specified.

Both parent and child item names in this category are children of *_item.name*, which ensures that all link relationships can be properly resolved. However, it is possible to define cyclical link relationships within this category. Any implementation of this DDL category should include a method to check for the existence of such pathological cases.

2.6.6.1.9. *ITEM_METHODS*

The *ITEM_METHODS* category is used to associate method identifiers with data items. Any number of unique method identifiers may be associated with a data item. The method identifiers reference the full method definitions in the parent *METHOD_LIST* category.

2.6.6.1.10. *ITEM_RANGE*

The *ITEM_RANGE* category defines a restricted range of permissible values for a data item. The restrictions are specified as one or more sets of the items *_item_range.minimum* and *_item_range.maximum*. These items give the lower and upper bounds for a permissible range. To specify that an item value may be equal to the upper or lower bound or a range, the minimum and maximum values of the range are equated. The special STAR value indicating that a data value is not appropriate (denoted by a period, ‘.’) can be used to avoid expressing an upper or lower bound value. When limits are applied to character data, comparisons are made following the collating sequence of the character set. When limits are applied to abstract data types, methods must be provided to

define any comparison operations that must be performed to check the boundary conditions.

2.6.6.1.11. *ITEM_RELATED*

The *ITEM_RELATED* category describes specific relationships that exist between data items. These relationships are distinct from the parent–child relationships that are expressed in the category. The related item is identified as the item *_item_related.related_name* that is a child of *_item.name*.

Item relationships defined by *_item_related.function_code* in this category include some of the following (Table 2.6.5.1): an item is related to another item by a conversion factor; an item is a replacement for another item; an item is replaced by another item; an item is an alternative expression of an item; items which differ only in some convention of their expression; and items which express a set of related characteristics. One can also identify whether the declaration of an item is mutually exclusive with its alternative item. Multiple related items can be associated with each data item and multiple relationship codes can be specified for each related item.

2.6.6.1.12. *ITEM_STRUCTURE*

The *ITEM_STRUCTURE* category holds a code which identifies a structure definition that is associated with a data item. A structure in this context is a reusable matrix or vector definition declared in category *ITEM_STRUCTURE_LIST*. The data item *_item_structure.code* is a child of the item *_item_structure_list.code*. The item *_item_structure.code* provides an indirect reference into the list of structure-type definitions in category *ITEM_STRUCTURE_LIST*. The *_item_structure.organization* item describes the row/column precedence of the matrix organization.

2.6.6.1.13. *ITEM_STRUCTURE_LIST*

The *ITEM_STRUCTURE_LIST* category holds definitions of matrices and vectors that can be associated with data items. A component of the key for this category is *_item_type_list.code*, which is referenced by *_item_structure.code* to assign a structure type to a data item. The definition of a structure involves the specification of a length for each dimension of the matrix structure. The combination of items *_item_structure_list.code* and *_item_structure_list.index* forms the key for this category. The latter index item is the identifier for the dimension, hence multiple unique dimensions can be specified for each structure code. The length of each dimension is assigned to *_item_structure_list.dimension*.

2.6.6.1.14. *ITEM_SUB_CATEGORY*

The *ITEM_SUB_CATEGORY* category is used to assign subcategory membership for data items. A data item may belong to any number of subcategories. Each subcategory must be defined in a category named *SUB_CATEGORY*.

2.6.6.1.15. *ITEM_TYPE*

The *ITEM_TYPE* category holds a code that identifies the data type of each data item. The data item *_item_type.code* is a child of the item *_item_type_list.code*. Data-type definitions are actually made in the *ITEM_TYPE_LIST* parent category. The item *_item_type.code* provides an indirect reference into the list of data-type definitions in category *ITEM_TYPE_LIST*. This indirect

reference is provided as a convenience to avoid the redeclaration of the full data-type specification for each data item. The key item for this category is `_item_type.name`, which is defined in the parent category `ITEM`. Only one data type may be specified for a data item.

2.6.6.1.16. `ITEM_TYPE_CONDITIONS`

The category `ITEM_TYPE_CONDITIONS` defines special conditions applied to a data-item type. This category has been included in order to comply with previous applications of STAR and CIF. Since the constructions that are embodied in this category are antithetical to the data model that underlies DDL2, it is recommended that this category only be used for the purpose of parsing existing data files and dictionaries.

2.6.6.1.17. `ITEM_TYPE_LIST`

The `ITEM_TYPE_LIST` category holds the list of item data-type definitions. The key item in this category is `_item_type_list.code`. Data types are associated with data items by references to this key from the `ITEM_TYPE` category. One of the data-type codes defined in this category must be assigned to each data item.

The definition of a data type consists of the specification of the item's primitive type and a regular expression that defines the pattern that must be matched by any occurrence of the item. The primitive type code, `_item_type_list.primitive_code`, can assume values of `char`, `uchar`, `numb` and `null`. This code is provided for backward compatibility with STAR and CIF applications that employ loose data typing. The data item `_item_type_list.construct` holds the regular expression that must be matched by the data type. Simple regular expressions can be used to define character fields of restricted width, floating-point and integer formats.

Molecular Information File (MIF) applications (Allen *et al.*, 1995) have extended the notion of the regular expression to include data-item components. This permits the construction of complex data items from one or more component data items using regular expression algebra. These extended regular expressions are defined in the category `ITEM_TYPE_CONDITIONS`.

Example 2.6.6.1 illustrates the data types that are defined within this DDL. The DDL uses a number of character data types which have subtly different definitions. For instance, the data type identified as `code` defines a single-word character string; `char` extends the `code` type with the addition of a white-space character; and `text` extends the `char` type with the addition of a newline character. Two special character data types `name` and `idname` are used to define the full STAR data name and the STAR name components, respectively. The data type `any` is used to match any potential data type. This type is used for data items that may hold a variety of data types. The data type `int` is defined as one or more decimal digits and the `yyyy-mm-dd` type defines a date string.

2.6.6.1.18. `ITEM_UNITS`

The `ITEM_UNITS` category holds a code that identifies the system of units in which a data item is expressed. The data item `_item_units.code` is a child of the item `_item_units_list.code`. Unit definitions are actually made in the `ITEM_UNITS_LIST` parent category. The item `_item_units.code` provides an indirect reference into the list of data-type definitions in category `ITEM_UNITS_LIST`. This indirect reference is provided as a convenience to avoid the redeclaration of the full data-type specification for each data item. The key item for this category is

Example 2.6.6.1. *The description of permitted data types in the DDL2 dictionary.*

```
#          DATA TYPE CONVERSION TABLE
#          -----
#
loop
  _item_type_list.code
  _item_type_list.primitive_code
  _item_type_list.detail
  _item_type_list.construct

code char 'A single word'           '[^\t\n "]*'
char char 'A single line of text'   '[^\n]*'
text char 'Text which may span lines' '.'*'
int  numb 'Unsigned integer data'   '[0-9]+'
name uchar 'A data item name'
      '[_A-Za-z0-9]+[.][ ][_A-Za-z0-9%/-]+'

idname uchar
      'A data item name component or identifier'
      '[_A-Za-z0-9]+'
any char 'Any data type'           '.'*'
yyyy-mm-dd
      char 'A date format'
      '[0-9][0-9][0-9][0-9]-[0-9]?[0-9]-[0-9][0-9]'
```

`_item_units.name`, which is defined in the parent category `ITEM`. Only one type of unit may be specified for a data item.

2.6.6.1.19. `ITEM_UNITS_CONVERSION`

The `ITEM_UNITS_CONVERSION` category holds a table of conversion factors between the systems of units described in the `ITEM_UNITS_LIST` category. The systems of units are identified by a `*.from_code` and a `*.to_code`, which are both children of the item `_item_units_list.code`. The conversion is defined in terms of an arithmetic operator and a conversion factor, `_item_units_conversion.operator` and `_item_units_conversion.factor`, respectively.

2.6.6.1.20. `ITEM_UNITS_LIST`

The `ITEM_UNITS_LIST` category holds the descriptions of systems of physical units. The key item in this category is `_item_units_list.code`. Units are assigned to data items by references to this key from the `ITEM_UNITS` category.

2.6.6.2. DDL2 definitions describing categories

In this section, the DDL definitions that describe the properties of categories, category groups and subcategories are presented. Fig. 2.6.4.2 illustrates the organization of these categories.

2.6.6.2.1. `CATEGORY`

The category named `CATEGORY` contains the data items that describe the properties of collections of related data items. A DDL category is essentially a table. In this category the characteristics of the table as a whole are defined. This category includes the data items `_category.id` to identify a category name; `_category.description` to describe a category; `_category.mandatory_code` to indicate whether the category must appear in a data block; and `_category.implicit_key`, which can be used to merge like categories between data blocks. The category identifier `_category.id` is a component of the key in most of the DDL categories in this section. The parent definition of the category identifier and all its child relationships are defined in this category.