2.2.4.3. Line lengths

The STAR File does not restrict the lengths of text lines. The original CIF specification introduced an 80-character limit to facilitate programming in Fortran and transmission of CIFs by email. Contemporary practices have enabled the line-length limit in the version 1.1 specification to be extended to 2048 characters. While the new limit in effect mandates the use of a 2048-character input buffer in compliant CIF-reading software, there is no obligation on CIF writers to generate output lines of this length.

2.2.4.4. Lengths of data names and block codes

CIF imposes another length restriction that is not integral to the STAR File syntax: data names, data-block codes and frame codes may not exceed 75 characters in length. This is an increase over the 32-character limit of the original specification, although this extension had already been approved by COMCIFS to coincide with the release of version 2 of the core dictionary (see Chapter 3.2).

There is no fundamental technical reason underlying this restriction; it permits assignment of a fixed-length buffer for recording such data names within a software application, but perhaps more significantly, it encourages a measure of conciseness in the creation of data names based on hierarchical component terms.

2.2.4.5. Case sensitivity

Following the general STAR File approach, the special tokens `loop_`, `save_`, the reserved word `global_`, data-block codes and save-frame codes, and data names are all case-insensitive. The case of any characters within data values must, however, be respected.

2.2.5. Common semantic features

As mentioned in Section 2.2.1, the STAR File structure allows retrieval of indexed data values without prior knowledge of the location of a data item within the file. Consequently there is no significance to the order of data items within a data block. However, molecular-structure applications will typically need to do more than retrieve an arbitrary string value. There is a need to identify the nature of individual data items (achieved portably through the definitions of standard data names in dictionary files), but also to be able to process the extracted data according to whether it is numerical or textual in nature, and possibly also to parse and extract more granular information from the entire data field that has been retrieved.

Different CIF applications have a measure of freedom to define many of the details of the content of data fields and the ways in which they may be processed – in effect, to define their semantic content. However, there are a number of conventions that are common to all CIF applications and this should be recognized in software applicable to a range of dictionaries.

These are discussed in some detail in the formal specification document in Section 2.2.7.4; in this section some introductory comments and additional explanations are given.

2.2.5.1. Data-name semantics

It is a fundamental principle of the STAR File approach that a data name is simply an arbitrary string acting as an index to a required value or set of values. It is equally legitimate to store the value of a crystal cell volume either as

```
_bdouiGFG78=z  1085.3(3)
```

or as

```
_cell_volume   1085.3(3)
```

provided that the users of the file have some way of discovering that the cell volume is indeed indexed by the tag `_bdouiGFG78=z` or `_cell_volume`, as appropriate.

However, it is conventional in CIF applications to define (in public dictionary files) data names that imply by their construction the meaning of the data that they index. Chapter 3.1 discusses the principles that are recommended for constructing data names and defining them in public dictionaries, and for utilizing private data names that will not conflict with those in the public domain.

Careful construction of data names according to the principles of Chapter 3.1 results in a text file that is intelligible to a scientist browsing it in a text editor without access to the associated dictionary definition files. In many ways this is useful; it allows the CIF to be viewed and understood without specialized software tools, and it safeguards some understanding of the content if the associated dictionaries cannot be found. On the other hand, there is a danger that well intentioned users may gratuitously invent data names that are similar to those in public use. It is therefore important for determining the correct semantic content of the values tagged by individual data names to make maximum possible disciplined use of the registry of public dictionaries, the registry of private data-name prefixes, and the facilities for constructing and disseminating private dictionaries discussed in Chapter 3.1.

2.2.5.2. Data typing

In the STAR File grammar, all data values are represented as character strings. CIF applications may define data types, and in the macromolecular (mmCIF) dictionary (see Chapter 3.6) a range of types has been assigned corresponding to certain contemporary computer data-storage practices (*e.g.* single characters, case-insensitive single characters, integers, floating-point numbers and even dates). This dynamic type assignment is supported by the relational dictionary definition language (DDL2; see Chapter 2.6) used for the mmCIF dictionary and is not available for all CIF applications.

However, a more restricted set of four primary or base data types is common to all CIF applications.

The type **numb** encompasses all data values that are interpretable as numeric values. It includes without distinction integers and non-integer reals, and the values may be expressed if desired in scientific notation. At this revision of the specification it does not include imaginary numbers. All numeric representations are understood to be in the number base 10.

It is, however, a complex type in that the standard uncertainty in a measured physical value may be carried along as part of the value. This is denoted by a trailing integer in parentheses, representing the integer multiple of the uncertainty in the last place of decimals in the numeric representation. That is, a value of ‘1085.3(3)’ corresponds to a measurement of 1085.3 with a standard uncertainty of 0.3. Likewise, the value 34.5(12) indicates a standard uncertainty of 1.2 in the measured value.

Care should be taken in the placement of the parentheses when a number is expressed in scientific notation. The second example above may also be presented as 3.45E1(12); that is, the standard uncertainty is applied to the mantissa and not the exponent of the value.

Note that existing DDL2 applications itemize standard uncertainties as separate data items. Nevertheless, since the DDL2 dictionary includes the attribute `_item_type_conditions.code` with an allowed value of ‘esd’, future conformant DDL2 parsers might be expected to handle the parenthesized standard uncertainty representation.