Centre de Données astronomiques

de Strasbourg

Astronomical Catalogues and Tables Adopted Standards

Version 2.0
February 2000

Prepared by François Ochsenbein
Centre de Données astronomiques de Strasbourg (CDS)

Document accessible at
http://vizier.u-strasbg.fr/doc/catstd.htx

Contents

List of Tables

List of Figures

Chapter 1

The Question of Standards

1.1 Introduction

The flow of incoming new astronomical catalogues and data sets is ever increasing; and since 1993, large tables published in some astronomical journals are also available as electronic files, due to the agreement with the A&A Editors[1], and to the development of the electronic publishing for the AAS journals.

The question of a uniform description is raised as soon as we expect to use any of the catalogues in a transparent way, e.g. to ingest the data into a data-base, to load a table of a published paper into our image processing system, etc...

1.2 FITS: why

FITS format *(Flexible Image Transport Format, see e.g. the FITS Support Office Home Page*[2]*)* is apparently the most popular data format for the exchange of computer data in our discipline, and is endorsed by IAU. FITS for coding images is widely available (e.g. in the popular *xv* image displayer), and the FITS way of specifying the position and the geometry of the field of view on the sky (the *World Coordinate System* or *WDS*) allows to combine images of the sky and data coming from catalogues in tools like Aladin. *FITS* defines also a format for tabular data.

1.3 FITS: why not

FITS is well accepted as a *transport* format; it apparently is also being commonly used for *data archiving* (e.g. ESO, OGIP) of image or spectral data. However, as far as tabular data is concerned, we prefer not to store the data in any of the two *FITS* formats defined for tabular data, mainly because

- the data stored at CDS are data *published* in the astronomical literature, and we want to be able to compare by eye the computer files with the published material; the *FITS ascii table* files have no carriage-return, and such tables are therefore not immediately printable.

[1] See Editorial note of A&A **266**, E1
[2] http://fits.gsfc.nasa.gov/

- The description in FITS of the table structure is well adapted for computer usage; however, an overview of the contents of a table — which parameters, and their meaning — is not pleasant to read by a human eye;
- due to the fixed-length record structure, FITS files are generally larger than standard ascii files, even for *FITS binary tables*;
- due to this structure of bulk files, no standard Unix tool like *grep*, *sort*, *join*, *paste* and other *awk* utility which contribute to make Unix so powerful can be applied to FITS files — even an editor cannot be used.

1.4 Proposed solution

We do not want to keep several versions of the same file: not only it would eat up a lot of disk space, but mainly the existence of astronomical catalogues in several data format standards would inevitably generate conflicts and incompatibilities.

In order to be able to use the various tools, we therefore chose to keep the astronomical catalogues as *plain ascii files*, and to store the description of the catalogue as a separate ascii file. This *description file* includes all the necessary information related to the catalogue: author(s), reference(s) of the related published papers, brief summary, scientific keys, caption and accurate description of each table of the catalogue.

This *description file* — the key of the catalogue — is especially important: it must be easily readable by a human eye, and simultaneously contain all the information required to achieve the conversion into other formats if required: for instance, the transformation of tabular data files into FITS can be done automatically from the **CDS FTP server**[3]; and a client/server prototype, described in the *cdsclient package*[4], has been developed for remote copies of the astronomical catalogues in a few formats, including FITS. This *description file* is also used for checking purposes: a standalone program named anafile (see section 5), is used in this context.

The standard discussed in this document mainly addresses the tabular data, but some catalogs may include data in a non-tabular form, like maps, images, video sequences, etc... Such non-tabular data should be described in the *description file* on the file level.

The standards proposed here are developed in the following chapters. Some points in the proposed standards are not completely fixed and possible alternatives are indicated by a note.

1.5 History of this document

The first version of this document was published as the paper *Adopted standards for catalogues at CDS* published in the *Bulletin d'Information du Centre de Données astronomiques de Strasbourg (BICDS)*, (Ochsenbein F., 1994, BICDS **44**, 19).

Version 1.4 of the document was dated 12 September 1994, and resulted from discussion with ADC colleagues, mainly N. Paul Kuin. Common conventions concerned the standard filename conventions — `ReadMe` as the description file, extensions `.dat` for the data files — and the basic label definitions.

[3]http://vizier.u-stasbg.fr/doc/ftp.htx
[4]http://vizier.u-stasbg.fr/doc/catstd-4.htx

Version 1.5 is dated 12 June 1996, and contained a few more conventions on file names and label definitions.

Version 2.0 is dated February 2000, and resulted from discussions with G. Schwarz (AAS, Tucson)

1.6 Related Documents

The following related documents are available on the network:

- `ftp://cdsweb.u-strasbg.fr/pub/cats/doc.ps`: this document, in Postscript format
- `http://cdsweb.u-strasbg.fr/doc/submit.htx`: short overview and tips for preparing a standardized catalogue
- `http://cdsweb.u-strasbg.fr/Cats.html`: list of catalogues and tables available at CDS
- `http://vizier.u-strasbg.fr/`: home page of the VizieR service, with links to the existing clones. VizieR is a direct application of the usage of the standardized description of catalogues for the ingestion of astronomical data into a database.
- `http://vizier.u-strasbg.fr/doc/astrores.htx`: a possibility of getting the results in XML, an emerging standard which basically allows to have the data and their meta-data (accurate description of the data) in a single document; XML allows to display the data, and offers a possibility of using the data in new applications.

Chapter 2

The Catalogue Files & Directories

2.1 The Directory Tree

Each catalogue available at CDS is made of several files stored in a *directory* of a Unix-like file system.

The directory tree naming conventions exactly follow the standards adopted at CDS in the mid 70's: astronomical catalogues have been assigned a chronological number in *categories* numbered I to IX (see Table 2.1) reflecting the main scientific interest of the catalogue; this numbering system is shared by the CDS and the participating Data centers, mainly NSSDC-ADC[1] (Astronomical Data Center at NASA Space Science Data Center).

The explosion of incoming catalogues from the beginning of 1993 due to the electronic publication (see Chapter 1) lead us to introduce the **J** category: within this category, the catalogue designation maps the reference of the published paper, e.g. `J/A+AS/97/729` for the article published in A&A *Suppl.* **97**, page 729.

Within this new **J** section, there is therefore no need for an agreement for the numbering of catalogues between data centers; finding out where a catalogue is stored, knowing its reference, is straightforward. But catalogues do not have to stay in this **J** section for ever: later, more "consistent" catalogues could be generated from one or several publications — typically a catalogue is created which is a merging of the results published as a set of several papers.

2.2 File Naming Conventions

All files making up the catalogue or publication are stored in a directory named according to the conventions described above. A *description file*, which contains the required information needed to understand the origin and the contents of a catalogue, is named `ReadMe`. The contents of this important file is described in chapter 3.

A file named `=obsolete=`, if existing, means that the catalogue is obsolete — typically is an outdated version. The contents of this file indicates which catalogue can be used instead of the obsolete version.

Besides these 2 "special" files — `ReadMe` always present and `=obsolete=` existing only for outdated catalogues — the *data files* are named according to the following rules:

[1]http://adc.gsfc.nasa.gov

Table 2.1: Directory Tree of Catalogues at CDS

I/*number*	Astrometric Catalogues
II/*number*	Photometric Catalogues (except Radio)
III/*number*	Spectroscopic Catalogues
IV/*number*	Cross-Identifications
V/*number*	Combined Data
VI/*number*	Miscellaneous Catalogues
VII/*number*	Non-stellar Objects
VIII/*number*	Radio Catalogues
IX/*number*	High Energy Catalogues
J/*abbr*/*Volume*/*first_page*	Publications ordered by Journals, with *abbr*:
	A+A = A&A
	A+AS = A&A *Suppl.*
	AJ = *Astron. J.*
	ApJ = *Astrophys. J.*
	ApJS = *Astrophys. J., Suppl.*
	MNRAS = *Mon. Not. R. Astron. Soc.*
	PASP = *Publ. Astron. Soc. Pacific*
	AZh = *Astron. Zhurnal* (Russia)
	PAZh = *Pis'ma Astron. Zhurnal* (Russia)
	other = Form **J/other**/*abbr*/*Volume.first_page*
	= for other journals, *abbr* being written as the bibcode[a]

[a] http://cdsweb.u-strasbg.fr/simbad/refcode.html

1. filenames should be compatible with MS-DOS limitations: filename is written *name.extension*, with at most 8 characters for *name* and 3 characters for *extension*; only alphanumeric characters, plus the minus sign and the underscore, are allowed; and case is not significant — filenames are normally displayed in *lowercase letters only.*

2. for files corresponding to published material, the names are consistent with the published paper, and we use `table`*n.extension* to refer to the table numbered n in the published paper, `fig`*n.extension* for the figure numbered n, etc.

3. if the rule above cannot be applied, we use mnemonic names like `main`, `catalog` or `data` for the main part of the catalogue, `refs` for the references, `notes` for the notes, etc...

4. the *extension* is related to the *format* of the file, with the following conventions:

   .csv for files containing tabular data as *character-separated values* i.e. columns separated by a special character, generally the semi-colon **;** (see also the **.tsv** extension)

   .dat for files containing the data in plain ascii form. The exact structure of such files — the column layout — is normally described in the `ReadMe` file.

   .fit for *FITS* files

   .fih for *FITS headers*, *i.e.* the top part of *FITS* files containing the keywords with embedded newlines.

   `.gif` for data files containing images in *GIF* format

   `.jpg` for data files containing images in *JPEG* format

- `.mpg` for data files containing video sequences in *MPEG* format
- **.ori** for the original files, when modifications had to be performed and the original files have to be available
- **.pdf** for Adobe's *PDF* format
- **.ps** for *PostScript* files
- **.sam** for *samples* files, when the whole catalogue can't be stored in the FTP directories. The total number of records is then indicated as the first number in the explanation. See e.g. the USNO Catalogue (I/252[2])
- **.sty** for *style files* related to TeX or LaTeX definitions.
- **.tar** for files in *Tape ARchive* format (Unix), allowing many files to be archived as a single file.
- **.tgz** for *Gnu-zipped Tape ARchive* format (Unix), a short-hand of the `.tar.gz` suffix.
- **.tex** for files in plain TeX or in LaTeX.
- **.txt** for files containing text in plain ascii form.
- **.tsv** for files containing tabular data as *tab-separated values*, i.e. columns separated by the *TAB* character (see also the **.csv** extension).

Files may also be *Unix–compressed* or *Gnu-zip compressed*; a `.Z` suffix is appended to the filenames described above in case of Unix compression (the *uncompress* Unix program has to be used), and a `.gz` or `.z` in case of *gzip* compression (the *gunzip* public-domain program has to be applied).

Large files may also be cut into pieces, generally not larger than 10 Megabytes. In this case, a *numeric* suffix of 2 or 3 digits can be added; an example can be found for the Tycho-2 Catalogue[3], where the data file was split into 20 parts named `tyc2.dat.00`, `tyc2.dat.01`, ⋯ `tyc2.dat.19`.

2.3 Catalogue Subdirectories

It may happen that some catalogues contain a large number of files, as in Catalogue III/166[4] which contains about 80 stellar spectra corresponding to some standard spectral types. These data files made of just 2-column tables were saved in a subdirectory named `sp`, and the characteristics of each of these 80 files containing spectra are summarized in a table named `spectra.dat` which is described in the `ReadMe` file. In other words, it is possible to describe files with a level of indirection, as a table which details characteristics of files stored in one or several subdirectories.

2.4 Data files

Data files in principle contain only the data, without titles, headers, commments, etc. However introductory comments stored at the beginning of the data files being handy, a possibility of specifying this feature has been added in the Byte-by-byte Description[5]. Two possible ways exist for introductory comments in data files:

[2] http://vizier.u-strasbg.fr/cgi-bin/Cat?I/252
[3] ftp://cdsarc.u-strasbg.fr/pub/cats/I/259
[4] http://vizier.u-strasbg.fr/cgi-bin/Cat?III/166
[5] catstd-3.1.htx#ByteByByte

1. by specifying a *number of introductory lines*, e.g. the first 20 lines are comments.
2. by specifying a *character used for introductory comments*, e.g. the first lines having a # as their leftmost character represent introductory comments.

Data files may also contain *empty lines* – empty lines are ignored wherever they are in the file.

2.5 Index Files

A set of files summarizing the catalogues currently available at CDS is updated regularly (normally on a weekly basis):

- `cats.all`[6]: lists all catalogues (flat ascii)
- `cats.lis`[7]: provides only basic information about each catalogue
- `cats.tex`[8]: is the LaTeX version used for publication in the *Bulletin d'Information du CDS*
- `cats.dvi`[9]: is the *dvi* translation of `cats.tex` which can used for remote display *e.g.* via *XMosaic*
- `cats.new`[10]: contains the same information as `cats.all`, for catalogues acquired during the last month;

Note that a facility exists to query this index remotely: the *findcat* program, which is a part of the *cdsclient* package, described in the *cdsclient* package, described in Chapter 4.

[6] ftp://cdsarc.u-strasbg.fr/pub/cats/cats.all
[7] ftp://cdsarc.u-strasbg.fr/pub/cats/cats.lis
[8] ftp://cdsarc.u-strasbg.fr/pub/cats/cats.tex
[9] ftp://cdsarc.u-strasbg.fr/pub/cats/cats.dvi
[10] ftp://cdsarc.u-strasbg.fr/pub/cats/cats.new

Chapter 3

`ReadMe` file Contents

This chapter details all items of the `ReadMe`. For just an overview of the contents and role of this file, and some tips on how to fill it, refer to the document *Submitting Data in Electronic Form*[1].

3.1 Structure of the `ReadMe` File

The *Description File*, named `ReadMe`, is aimed to provide all necessary information to locate the catalogue (authors, title, references, summary, etc...) and to interpret its contents by automatic procedures.

An example of the `ReadMe` file of the catalogue I/221 is given in the Figure 3.1. The *description file* contains severals sections; as a general rule, **only** *section headers* **are left flushed, while the text is indented** — with the noticeable exceptions of the *title*, the file names in the `File Summary` section, and of *Note headers* (section 3.5). No line in this *description file* can exceed 80 characters; it is moreover suggested to limit the textual parts to *70 characters*, such that a conversion to FITS could keep the text as `COMMENT` cards.

The *description file* contains the following parts:

1. First line: catalogue designation, an abbreviated title followed within parenthesis by the last name of the first author, a + sign if there are multiple authors, and the year — this information has to be condensed in a single line of 80 characters or less;

2. Full title(s), authors, and reference(s) of the catalogue. Each title is left-adjusted (no indentation); the line(s) containing the authors' names are indented (at least two blanks), and the bibliographic reference is enclosed between angle brackets. The standard SIMBAD/NED/ADS 19-byte bibliographical reference code(s) named *BibCode*[2] is introduced by an equal sign, as a word without embedded blank of exactly 20 characters (with the equal sign).

3. The Keywords.

   There are three categories of keywords:

   - `Keywords:` introduces the list of keywords as in the printed publication
   - `ADC_Keywords` introduces the list of data-related keywords, out of a controlled set[3] (see

[1] http://vizier.u-strasbg.fr/doc/submit.htx

[2] see *e.g.* a description in the *SIMBAD User's Guide*, page 65, or on the WWW page http://cdsweb.u-strasbg.fr/simbad/refcode.html

[3] ADCkwds.htx

```
I/221              The Magellanic Catalogue of Stars - MACS (Tucholke+ 1996)
================================================================================
The Magellanic Catalogue of Stars - MACS
     Tucholke H.-J., de Boer K.S., Seitter W.C.
    <Astron. Astrophys. Suppl. Ser., 119, 91-98 (1996)>
    <The Messenger 81, 20 (1995)>
    =1996A&AS..119...91T
    =1995Msngr..81...20D
================================================================================
ADC_Keywords: Magellanic Clouds ; Positional data

Description:
    The Magellanic Catalogue of Stars (MACS) is based on scans of ESO
    Schmidt plates and contains about 244,000 stars covering large areas
    around the LMC and the SMC. The limiting magnitude is B<16.5m and the
    positional accuracy is better than 0.5" for 99% of the stars. The
    stars of this catalogue were screened interactively to ascertain that
    they are undisturbed by close neighbours.


File Summary:
--------------------------------------------------------------------------------
 FileName    Lrecl    Records    Explanations
--------------------------------------------------------------------------------
ReadMe          80          .    This file
lmc.dat         52     175779    The Large Magellanic Cloud
smc.dat         52      67782    The Small Magellanic Cloud
--------------------------------------------------------------------------------

Byte-by-byte Description of file: lmc.dat smc.dat
--------------------------------------------------------------------------------
   Bytes Format  Units   Label    Explanations
--------------------------------------------------------------------------------
   1- 12  A12    ---     MACS     Designation
  14- 15  I2     h       RAh      Right Ascension J2000 , Epoch 1989.0 (hours)
  17- 18  I2     min     RAm      Right Ascension J2000 (minutes)
  20- 25  F6.3   s       RAs      Right Ascension J2000 (seconds)
      27  A1     ---     DE-      Declination J2000 (sign)
  28- 29  I2     deg     DEd      Declination J2000 , Epoch 1989.0 (degrees)
  31- 32  I2     arcmin  DEm      Declination J2000 (minutes)
  34- 38  F5.2   arcsec  DEs      Declination J2000 (seconds)
      40  I1     ---     Npos     Number of positions used
  42- 46  F5.2   mag     Mag      ?=99.00 Instrumental Magnitude
                                        (to be used only in a relative sense)
      48  I1     ---     PosFlag  [0,1] Position Flag   (0: ok,
                                        1: internal error larger than 0.5")
      50  I1     ---     MagFlag  [0,1] Megnitude Flag  (0: ok,
                                        1: bad photometry or possible variable)
      52  I1     ---  BochumFlag *[0] Bochum Flag
--------------------------------------------------------------------------------
Note on BochumFlag: 1 if in Bochum catalog of astrophysical information
    on bright LMC stars) (yet empty)
--------------------------------------------------------------------------------

Author's address:
    Hans-Joachim Tucholke    <tucholke@astro.uni-bonn.de>

================================================================================
(End)            Hans-Joachim Tucholke [Univ. Bonn]                   20-Nov-1995
```

Figure 3.1: The ReadMe file of a catalogue

also examples at ADC[4]). Unlike the `Keywords:` set which is generally related to the scientific goal of a paper, the `ADC_Keywords` are stricly related to the tabular material collected in the paper.

- `Mission_Name:` for data originated from satellite mission, this header precedes the satellite name.

Whatever the category, multiple keywords are separated by a semicolon (`;`) or a dash (`-`) embedded in blanks.

4. A short description of the contents, the purpose and remarks of special importance of the catalogue introduced by `Description:` and/or `Abstract:` section headers;

   This description contains a plain ascii text; special symbols are written with the the following conventions shared with data-base:

   - exponents are within carrets (`^`) signs, e.g. $x^2 + y^2$ is written `x^2^+y^2`, and indices are within underscores _ like H_2O written `H_2_O`
   - special symbols can be written as text within braces, like {`delta`} for δ, {`Delta`} for Δ, {`+/-`} for $\pm$, {`prop.to`} for $\propto$, ... (i.e. mostly standard LaTeX type aliases)

5. *(optional)* The list of *observed objets* introduced by `Objects:` only in the case where no data table contains the list and position of the astronomical objects observed or studied, as for example in the study of a high-resolution spectrum of a single star. Such a list is normally restricted to very few objects – less than 10 or 20 typically; when the list of objects is large, it is suggested to store the list of objects with their positions in a dedicated file named `objects.dat` or stars.dat described in a standard way.

   The structure for this list must follow the following template:

```
Objects:
    --------------------------------------------------
      RA  (equinox)  DE     Name(s)
    --------------------------------------------------
    hh mm ss.s +dd mm ss    Name1 = Name1
    --------------------------------------------------
```

   The header line (the one with `RA`, `DE`, and the equinox) should be aligned with the data, in order to give a measure of the starting byte of the object designations (names).

   When each object in this list is related to a file, the name of this file can be put within brackets at the end of each line, as for instance in catalogue J/MNRAS/301/1031[5].

6. A list of the files making up the catalogue is introduced by the `File Summary:` section header. This list includes the following basic information for each file: its *name*, its *record length* (length of the longest line), the *number of records*, and a short *title* (caption).

7. *(optional)* The list of related catalogues, data sets or services are introduced by the `See also:` header. In this section, each catalog or service starts on a new line, and is followed by a *colon* embedded in blanks, *e.g.* :

[4] http://adc.gsfc.nasa.gov/adc/adc_keyword_index.html
[5] http://vizier.u-strasbg.fr/cgi-bin/Cat?J/MNRAS/301/1031

```
See also:
    J/A+AS/97/729 : O-rich stars in 1-20um range
    http://machine/description.html : Detailed Description
```

8. *(optional)* Due to the frequent difficulties encountered the the *nomenclature of astronomical objects*, a dedicated section introduced by the `Nomenclature Notes:` header provides the necessary explanations in the peculiar conventions used in this matter.

9. A description explaining which are the columns of the tables, how to get the values stored in these columns, and what is their meaning is introduced by the `Byte-by-byte Description of file:` section header. This line may specify that data files include *header lines* which are not part of the data (see section 2.4[6]); the existence of header lines can be specified either by a number (i.e. the first n lines of the files are not data), or by a character which introduces a comment (usually a hash `#`). The existence of header lines is specified in parentheses as e.g. `Byte-by-byte Description (12 headlines):` or when header lines are introduced by a has sign:
   `Byte-by-byte Description (# headlines):`

   The description is presented as a five-column table with the following elements:

   (a) the *starting* (from 1) and *ending* byte of a column, separated by a dash `-`; this dash is however not required for a single-byte column .
   (b) *à la FITS* format which specifies how to interpret numbers or symbols, composed of
      - a letter **A, I, F** or **E** indicating to intrepret the data as **A**scii text, **I**nteger number, **F**loating-point number with a fixed number of decimals, or a floating-point number written in **E**xponential notation
      - followed by a *number* indicating the width of the column in bytes,
      - eventually followed by a dot and a number indicating the number of decimal digits (for **F** and **E** notations)

      This format could be preceded by an *iteration factor* to designate an *array* of values all written with the same format.
   (c) the *Unit* in which the value is expressed; unit standards are detailed below (section 3.2). The symbol `---` indicates unitless values, and the square brackets [*unit*] indicates values tabulated as *decimal logarithmic values.*
   (d) a *label* or column header. Standard names and name building rules are detailed in section 3.3.
   (e) a short explanation of the contents of the column. This last field may also specify:
      - a set of valid characters for an alphabetical column, or limits for numeric columns: see section 3.4.
      - whether the column is ordered: see section 3.4
      - whether blank (unspecified) numbers are allowed: see section 3.4
      - a key to more detailed notes: see section 3.4

10. *(optional)* Global notes — notes which apply to several tables — are introduced by `Note (G`n`):` n being the number of the global note referenced in the `Byte-by-byte Description of file:` sections.

[6] catstd-2.htx#datafiles

11. *(optional)* some other sections may exist when required, e.g. `History;` introduces notes about the modification history, `Acknowledgements:` etc...

12. *(optional)* The list of references is introduced by the `References:` header; the 19-character *BibCode* is used when possible, to enable an automatic link to the existing Abstract Services like ADS.

13. the very last line includes just the left-flushed word `(End)`, the name of the person who took care of the standardisation, and the *date* of the last modification.

3.2 Units

The unit in which physical parameters are expressed is a fundamental parameter which becomes especially important when data are to be compared, or used outside a specialized field of science — most physicists have never heard of astronomy-specific units.

Special care has been taken to try to use the standard SI units, and to convert if necessary the unit to such standards: for instance, we use the string `0.1nm` to express Angströms (Å$= 10^{-10}m$), since the Angström is a non-standard unit. Another example is `mW/m2` , the milliwatt per m^2, which is identical to the CGS erg/cm^2/s unit, which is not used outside our discipline.

Only simple power functions of physical units are accepted, which means that e.g. `solMass`$^{3/2}$ (solar mass at a $\frac{3}{2}$ power) cannot be a valid unit. Some values are however traditionnally written in a decimal logarithmic scale, and we introduced **bracketed units** like `[solMass]` to indicate $\log M_\odot$ unit, i.e. a value representing solar masses expressed on a logarithmic scale.

The standard adopted here differs from the OGIP ones[7] for the syntax of composite units (operator symbols), and in the usage of math functions (only the square bracketted units, representing the log function is accepted here) and of obsolete CGS units which are not avoided here; the basic symbols however agree.

3.2.1 Syntax of Units

The syntax of the unit expression is summarized by the following rules:

Rule 1: any unit is described by a *single word* — no space is allowed. For instance, the Angström is coded `0.1nm`, and never `0.1 nm`; the kilometer-per-second is coded `km/s`, or `km.s-1`, but never `km / s` or `km s-1`.

Rule 2: the only allowed numerical factor is at the very beginning of the Unit string. The structure of the unit is therefore the concatenation of $\boxed{\textit{factor}}$ and $\boxed{\textit{unit\_expression}}$ and we will write the "number of pixels per Å" as `10pix/nm`, and neither `pix/0.1nm` nor `pix/(0.1nm)`.

The numerical *factor* may include the letter `x` for the multiplication, as `1.5x10+11` to express the number 1.5×10^{11}

Rule 3: The operators to express a compound unit are

/ for the division — as in `km/s`

[7] OGIP memo 93-001 about "Specification of Physical Units within OGIP FITS files" by Ian M. George & Lorella Angelini, August 1993

`.` for the multiplication (the dot is however understood as a decimal point in the leading numerical factor) — as in `kW.h`

nothing for a power — as `m2` for m^2 or `10+21` for 10^{21}.

Note that + or − signs are not operators, but represent the leading sign of numeric values.

Rule 4: a simple (non-compound) unit is made of a *basic unit* symbol, eventually preceded by a *multiple prefix.*

Among several possible expressions of a unit, it is preferable to choose the shortest one; this leads also to prefer the division (/) to the multiplication of the inverse, e.g. prefer `km/s` to `km.s-1`.

3.2.2 Basic symbols

The basic symbols listed include basic standard SI units (*b*), the extensions listed by the IAU style book marked (*a*), other frequent physical extensions (*e*), and a few further extensions used at CDS (*c*).

3.2.3 Multiples

The standard SI multiple and submultiple prefixes can be used; these are summarized in Table 3.2. Note that a single prefix can only be attached to a unit symbol, which means that e.g. `mmas` must not be used to designate a μ-arc-second, but rather `uarcsec`.

See the *Unit Calculator*[8]

3.3 Labels

Remember that a *label* is always a single word: it cannot contain any embedded blank or space. Other characters are in principle accepted, e.g. parentheses or other punctuation signs, with the following restrictions:

- the characters `$ @ ! ‘ \ ^ ~ &` are not accepted in a label
- the first character of a label can't be one of the symbols `+ - * / = . # _ ' "`
- the last character of a label can't be one of the symbols `, ;`
- the various kinds of parentheses and brackets `( ) [ ] { }`, when used, must be balanced
- a label can't be made of digits only
- a single-character label must be alphabetic.
- the label `---` is used for filling bytes, like the colon fond sometimes between the hours and minutes, or a `x` existing between the major and minor axises of a galaxy.

[8] http://vizier.u-strasbg.fr/cgi-bin/Unit

Table 3.1: Basic Unit Symbols: (b) are basic SI units, (a) are IAU extensions, (c) are CDS extensions, and (e) are extensions used in physics.

	Symbol	Explanation	Definition
(c)	`---`	Unitless value	
(c)	`%`	Unitless value, in percent	10^{-2}
(a)	`a`	year (also `yr`)	$365.25d = 31.5576 \times 10^6 s$
(b)	`A`	Ampere	
(a)	`AU`	astronomical unit	$1.49598 \times 10^{11} m$
(a)	`arcmin`	minute of arc	$1/60\ ^\circ$
(a)	`arcsec`	second of arc	1/60 arcmin
(e)	`barn`	barn (cross-section)	10^{-28} m^2
(c)	`bit`	binary information unit (computer storage)	
(c)	`byte`	byte (computer storage)	8 bit
	`C`	Coulomb (electric charge)	A·s
(b)	`cd`	Candela (luminous intensity)	
(c)	`ct`	Count (events)	
	`D`	Debye	$(\frac{1}{3}) \times 10^{-29}$ C·m
(a)	`d`	day	$24h = 86.4 \times 10^3 s$
(a)	`deg`	degree of arc (°)	$\pi/180$ rad
(e)	`eV`	electron-Volt	1.602177×10^{-19} J
	`F`	Farad (electric capacitance)	C/V
(b)	`g`	gram	10^{-3} kg
(a)	`h`	hour of time (sideral if appropriate)	3600s
	`H`	Henry (inductance)	Wb/A
	`Hz`	Hertz (frequency)	s^{-1}
	`J`	Joule (energy)	N·m
(a)	`Jy`	Jansky	10^{-26} W/m^2/Hz
(b)	`K`	Kelvin	
	`lm`	lumen (luminous flux)	cd·sr
	`lx`	lux (illuminance)	lm/m^2
(b)	`m`	metre	
(a)	`mag`	magnitudes	
(a)	`mas`	millisecond of arc	$(\pi/6.48) \times 10^{-8}$ rad
(a)	`min`	minute of time (sideral if appropriate)	
(b)	`mol`	mole	
	`N`	Newton (force)	kg·m/s^2
	`Ohm`	(Ω) Ohm (electric resistance)	V/A
	`Pa`	Pascal (pressure)	N/m^2
(a)	`pc`	parsec	$3.0857 \times 10^{16} m$
(c)	`pix`	pixel (image element)	
(b)	`rad`	radian (angle)	
(e)	`Ry`	Rydberg (energy)	$(\frac{1}{2})(\frac{2\pi e^2}{hc})^2 m_e c^2 = 13.60583$ eV
(b)	`s`	second of time	
	`S`	Siemens (electric conductance)	A/V
(c)	`solLum`	Solar luminosity	3.826×10^{26} W
(c)	`solMass`	Solar mass	1.989×10^{30} kg
(c)	`solRad`	Solar radius	6.9599×10^8 m
(c)	`Sun`	Unit referring to the Sun (e.g. abundances)	
	`sr`	steradian (solid angle)	
	`T`	Tesla (magnetic field intensity)	Wb/m^2
	`V`	Volts (electric potential)	W/A
	`W`	Watt (power)	J/s
	`Wb`	Weber (magnetic flux)	V·s
(c)	`yr`	year (also `a`)	$365.25d = 31.5576 \times 10^6 s$

Table 3.2: Symbols used to express multiples and submultiples

Symbol	Explanation	Value
`d`	deci	10^{-1}
`c`	centi	10^{-2}
`m`	milli	10^{-3}
`u`	micro (μ)	10^{-6}
`n`	nano	10^{-9}
`p`	pico	10^{-12}
`f`	femto	10^{-15}
`a`	atto	10^{-18}
`z`	zepto	10^{-21}
`y`	yocto	10^{-24}

Symbol	Explanation	Value
`da`	deca	10
`h`	hecto	10^{2}
`k`	kilo	10^{3}
`M`	mega	10^{6}
`G`	giga	10^{9}
`T`	tera	10^{12}
`P`	peta	10^{15}
`E`	exa	10^{18}
`Z`	zetta	10^{21}
`Y`	yotta	10^{24}

Some commonly used labels are presented in Table 3.3. This table is not an exhaustive list of all possible column labels; it merely represents the standards adopted for the most common columns headings — like celestial coordinates, times and dates, and some of the most frequently found parameters. The VizieR[9] service contains all definitions of all colums; these definitions are stored in the **METAcol**[10] (field meta-data) table which can be queried like any of the tables accessible in VizieR. For instance, finding the columns having *oscillator strength* in their explanation can be found by entering the constraint `*oscillator strength*` (the `*` matches any number of characters) in the *explain* field of the *METAcol*

Some comments on the labels defined in Table 3.3:

- **unicity** principle: two columns can't have the same label in a table (the `---` label however can be present several times). When a table contains for instance the equatorial position in two different systems, B1950 and J2000, choose the labels `RAh` ...`DEs` for the basic (original) position, and choose another set of names, e.g. `RA1950h` ...`DE1950s` for the other set of positions.

- Julian dates should be **complete** – no truncated Julian date, no Modified Julian Date (MJD) which are the source of many confusions.

- The following *suffix*es can be appended to a label to designate the components of time or angle:

  `.Y` Year part (in order to avoid any ambiguity, the year should include all 4 digits)
  `.M` Month name or number in range $[1 \cdots 12]$
  `.D` Day number in month (2 digits, range $[1 \cdots 31]$) or year (3 digits, range $[1 \cdots 366]$)
  `.h` Hour (range $[0 \cdots 24[$) For
  `.m` Minutes (range $[0 \cdots 60[$)
  `.s` Seconds (range $[0 \cdots 60[$)
  `.d` Degrees (range $[0 \cdots 360[$)
  instance, the various components of an observation date and time can be labelled `Obs.Y`, `Obs.Y`, `Obs.M`, `Obs.D`, `Obs.h`, `Obs.m`.

[9]http://vizier.u-strasbg.fr
[10]http://vizier.u-stasbg.fr/cgi-bin/VizieR-3?-source=METAcol

Table 3.3: Conventions used for some *labels*

Symbol	**Explanation**	**Default Limits**
`RAh`	Part of the right ascension expressed in *hours*	$[0, 24[$
`RAm`	Part of the right ascension expressed in *minutes*	$[0, 60[$
`RAs`	Part of the right ascension expressed in *seconds*	$[0, 60[$
`RAdeg`	Right ascension expressed in *decimal degrees* [a]	$[0, 360[$
`RArad`	Right ascension expressed in *radians*	$[0, 2\pi[$
`RAsec`	Right ascension expressed in *seconds*	$[0, 86400[$
`DE-`	*Sign* of declination	`[+-]`
`DEd`	Part of the declination expressed in *degrees*	$[0, 90]$
`DEm`	Part of the declination expressed in *arc minutes*	$[0, 60[$
`DEs`	Part of the declination expressed in *arc seconds*	$[0, 60[$
`DEdeg`	Declination expressed in *decimal degrees* [b]	$[-90, +90]$
`DErad`	Declination expressed in *radians*	$[-\frac{\pi}{2}, +\frac{\pi}{2}]$
`DEsec`	Declination expressed in *arcsec*	$[-324000, +324000]$
`PosErr`	Position mean error	≥ 0
`dRA` `dDE`	Difference in Right Ascension or Declination	
`oRA` `oDE`	Offset from a reference position in R.A. or Dec.	
`ELON`	ecliptic longitude	$[0, 360[$
`ELAT`	ecliptic latitude	$[-90, +90]$
`GLON`	galactic longitude	$[0, 360[$
`GLAT`	galactic latitude	$[-90, +90$
`Epoch`	Epoch — write `Ep-1900` for a year with 1900 offset	
`plx`	Parallax	
`pmRA`	Proper motion in Right Ascension	
`pmDE`	Proper motion in Declination	
`pmX`	Proper motion along X direction	
`Seq`	a Sequential number used e.g. to number the objects	
`Name`	the main name of an object or feature	
`Diam`	Diameter – or `MajDiam` and `MinDiam`	≥ 0
`Axis`	Half Diameter – also `MajAxis` and `MinAxis`	≥ 0
`PA`	Position Angle, normally North to East	$[0, 360[$
`Rad`	Radius	≥ 0
`HRV`	Heliocentric Radial velocity	
`RVel`	Radial velocity	
`Sep`	Separation (anglular or linear)	≥ 0
`SpType`	Spectral type	
`MType`	Morphological type	
`Vmag`	Magnitude (apparent) in V filter	
`VMag`	Absolute Magnitude in V filter	
`Kmag`	Magnitude (apparent) in K filter ...	
`Sp+Index`	Spectral index (α in relation $S(\nu) = \nu^{\alpha}$)	
`Sp-Index`	Spectral index (α in relation $S(\nu) = \nu^{-\alpha}$)	
`BibCode`	The 19-digit bibcode[c]	
`Text`	Free-format text like list of authors, titles...	

[a] ⇒ **`RAhms`** could be envisaged for a right ascension expressed in sexagesimal with no embedded blanks and leading zeroes, **`RAbhms`** for sexagesimal values with embedded blanks. Such extensions would however require special interpretation by FITS readers.

[b] ⇒ **`DEdms`** could be envisaged for a declination expressed in sexagesimal with no embedded blanks and leading zeroes, **`DEbdms`** for sexagesimal values with embedded blanks; questions similar to the **`RAhms`** note have to be addressed.

[c] http://cdsweb.u-strasbg.fr/simbad/refcode.html

- remember that labels are normally written at the top of each column; very long names should be avoided, especially for short (a couple of bytes) columns
- ... but names should however not be too cryptic !
- Columns containing file names should contain `file` or `File` in their labels – e.g. `FITSfile` to indicate an associated image in FITS format

A parameter has frequently associated values, and we have adopted the rule of association with the *one-letter-underscore prefix*: if a column is obviously associated to another one — typically mean errors or uncertainty flags — we use one of the *underscore prefixes* listed in Table 3.4.

Table 3.4: Conventions used for label *prefixes*

Symbol	**Explanation**	**Default Limits**
`a_`*label*	*aperture* used for parameter *label*	≥ 0
`d_`*label*	for a number of degrees of freedom or for number of digits on parameter *label*	> 0
`E_`*label*	mean error (upper limit) on parameter *label*	≥ 0
`e_`*label*	mean error (σ) on parameter *label*	≥ 0
`f_`*label*	*flag* on parameter *label*	
`L_`*label*	*Likelihood* on parameter *label*	
`l_`*label*	*limit flag* on parameter *label*	`[<>]`
`m_`*label*	*multiplicity index* on parameter *label* to resolve ambiguities	
`n_`*label*	*note* (remark) on parameter *label*	
`o_`*label*	number of *observations* on parameter *label*	≥ 0
`q_`*label*	*quality* on parameter *label*	
`r_`*label*	reference (source) for parameter *label*	
`u_`*label*	*uncertainty flag* on parameter *label*	`[ :]`
`w_`*label*	*weight* of parameter *label*	≥ 0
`x_`*label*	unit in which parameter *label* is expressed	

Usual mathematical functions may be specified in the *label*, with parentheses or a dot; for instance, the logarithm of the effective temperature could be labelled `log(Te)` or `log.Te`.

3.4 Data Checking

The first *word* (i.e. set of characters followed by a blank) of the explanation of a column may specify *validity checks* to be performed about:

1. the available *range*[11] of the value in the column
2. the possibility of having unspecified or *NULL*[12] values
3. the *order*[13] of the values within the table (increasing or decreasing order)

[11] See 3.4
[12] See 3.4.1
[13] See 3.4.2

The three different checks have to be specified in this order: range, *NULL*, order, without any embedded blank.

All these checks are performed by the standalone program *anafile* detailed in Chapter 5.

3.4.1 Limits

Limits in a column can be specified in the *explanation* column, with values enclosed within square brackets `[` or `]`; the *square bracket must be the first character* of the explanation if present. The only exception arises when an asterisk is present before the first square bracket to indicate the presence of extended notes (*see Notes*).

1. For an *alphabetic* column (`A`-format), the limits describe the valid *character set*, i.e. the list of valid characters in the column. As an example, an uncertainty column labelled `u_lab` with an `A1` format may only contain a blank, a question mark or a colon (which is different from the default shown in section 3.3); this feature can be specified as follows:

   ```
   3   A1  ---  u_lab   [ :?] Uncertainty flag on parameter lab
   ```

   The dash sign (`-`) may be used to specify consecutive characters, e.g. `[A-F]` for any character of the set $\{A, B, C, D, E, F\}$.

2. For a *numeric* column, the limits can be specified with two numbers separated by a comma or slash and enclosed in square brackets. The inclusion or non-inclusion of the limits as acceptable values follow the standard mathematical conventions, i.e. an opening `]` bracket means that the lower value is excluded, a closing `]` bracket that the upper value is included.

   As an example, the following specifies a parameter $350 < \lambda < 650$:

   ```
   75- 80  F6.2  nm  lambda  ]350,650[ Wavelength
   ```

   Both limits are not required: to express that a value has to be strictly positive, use the expression `]0,]`; the expression `[,0]` specifies a negative or null value. Writing `[]` is acceptable when no range checking applies; this writing is required if a *not-NULL* or a *sorting order* has to be specified.

   When specified, limiting numbers should represent *actual* limits, and not the range of all possible values which can be inferred from the format (e.g. `[-999,9999]` for an `I4` number).

Some labels have implicit limits, listed in the column *Limits* of the tables in section 3.3. These defaults *are overridden* (for *numeric* columns only) by the limits specified within square brackets in the *description file* : writing *e.g.* `[]` as the first word of the explanation of a column labelled `GLON` removes the condition $0 \leq$`GLON`< 360.

3.4.2 *NULL* values

A *NULL* or unspecified value is always indicated by setting all bytes of the column to *blanks*[14]. The range or character set specification may be followed by the characters

[14]⇒ using out-of-range values to specify *NULL*s — typically numbers made of `9`'s only — is required by some FORTRAN users.

- **!** (exclamation mark) indicates that a *NULL* value is forbidden (*i.e.* the column can never be blank)
- **?** (question mark) indicates that a *NULL* value is allowed (*i.e.* the column may be blank). When the **?** is followed by the = sign and a numeric value without intervening space, the value is the alternative NULL value and is equivalent to the FITS TABLES extension TNULL keyword, e.g. `?=99.99`

The default rule is the following:

1. *NULL* value is allowed for an *alphabetic* column (`A`-format);
2. *NULL* value is not allowed for a *numeric* column

3.4.3 Order

Following the range, it is also possible to specify that this column is increasing or decreasing throughout the table. If n is the row number, the conventions are:

$+$	the value in the column is *strictly increasing*,	$val(n+1) > val(n)$
$+=$	the value in the column is *increasing*:	$val(n+1) \geq val(n)$
$-$	the value in the column is *strictly decreasing*:	$val(n+1) < val(n)$
$-=$	the value in the column is *decreasing*:	$val(n+1) \leq val(n)$

3.5 Notes

Lengthy explanations can hardly be inserted in the *Byte-by-byte Description* table; such explanations are easier to read when they are grouped at the end of the description table. Two options are presently proposed:

1. an $*$ at the very beginning of the *short explanation* column indicates the existence of a note; this note is introduced at the end of the table by a

   Note on label: section header. An example of this way is illustrated in the above figure

2. a parenthesed number representing a footnote number as the very last word of the *short explanation* column; the details are introduced at the end of the table by a

   Note (n): section header. An example of this way is illustrated in the above Figure 3.3

3. Finally, **Global Notes** can be used in the case where several tables are assigned the same remark. Such global notes are simply refered as (**G**n) as the last word in the explanations of a column, and the corresponding lengthy note must be introduced by a left-flushed text:
   `Global Notes:` followed by the details of each of the global notes:
   `Note (G1):`
   An example of the usage of global notes may be found for catalog IX/22[15].

3.6 Transformation to FITS

[15]http://vizier.u-strasbg.fr/cgi-bin/Cat?IX/22

```
Byte-by-byte Description of file: table1
--------------------------------------------------------------------------------
   Bytes Format  Units   Label    Explanations
--------------------------------------------------------------------------------
   1-  5  A5     ---     Cluster  Abell cluster designation
   7- 22  A16    ---     Name    *Galaxy name
  24- 25  I2     h       RAh      Right Ascension 1950 (hours)
  27- 28  I2     min     RAm      Right Ascension 1950 (minutes)
  30- 33  F4.1   s       RAs      Right Ascension 1950 (seconds)
      35  A1     -       DE-      Declination 1950 (sign)
  36- 37  I2     deg     DEd      Declination 1950 (degrees)
  39- 40  I2     arcmin  DEm      Declination 1950 (minutes)
  42- 43  I2     arcsec  DEs      Declination 1950 (seconds)
  47- 51  F5.2   mag     B(0)     []? Integrated magnitude
      52  A1     ---     n_B(0)  *[RMCPHK] Origin of B(0)
  55- 59  I5     km/s    RV       Velocity
  62- 64  I3     km/s    e_RV     Mean error on RV
--------------------------------------------------------------------------------
Note on Name:
    Names with RA given with five digits are normally from
    Zwicky catalogues; the other (with four digits in RA,
    starting or not with the letter A) are anonymous
    galaxies.
Note on n_B(0):
    C = CCD photometry (CfA redshift survey 1992)
    H = Zwicky magnitude split by observers
    K = Markarian catalogue of galaxies (1967)
    M = MCG (Vorontsov-Velyaminov et al. 1962-68)
    P = observer's eye estimate
    R = RC1 (de Vaucouleurs & de Vaucouleurs 1964)
--------------------------------------------------------------------------------
```

Figure 3.2: A first alternative to write lengthy explanations

```
Byte-by-byte Description of file: table1
--------------------------------------------------------------------------------
   Bytes Format  Units   Label    Explanations
--------------------------------------------------------------------------------
   1-  5  A5     ---     Cluster  Abell cluster designation
   7- 22  A16    ---     Name     Galaxy name (1)
  24- 25  I2     h       RAh      Right Ascension 1950 (hours)
  27- 28  I2     min     RAm      Right Ascension 1950 (minutes)
  30- 33  F4.1   s       RAs      Right Ascension 1950 (seconds)
      35  A1     -       DE-      Declination 1950 (sign)
  36- 37  I2     deg     DEd      Declination 1950 (degrees)
  39- 40  I2     arcmin  DEm      Declination 1950 (minutes)
  42- 43  I2     arcsec  DEs      Declination 1950 (seconds)
  47- 51  F5.2   mag     B(0)     []? Integrated magnitude
      52  A1     ---     n_B(0)   [RMCPHK] Origin of B(0) (2)
  55- 59  I5     km/s    RV       Velocity
  62- 64  I3     km/s    e_RV     Mean error on RV
--------------------------------------------------------------------------------
Note (1): Names with RA given with five digits are normally from
    Zwicky catalogues; the other (with four digits in RA, starting or
    not with the letter A) are anonymous galaxies.
Note (2):
    C = CCD photometry (CfA redshift survey 1992)
    H = Zwicky magnitude split by observers
    K = Markarian catalogue of galaxies (1967)
    M = MCG (Vorontsov-Velyaminov et al. 1962-68)
    P = observer's eye estimate
    R = RC1 (de Vaucouleurs & de Vaucouleurs 1964)
--------------------------------------------------------------------------------
```

Figure 3.3: A second alternative to write lengthy explanations

```
XTENSION= 'TABLE   '           / Ascii Table Extension
BITPIX  =                    8 / Character data
NAXIS   =                    2 / Simple 2-D matrix
NAXIS1  =                   52 / Number of bytes per record
NAXIS2  =               175779 / Number of records
PCOUNT  =                    0 / Get rid of random parameters
GCOUNT  =                    1 / Only one group (isn't it obvious?)
TFIELDS =                   13 / Number of data fields (columns)
EQUINOX =                2000. / Equinox of coordinates (J system)
EPOCH   =               1989.0 / Epoch of coordinates
EXTNAME = 'lmc.dat '           / The Large Magellanic Cloud
         =======================================================================
TBCOL1  =                    1 / ============== Start column +0
TFORM1  = 'A12     '           / Fortran Format
TTYPE1  = 'MACS    '           / Designation
TBCOL2  =                   14 / ============== Start column +13
TUNIT2  = 'h       '           / Unit: hour
TFORM2  = 'I2      '           / Fortran Format
TTYPE2  = 'RAh     '           / Right Ascension J2000 , Epoch 1989.0 (hours)
TAMIN2  =                    0 / Allowed minimal value
TAMAX2  =                   24 / Allowed maximal value EXCLUSIVE (never reached)
TBCOL3  =                   17 / ============== Start column +16
TUNIT3  = 'min     '           / Unit: minute
TFORM3  = 'I2      '           / Fortran Format
TTYPE3  = 'RAm     '           / Right Ascension J2000 (minutes)
TAMIN3  =                    0 / Allowed minimal value
TAMAX3  =                   60 / Allowed maximal value EXCLUSIVE (never reached)
TBCOL4  =                   20 / ============== Start column +19
TUNIT4  = 's       '           / Unit: second
TFORM4  = 'F6.3    '           / Fortran Format
TTYPE4  = 'RAs     '           / Right Ascension J2000 (seconds)
TAMIN4  =                0.000 / Allowed minimal value
TAMAX4  =               60.000 / Allowed maximal value EXCLUSIVE (never reached)
TBCOL5  =                   27 / ============== Start column +26
TFORM5  = 'A1      '           / Fortran Format
TTYPE5  = 'DE-     '           / Declination J2000 (sign)
TASET5  = '+-      '           / Allowed character set
TBCOL6  =                   28 / ============== Start column +27
TUNIT6  = 'deg     '           / Unit: degree
TFORM6  = 'I2      '           / Fortran Format
TTYPE6  = 'DEd     '           / Declination J2000 , Epoch 1989.0 (degrees)
TAMIN6  =                    0 / Allowed minimal value
TAMAX6  =                   89 / Allowed maximal value
TBCOL7  =                   31 / ============== Start column +30
TUNIT7  = 'arcmin  '           / Unit: minute of arc
TFORM7  = 'I2      '           / Fortran Format
TTYPE7  = 'DEm     '           / Declination J2000 (minutes)
TAMIN7  =                    0 / Allowed minimal value
TAMAX7  =                   60 / Allowed maximal value EXCLUSIVE (never reached)
TBCOL8  =                   34 / ============== Start column +33
TUNIT8  = 'arcsec  '           / Unit: second of arc
TFORM8  = 'F5.2    '           / Fortran Format
TTYPE8  = 'DEs     '           / Declination J2000 (seconds)
TAMIN8  =                 0.00 / Allowed minimal value
TAMAX8  =                60.00 / Allowed maximal value EXCLUSIVE (never reached)
TBCOL9  =                   40 / ============== Start column +39
TFORM9  = 'I1      '           / Fortran Format
TTYPE9  = 'Npos    '           / Number of positions used
```

Figure 3.4: The FITS extension header constructed from the `ReadMe` file of Figure 3.1

```
TBCOL10 =                   42 / ============== Start column +41
TUNIT10 = 'mag     '            / Unit: magnitude
TFORM10 = 'F5.2    '            / Fortran Format
TTYPE10 = 'Mag     '            / []?=99.00 Instrumental Magnitude (to be used
                                  only in a relative sense)
TBNUL10 =                99.00 / NULL (undefined) value
TBCOL11 =                   48 / ============== Start column +47
TFORM11 = 'I1      '            / Fortran Format
TTYPE11 = 'PosFlag '            / [0/1] Position Flag (0: ok, 1: internal error
                                  larger than 0.5")
TAMIN11 =                    0 / Allowed minimal value
TAMAX11 =                    1 / Allowed maximal value
TBCOL12 =                   50 / ============== Start column +49
TFORM12 = 'I1      '            / Fortran Format
TTYPE12 = 'MagFlag '            / [0/1] Megnitude Flag (0: ok, 1: bad photometry
                                  or possible variable)
TAMIN12 =                    0 / Allowed minimal value
TAMAX12 =                    1 / Allowed maximal value
TBCOL13 =                   52 / ============== Start column +51
TFORM13 = 'I1      '            / Fortran Format
TTYPE13 = 'BochumFlag'          / *[0] Bochum Flag
TAMIN13 =                    0 / Allowed minimal value
TAMAX13 =                    0 / Allowed maximal value
         ======================================================================
END
```

Figure 3.4: The FITS extension *(continued)* header constructed from the ReadMe file of Figure 3.1

The data files can be automatically transformed into FITS: the above figure shows the *ascii table extension* header generated from the description file shown in Chapter 3. Files can be copied directly in this format with the *cdsclient* routines. This set of programs also includes for instance an access to the 1.1 version of the Guide star catalogue *(findgsc)* as well as a few other utilities related to the **Catalogue Service**[16] or SIMBAD at CDS.

The transformation to FITS is made directly in the CDS *ftpd* server at the Internet Node **cdsarc.u-strasbg.fr** (detailed documentation[17] on the CDS *ftpd* server).

[16] http://vizier.u-stasbg.fr//Cats.html
[17] http://vizier.u-stasbg.fr/doc/ftp.htx

Chapter 4

The *cdsclient* package

The **cdsclient** package is a set of C and csh routines which can be built on Unix stations or PCs running Linux, which once compiled allow to query some databases located at CDS or on mirrors over the network.

The cdsclient package includes two generic query programs:

- vizquery[1], a program to remotely query VizieR. It connects the *VizieR* server via the HTTP protocol (requires an access to the **DarkOrangeport 80**)
- find_cats[2], a program for fast access to large surveys from a list of positions, via a dedicated client (requires an access to the **DarkOrangeport 1660**)

4.1 Package Installation

The tar file is available at http://cdsarc.u-strasbg.fr/ftp/pub/sw/cdsclient.tar.gz or as the file **cdsclient.tar.gz** in the ftp://cdsarc.u-strasbg.fr/pub/sw/ directory.

1. Once the file is copied e.g. as the file `cdsclient.tar.gz` in your home directory, untar the file:

   tar xvfz cdsclient.tar.gz

   (the `z` *option is available in the GNU tar; if your installation doesn't know the* `z` *option, execute*
   gzip -cd cdsclient.tar.gz — tar xvf -
)

   A subdirectory `cdsclient-purple4V.vv` (where `purple4V.vv` represents the version number) is created; move to that directory by
   cd cdsclient-purple4V.vv

2. To configure the package for your system, type

   ./configure
   (on some old versions of System V, you might need to type **sh configure** *)*

[1] vizquery.htx
[2] See 4.2

If you wish a final installation (executables, libraries, and manpages) in a non-standard directory tree structure (the default is `/usr/local`), specify the preferred **prefix**, e.g.
./configure --prefix=$HOME
to prepare an installation of executables in your `~/bin` directory — an installation which does not require any `root` privilege.

3. Type **make** to compile the package.

4. Type **make install** moves the various pieces to the standard directories (`/usr/local/bin` by default for the executables, see item 2 above to reconfigure, or just modify the value of `purple4PREFIX` in the `Makefile`)

There are several query programs; as a rule, each program can be executed with the `-help` option to list the basic available options, e.g.

findgsc -help

A `-HELP` option is also available for details on the column contents for the programs that return data, like `findgsc` or `findpmm`

4.2 Package Contents

The available programs are:

- **findcat** allows to retrieve the existing catalogues in CDS Archive. Usage:

  findcat [key... | catalogue_number]

  keys are words to look for, e.g. RADIO, Author's name, etc. . . , or catalogue number e.g. 8059 B/hst J/A+AS/94/519

- **lscat** displays a summary of the files making up a catalogue. Usage:

  lscat catalogue_number [catalogue_number...]

- **catcat** allows to copy a catalogue or a file extracted from a catalogue, either in plain ascii, in tar format, or in FITS format. Usage:

  catcat [-#] [-fits[.Z|.gz]|-tar[.Z|.gz] catalog[/file]

  -#: Specify how many lines from each file (default all) are to be printed.
  -tar: Get catalogue file(s) in TAR format
  compressed versions as tar.Z (Unix compress) or tar.gz (gzip)
  -fits: Get catalogue file(s) as FITS tables
  compressed versions as fits.Z (Unix compress) or fits.gz (gzip)
  catalog: catalog designation, e.g. 8059, VIII/59, J/A+AS/94/519 ...

  Note: The description file is named `ReadMe`, which means that short explanations may be displayed with commands

  catcat VIII/59/ReadMe
  catcat J/A+AS/94/519/ReadMe

- **findgsc** allows to query (by coordinates or from the GSC number) the Guide Star Catalogue (versions 1.1, 1.2 ot ACT — see the VizieR GSC pages[3]).

  Usage:

  findgsc [1.2] J2000-center [-r radius_arcmin] [-n max_found_stars]
  findgsc [1.2] -g GSCfld[-number] [-n max_found_stars]
  findgsc [1.2] - [options] green4(centers in stdin)

  Described by A. Preite-Martinez & F. Ochsenbein in "Handling & Archiving Data from ground-based Telescopes", Trieste April 21-23, 1993, Eds. M. Albrecht & F. Pasian, ESO Conference and Workshop Proceedings No. 50, p. 199

  Example: find out GSC stars within 5' of centers specified in file `mycenters` which contains:

  ```
  12 34 12.5 -34 23 12
  13 24 57.1 +61 12 34
  ```

  just execute

  findgsc - -r 5 ¡ mycenters

  green4**Note**: in order to avoid congestions, the size of the file containing the list of centers (`mycenters`) is limited to a maximum of about 40Mbytes.

- **findgsc2.2** *(2001)* allows to query the GSC2.2 Catalog (Cat. I/271) in a similar way; this version of the GSC2 includes about 500 million sources.

- **findgsc2.3** *(2006)* allows to query the GSC2.3 Catalog (Cat. I/305) in a similar way; this version of the GSC2 includes about 945 million sources.

- **finducac1** *(2000)* allows to query the UCAC1 catalog *(First U.S. Naval Observatory CCD Astrograph Catalog, Zacharias et al.)* including about 27million sources (Cat. I/268)

- **finducac2** *(Oct. 2003)* allows to query the UCAC2 catalog *(Second U.S. Naval Observatory CCD Astrograph Catalog, Zacharias et al.)* including about 48million sources (Cat. I/289)

- **findpmm** allows to query the USNO-A2.0 Catalog (Dave Monnett et al., see http://psyche.usno.navy.mil/pmm) which contains over 500,000,000 sources compiled from the blue/red overlaps of the detection lists generated from scans of POSS-I O and E, SRC-J and ESO-R plates. Usage similar to findgsc.

  Example: find out USNO-A2.0 stars within 3' of centers specified in file mycenters which contains, ordered by increasing distances from specified centers:

  ```
  12 34 12.5 -34 23 12
  13 24 57.1 +61 12 34
  ```

  just execute

  findpmm - -r 3 -sr ¡ mycenters

  Note: the program **findpmm1** allows to query the USNO-A1.0 catalogue

[3] /viz-bin/VizieR?-source=GSC

- **findusnob1** (Dec. 2002) allows to query the USNO-B1.0 Catalog (Dave Monnett et al., http://www.nofs.navy.mil/data/fchpix/; see also Catalog I/284) which contains over 1 billion sources compiled from the blue/red/infrared plates of several surveys made over 50 years including the POSS-I and POSS-II. Besides the position and proper motions, magnitudes and star/galaxy extimators are provided. **red3 Note that the format of the catalog has changed slightly in November 2005**, some edition problems were discovered near the poles, see details in the "History" section of the ReadMe file of catalog I/284; we apologize for any inconvenience these changes might have introduced.

- **find2mass** (Jul. 2003) allows to query the 2MASS Final Point Source Catalog (http://www.ipac.caltech.edu/2mass/ ; see also Catalog II/246) which contains 470,992,970 sources covering virtually the while sky in the infrared bands JHK.

- **findenis** (Sep. 2005) allows to query the DENIS 3rd Release (http://cdsweb.u-strasbg.fr/denis.html; see also Catalog B/denis) which contains 355,220,325 sources in the infrared bands IJK. Older versions of DENIS can be queried with **findenis1** (version 1), **findenis2** (second version), etc.

- **findsdss3** (Oct. 2005) allows to query a subset of the SDSS 3rd release (see Catalog II/259, and http://www.sdss.org).

- **findsdss4** (Dec. 2006) allows to query a subset of the SDSS 4th release (see Catalog II/267, and http://www.sdss.org).

- **findsdss5** (Feb. 2007) allows to query a subset of the SDSS 5th release (see Catalog II/276, and http://www.sdss.org).

- **findsdss6** (Feb. 2008) allows to query a subset of the SDSS 6th release (see Catalog II/282, and http://www.sdss.org).

- **findnomad1** (Jul. 2006) allows to query the NOMAD1 catalog. (see Catalog I/297, and http://www.nofs.navy.mil/nomad/). The catalog merges USNO-B1, UCAC2, Tycho-2, 2MASS and contains over 1.1billion sources. Efficient queries available from position, magnitudes and colors.

- **findcmc14** (Dec. 2006) allows to query the *Carlsberg Meridian Catalogue La Palma* number 14 (see Catalog I/304). The catalogue gather about 100 million stars in the declination range $-30°$ to $+50°$.

- **find_cats** (Jan. 2008) is a program to query the large surveys (GSC, UCAC, USNO, 2MASS, SDSS ...) from a list of positions. For each position given, *find_cats* returns the results from one or several of the large surveys. To minimize the number of bytes transferred on the network, the program **find_cats.gz** returns gzipped (compressed) output.

  Example: **find_cats my_file -r 0.5 GSC1.2 UCAC2: -r 0.3 2MASS: -lmJ 6,11**

  to query around the positions (stored in the file *my_file*) on catalogues:

  - GSC1.2 within a target radius of 0.5arcmin
  - UCAC2 within a target radius of 0.3mrcmin
  - 2MASS within a target radius of 0.5arcmin and J mag in range 6-11.

  The rule is that, when a catalog is followed by a colon (**:**), the arguments following the catalog apply only to that catalog.

- **vizquery** (Sep. 2003) a generic program to query VizieR (or any mirror) and get the results in one of the various available formats: ascii, HTML, TSV, XML, VOTable, FITS... A more complete description can be cound in the vizquery page[4].

- **simref** allows to retrieve bibliographical references (authors & title) from SIMBAD database. Usage:
  simref *bibcode*

  (Note that the bibcode need not to be complete, full volumes are listed when the page numbers are omitted. The year is not required for the most important journals)

- **simbib** allows to retrieve bibliographical references from author's names or words from title. Usage:
  simbib [minimal_year] word(s)...

  Example: **simbib 1990 white dwarf X-ray**

  retrieves the references simce 1990 dealing about white dwarfs in X-ray domain.

- **abibcode** adds, to an input file assumed to contain LaTeX references in the form

  `\bibitem`{*label*} ...

  into one of the two macros:

  `\simOK`{*bibcode*} (reference known in Simbad), or

  `\NOsim`{*text*} (not existing in SIMBAD)

[4] vizquery.htx

Chapter 5

Selected "Manual Pages"

This chapter details a few utilities which help for the standardisation of the electronic tables *(acut, trcol)*, for checking the standardized description and its associated data files *(anafile)*, and for transforming the data files into FITS *(tofits)*. These programs are readily available at ftp://cdsarc.u-strasbg.fr/pub/sw/ana.tar.gz

(1) (*Rev.* Feb-2000)	*Gets parts of a file*	**acut**

SYNTAX

acut [−**d***c*] [−**T** *s1 s2*] [−**a***cols*|−**c***cols*|−**f***cols*]... [−**i***text*]... [*file...*]

DESCRIPTION

The **acut** command allows to extract and reformat columns of a file, and to insert fixed text. Unlike **cut**(1), the order of the columns is arbitrary, and several cut/insert arguments can be specified. The columns can be raw characters (−**c**) or intervals delimited by a *delimiter* character (−**f**).

acut also allows to *translate* (replace characters) in specified columns (**T**), and to remove leading and/or trailing blanks in specified columns (**h**, **j** and **t**), or to perform regular expression substitution in the specified columns.

acut is especially useful for reformatting decimal numbers which can be properly aligned (see below)

OPTIONS

−**d***c* defines the character used as a *delimiter* between columns; this delimiter is only used in cunjunction with the **-f** option. Its default is the *tab*.

−**T** *s1 s2* defines the *translation table*: characters from *s1* are converted into the corresponding characters of *s2* (byte-by-byte translation). Strings *s1* and *s2* must have the same length. The dash (-) may be used to specify intervals, e.g. `a-z` for the 26 lowercase letters. Non-printable characters can be specified using the C conventions (e.g. `\n` for the newline, or `\e` for the Escape).

−**a***cols* asks to insert a *delimiter* (the character defined by the −**d***c* option) between character columns; for example, **acut** `-a1-15,16-` inserts a tab between bytes 15 and 16.
Note that *only one* −**a** option can be specified.

−**c***cols* defines a set of *character columns*. *cols* expressions can make use of commas and dashes, e.g. `-c1,80,2-5` asks to pick successively the columns 1, then 80, then the range 2 to 5. *cols* may be empty, meaning *all consecutive columns between the previous* −**c** *and next* −**c** *definitions.* For details on how *cols* can be written, refer to the *cols* section.

−**f***cols* is similar to the −**c** option, but columns are defined as the set of bytes between two *delimiter* characters.

−**i***text* to insert the fixed text following the −**i** into each line of the output. The −**i** alone asks to insert the *delimiter* (specified by the −**d** option)

cols EXPRESSIONS

cols expressions allow to define a set of positions (starting from 1) representing either byte positions (**−c**) or column numbers (**−f**). Numbers may be separated by commas, e.g. `-c1,5` for byte numbers 1 and 5, or by a *dash*, like `-c5-10` for bytes 5 through 10 (i.e. 6 consecutive bytes), or by a *plus* like `-c5+6` for 6 bytes starting at byte #5 (i.e. bytes 5 through 10).

A missing number after the *dash* means *up to end of line*, and a missing number before the dash means *from byte following the last defined column.* A *repetition factor* can be added, e.g. `-c5+6*4` to specify the 4 fields laying over bytes $5-10$, $11-16$, $17-22$ and $23-28$.

cols may also be

- **−c** meaning *all consecutive columns between the previous and the next* cols *definition*
- **−c..** meaning *all remaining columns*, i.e. columns not mentioned in previous arguments.

cols expressions may end with the following reformatting options:

- **%** (format %[+][0]*n*[**:**—**.**]*d*[t] asks to align the numeric value along a decimal point (.) or at the end of an integer number, assuming up to *d* decimals or trailing spaces reserved for comments like uncertainty flags; a sign can be inserted (+), and the number may be zero-filled (**0**). See the section below on *Decimal Numbers* for details
- **s** *regular_expression_substitution* asks to perform a substitution in the specified set of bytes in a way similar to **sed(1)** See the section below on *pattern substitution* for details.
- **a** *value* asks to add to the column an (integer) *value* specified.
- **+** = (format +*n*[**a**|**A**|**d**|**x**|**o**]) asks to edit a counter of width *n* when the field defined is identical to same field in the preceding line (see *Counter* section below)
- **b** asks to *blank* the *cols* if their contents is identical to the previous line. This option is only available for **−c** column specifications.
- **B** asks to replace blank columns by the contents of the previous record. This option is only available for **−c** column specifications.
- **h** asks to remove the *head* blanks; it is identical to %− (left-aligned format)
- **j** asks to *justify* blanks; it has the same effect as **ht**
- **l** asks to adjust to the left, and is identical to **h**
- **n** indicates a *numeric* field (see below the section on *Decimal Numbers*)
- **r** asks to adjust to the right, and is identical to **t**
- **t** asks to remove the *tail* blanks;
- **T** asks to *translate* the *cols* according to the translation table specified by the **−T** option. The **−T** option must of course precede the *cols* definitions.

For instance, translating the first column in uppercase and removing leading & trailing blanks in the other columns can be written as

`-T a-z A-Z -f1T -f2-j`

Decimal Numbers (%)

With the %[+][0]*n*[**:**—**.**]*d*[t] option, the field is assumed to contain a decimal number with up to *d* decimals. This option performs also realignments of possible *prefix* (typically a limit flag) and of possible *suffix* (typically an uncertainty flag like a colon).

In this option, the following transformations are performed:

- the sign can be systematically inserted at the very left whith the %+ option.
- the decimal point is aligned in column $(n-d)$ with the %$n.d$ option, or the rightmost digit is aligned in column $(n-d)$ with the %$n:d$ option;
- the blanks between the sign and the digits are removed, or are *filled with zeroes* when the n number starts by **0**
- the prefix, if any, is left flushed,
- the suffix, if any, is right flushed.
- with the `t` suffix, numbers are *truncated* to the specified number of decimals if more than *d* decimals are supplied in the input.

With a "format" `%10.3`, the following lines show the transformations on a few numbers:

```
|1234567890|1234567890|1234567890|
 -------|----------|----------|----------|
  Input | -  12    |12.2:     | <10?     |
 Output |   -12.   |     12.2 :|<   10.  ?|
 -------+--------------------------------+
```

When the alignment is not possible or a truncation occured, a warning message is issued.

PATTERN SUBSTITUTION (s)

In a way similar to **sed(1)**, the column defined can be substituted. For instance, removing all blanks in the column between bytes 1 and 12 can be written:

```
acut -c1-12s'/ //g' -c13-
```

or squeezing the blanks (leaving just one when there are several)

```
acut -c1-12s'/  */ /g' -c13-
```

The substitution can use the \1 ... patterns matched as in sed: for instance inverting the texts separated by a comma between columns 5 and 14, leaving the rest untouched, can be written

```
acut -c -c5-14s'/^\([^,]*\),\(.*\)/\2,\1/' -c
```

COUNTERS

It is possible to ask to write a *counter* which is increased each time two consecutive lines have an identical column, and reset to 1 when the column differs from the previous line. This option may also follow the **B** *cols* expression, as in the example
`-c1-11B+a`
which asks to replace a blank field in bytes 1-11 by a repetition of the contents from previous line, followed by `a` for the first (non-blank) line, `b` for the next (blank) line, etc.
The complete option is $+n$[**a**|**A**|**d**|**x**|**o**] where:

- the number n is the width of the extra column with the edited counter; the default value is 1
- the letter specifies how to edit the counter:
  - **a** for lowercase alphabetic
  - **A** for uppercase alphabetic
  - **d** for decimal
  - **o** for octal
  - **x** for hexadecimal lowercase
  - **X** for hexadecimal uppercase

RESTRICTIONS

The **–c** and **–f** options are mutually exclusive.

RETURNED STATUS

The **acut** command returns 0 in case of success, and non-zero when bad options or unreadable files are found.

EXAMPLES

1. To generate a tab-separated-table from an ascii file, column 1 from bytes 1 to 10, column 2 from bytes 11 to 20, and the rest as column 3:
   `acut -a1-10,11-20,21-`
2. To rewrite the degree part of the declination in bytes 10-12:
   `acut -c1-9 -c10-12%+03 -c13-`
3. To generate a file containing on each line: the column 1, then column 80, then the columns 11 through 20 which are assumed to contain a (possibly badly aligned) number with up to 3 decimals, and each field preceded with an obvious explanation, then all other columns:
   `acut -i"Col.1=" -c1 -i" Col.80=" -c80 -i" Cols11-20:  " -c11-20%10.3 -i/ -c..`
4. To remove all blanks in the columns 10 to 20, leaving other columns intact:
   `acut -c -c10-20s'/ //g' -c`

SEE ALSO

cut(1) sed(1)

trcol (*Rev.* Mar 1993) *Delete or Translate columns of a file* **(1)**

SYNTAX

trcol [–**b***cols*] [–**B***cols*] [–**d***cols*] [–**t***cols s1 s2*] [*file...*]

DESCRIPTION

The **trcol** command allows to delete or translate specified columns from a file. A test for blanks (spaces or tabs) may be performed (–**b** option).

The *cols* selection has the form *start[–end[,...]]*, *start* and *end* being the starting and ending column of a field. For instance, colums 1, 3 to 5, and beyond 80 (81 or more) are written `1,3-5,81-`. The complete line is written `1-`

Note that **trcol** cannot exchange columns nor insert text; see **acut**(1) for such facilities.

OPTIONS

The options allow to specify whether we wish to *delete* or *translate* columns.

–**b***cols* asks to *delete with blank test* the specified columns. Non-blank columns are deleted, but a warning message is issued for each non-blank character which was removed. The returned status is the number of non-blank deleted characters.

–**B***cols* is similar to the *-b* option, but no warning message is issued; just the number of deleted non-blank characters is returned as a status.

–**d***cols* asks to *delete* the specified columns without any test.

–**t***cols* *s1 s2* asks to translate in the columns defined by *cols* characters from *s1* into the corresponding characters of *s2* (byte-per-byte translation). Strings *s1* and *s2* must have the same length. The dash (-) may be used to specify intervals, e.g. `a-z` for the 26 lowercase letters. Non-printable characters can be specified using the C conventions.

file is the name of one or several (concatenated) input files. *stdin* is the default input file.

RESTRICTIONS

Only one operation is performed for each column; when the column ranges specified by the various options overlap, the operation applied on the shortest column is applied.

RETURNED STATUS

The **trcol** command returns the number of non-blank deleted bytes (with the –**b** or –**B** option).

EXAMPLE

Remove from standard input columns 2 to 5 with error message for non-blank columns, change the blank of column 6 to a zero, and convert cols 73 and above to lower-case letters:

```
trcol -b2-5 -t6 ' ' '0' -t73- 'A-Z' 'a-z'
```

SEE ALSO

acut(1)

(1) (*Rev.* February 2000) *analyze a file and print statistics.* **anafile**

SYNTAX

anafile [**−v**] [**−l**] [**−a**] [**−u**] [**−cc**] [**−f**[*x*] *format_file*] [**−w** *width*] [[**−t** *table_structure*] *input_file*...]

DESCRIPTION

The **anafile** program is meant mainly to check the concordance of a *description file* (generally named `ReadMe`) with the actual data files, and to report the inconsistencies.

Without a description file, **anafile** analyses the beginning or the complete *file*(s), assigns a *file class* to each file, and lists on request the column (byte-per-byte) statistics. The possible *file class* values are detailed in the section *File Classes* below.

OPTIONS

Without any option, **anafile** examines the first 2880 bytes, and assigns the *file class*.

−a lists the position (line and column numbers) of each character which leads to a classification as *ascii binary* or *binary* file.

−cc generates a detailed list of the frequencies of every character in each column: the characters of each column are listed in order of their decreasing frequency.

−f*x* *format_file* uses the contents of *format_file* to check the compliance of the file to the specified format. The *x* may be used for further options concerning this format file like computation of ranges, verifications against the *CDS Standards*, etc... (see the section *Format File* below).

−l asks to examine the complete files to assign the *file class*, and lists the number of lines as well as the number of bytes of the longest line.

−t *table_structure* indicates that the next *file* argument designates a data file which contains data structured like *table_structure* (for instance an excerpt of a table). *table_structure* is therefore a name which must exist in one of the *Byte-by-byte Description ...* section in the `ReadMe` file.
A value of - for *table_structure* asks to stop this behaviour.
Note that this option can only work when a **−f** specification precedes -bf–t.

−u lists the columns which are constant (i.e. have exactly the same contents) over all lines. This option may be used to check that e.g. that the decimal points are correctly aligned, or to find out the blank columns which could be removed with *trcol*(1).

−v is a verbose option.

−w *width* specifies the assumed column width for *ascii bulk* or *ebcdic* files; such files have no *linefeed* embedded, and the length of each line must be assumed.

FORMAT FILE

A typical format file (specified via the **−f** option) contains the following:

```
Byte-per-byte Description of file: hbc
--------------------------------------------------------------------
   Bytes Format  Units   Label    Explanations
--------------------------------------------------------------------
   2-  4  I3     ---     HBC      [1,423]+ HBC number.
       5  A1     ---     NEBUL    [n] Nebulosity association flag.
       6  A1     ---     REMARK   [*] Remark flag.
   8- 18  A11    ---     NAME     [A-Z0-9@.+-]! Star name.
  20- 56  A37    ---     OTHER    Other designation.
  59- 60  I2     h       RAh      Hours of right ascension (1950.0).
  62- 63  I2     min     RAm      Minutes of right ascension.
  65- 69  F5.2   s       RAs      Seconds of right ascension.
      71  A1     -       DE-      Sign of declination (1950.0).
  72- 73  I2     deg     DEd      Degrees of declination.
  75- 76  I2     arcmin  DEm      Minutes of declination.
  78- 81  F4.1   arcsec  DEs      Seconds of declination.
  83- 90  A8     ---     REF      References to the position.
```

The *format file* is made of five columns: the *byte position*, the *format*, the *units*, the *label* and an *explanation text*. Such a file is interpreted by **anafile** , is reedited in a standard form (on the screen if the **–v** option is present, or the *format file* is rewritten with the **–fw** option), and is used for data check. Note also that special *labels* are understood by **anafile** (see *Special Labels* below).

The *explanation text* may contain further restrictions concerning the *range* for numeric fields or the *character set* allowed for alphabetical field; refer to *Validity Checks* section below.

Note that the *byte position* may be specified as *relative* from the end of the previous field when followed by the `X` letter, as it is in Fortran. The number preceding the `X` therefore represents the number of blanks between the two columns.

With the **–f.** (a dot following the f) option, the input format file does not include the *units* column; the reedition fills this column with dashes —

With the **–f1** (a one following the f) option, the first column of the input format file contains only the *starting byte* of the column; the ending byte is derived from the *format* column. The reedition (with option **–f1w**) completes this column with the ending byte.

With the **–f1X** (a one and X following the f) option, it is assumed that a blank always separates two adjacent columns; the contents of the *starting byte*, if existing, is ignored. The reedition (with option **–f1Xw**) computes the *starting–ending* byte column.

With the **–fr** option, the actual *ranges* (minimal / maximal values) of each column are computed.

With the **–fs** option, the *format file* is assumed to conform to the *Standards for Astronomical Catalogues*; further compliance checks (presence of titles, correctness of units, etc...) of this *format file* are then performed.

With the **–fw** option, the *format file* is rewritten according to standards.

The *format options* may be combined, e.g. **–f1.w** asks to rewrite *format file* where no units and no ending columns were supplied.

Special Labels

Some labels are recognized and further checks are performed. The complete list with implied defaults are listed in the *Standards for Astronomical Catalogues.* A few frequent such labels are:

- Right ascension fields `RAh`, `RAm` and `RAs`
- Declination sign `DE-` (only a sign or a blank is accepted)
- Declination values `DEd`, `DEm` and `DEs`
- Positions in decimal degrees `RAdeg` (range $[0, 360[$) and `DEdeg` (range $[-90, +90]$)
- Galactic positions `GLON` (range $[0, 360[$) and `GLAT` (range $[-90, +90]$)
- Position angle `PA`

Validity Checks

The first word (i.e. set of characters followed by a blank) of the explanation of a column may specify *validity checks* to perform about:

1. the available *range* of the value in the column
2. the possibility of *blank* or *NULL* values
3. the *order* of the value within the table (increasing / decreasing value)

For a *numerical* field, a *range* can be specified in the format file if the explanation text starts with a square bracket [ ] as in the `HBC` column in the above example. The opening bracket is [if the lower value is included, and] if the lower value is excluded — *i.e.* the standard mathematical conventions apply. Both lower and upper values are not required; for instance, the specification of any value lower than 100 (100 excluded) is specified by **[,100[**. Writing `[]` is acceptable when no range checking applies — *e.g.* to override the default range implied by the *label* (see *Special Labels* section above).

For an *alphabetical* (*i.e.* A-format) field, the set of the allowed characters may be specified in the format file if the explanation text starts with a square bracket [: permitted characters are surrounded by square brackets `[...]`, and the dash indicates a range (in the ascii sequence). The closing bracket is accepted as a character permitted in the set if it is specified first (i.e. `[]]` means that only the closing bracket is acceptable); the dash is accepted when it is first or last character. On the above example, only `n` (or blank) is accepted in the `NEBUL` column, and uppercase letters, digits, and the symbols `@ . + -` in the `NAME` column.

An exclamation mark **!** *following immediately* the range or character-set specification indicates that the field *cannot be blank*, i.e. cannot be filled with only blanks. In the above example, the Name field can never be blank.

A question mark **?** *following immediately* the range or character-set specification indicates that the field *can be blank*, i.e. can be filled with only blanks. The default concerning blank fields is:

- *numeric* field: the column cannot be filled with blanks, unless the **?** symbol exists;
- *alphabetic* (A-format) field: the column can be filled with blanks; the **!** means that a completely blank field can't exist.

The question mark may be followed immediated by a numeric value to designate non-blank *NULL* values, e.g. **?=99.99** telling that values 99.99 indicate unknown values.

The *order* within a column can be specified with the signs:

+ indicates *strictly increasing* values in the table;

− indicates *strictly decreasing* values in the table;

= , associated to + or −, indicates that the variation is not strict, i.e. the row number $n+1$ may have a value larger than or equal to that of row n if += is specified.

File Classes

Outside the context of a description file, an input file is classified into one of the following categories according to its contents:

- *ascii file*
  standard ascii file, where no other control characters than *tabs*, *carriage-return* and *line-feed* can be found.
- *ascii mailable*
  may include other control characters like *bell* or *escape* which can be sent over the network without problem.
- *ascii binary*
  includes control characters which cannot be sent over the network, like *Control-Q*. (see also the **−a** option)
- *ascii bulk*
  does not include any *line-feed*, which means that the file cannot be seen as a set of lines. (see **−w** option)
- *ebcdic*
  EBCDIC (IBM-specific) file
- *fits*
  FITS file (bulk file with all headers)
- *hdf*
  FITS header (starting with `SIMPLE` or `XTENSION`) without the data part, and stored as a standard ascii file.
- *directory*
  (not an actual file)
- *binary*
  file which cannot be classified in one of the above categories.

Returned Status

The **anafile** command returns 0 in case of success. A non-zero status may indicate bad options or unreadable files; it is also returned when the *format_file* indicated by a **−fs** option does not conform to the *Standards for Astronomical Catalogues*.

See also

awk(1) gawk(1) trcol(1) "*Standards for Astronomical Catalogues*"

tofits (*Rev.* Mar-1997) *Transform a set of documented files into FITS.* **(1)**

SYNTAX

tofits [**–v**] [**–z**|**Z**] [**–u**[**1**|**2**]] [**–fih**] [**–noh**] [**–nodata**] [**–m** *max_records*] [**–d** *data_directory*] [**–i** *description_file*] [**–h** *SIMPLEheader_file*] [**–o** *output_file*] [**–w**[**.fih**]] [[**–t** *table_structure*] *input_file...*]

DESCRIPTION

The **tofits** command translates a set of files described according to the CDS Standards (see the *Standards for Astronomical Catalogues* document) into FITS tables.

The result may consist of either plain FITS files (bulk files including data description and the corresponding values), or of only the *headers*, i.e. the description of the data as plain ascii files; the latter files are normally named $*$.*fih* files.

The creation of the plain FITS files follows the following algorithm:

- With the **–fih** option, from a set of *FITS headers* files ($*$.*fih* files) and of data files as follows:
  1. the basic *FITS header* (on top of all tables) is specified via the **–h** option; its default name is `SIMPLE.fih`
  2. for each table, the *TABLE header* and the data file, files named *file*`.fih` and *file*.

  The complete FITS creation for a set of n tables therefore requires $2n + 1$ files, the data tables plus $n + 1$ $*$.*fih* files.
- Without the **–fih** option, the *Documentation File (*`ReadMe`*)* (see also anafile(1)) is used to generate the missing $*$.*fih* files. A full FITS generation may if necessary be driven by a special shell script stored in a file named `.MakeFITS`.

Note that some files (e.g. tex files, HTML, postscript files) are not converted to FITS.

OPTIONS

Without any option, **tofits** uses the `ReadMe` file and the data files in the current directory and transforms all files described in `ReadMe` into a single FITS file displayed on the terminal.

–d *data_directory* indicates the directory containing the relevant files (data files, $*$.*fih* files, the `ReadMe` if not specified by the **–i** option). The default *data_directory* is the current directory.

–fih tells that no `ReadMe` file is required, all descriptions do exist as $*$.*fih* files; the top header is named `SIMPLE.fih`, unless another name is specified via the **–h** option.

–h *SIMPLEheader_file* provides an alternative name to `SIMPLE.fih` (see option **–fih**)

–help displays a detailed help.

–i *description_file* provides the non-standard (`Intro` or `ReadMe`) name of the *Description file*;

–m *max_records* asks to limit the generated *FITS* file to the specified number of records. By default, all records are converted to *FITS*.

–nodata asks to generate only *FITS headers* (similar to $*$.*fih* files); the data files are ignored.

–noh asks not to generate the top (*SIMPLE*) header; this option is especially useful to append a new FITS table to an existing FITS data set.

–o *output_file* designates the file replacing the default *stdout* (terminal). The usage of a disk file allows to rewrite the actual number of records (the `NAXIS2` parameter), when the number of records stored in the `ReadMe` is not correct (e.g. with the **–t** option). *Note that this option is not compatible with the compression.*

–t *table_structure* indicates that the next *file* argument designates a data file which contains data *structured like table_structure* (for instance an excerpt of a table). *table_structure* is therefore a name which must exist in one of the *Byte-by-byte Description ...* section in the `ReadMe` file.
A value of `-` for *table_structure* asks to stop this behaviour.

–u allows to select how to write the comments related to the units: the **–u1** option comments each unit with its SI (*Système International*) value, while **–u2** option comments each unit with both litteral and SI counterpart.

–v is a verbose option.

–w.fit asks to write a new file for each FITS table; with this option, there are as many FITS files generated as data files, each being a complete FITS file including the *top header*, a *table header* and the corresponding data. The names of the files generated this way have a `.fit` extension.

–w.fih asks to write a new file for each *table header* (generation of **.fih* files). The names of the files generated this way have a `.fih` extension.

–Z compresses the resulting files with the standard Unix **compress**(1) routine. The output cannot be put in a file (incompatible with **–o** option)

–z compresses the resulting files with the **gzip**(1) utility. The output cannot be put in a file (incompatible with **–o** option)

FILES

The *file...* specified on the command line represent the files which should be converted to FITS. If a *file* was preceded by a **–t** option, *file* is assumed to have the same structure as the **–t** *table_structure* argument (– stands for the standard input).

The default is to convert *all described files*, i.e. all files included in the *File Summary* section of the `ReadMe` file, with the exception of the non-convertable files.

A dash – or a dollar sign **$** as a file name asks to stop the conversion; this is useful if one wants to generate only the *top header*, as in

```
tofits -nodata -i DescriptionFile - > SIMPLE.fih
```

Note that if the *file* argument is a directory, it has the same effect as it were preceded by the**–d** flag.

NON-CONVERTED FILES

The extension of filenames is used to decide that a file should not be converted into FITS. Files with names ending by `.fih`, `.fis`, `.tex`, `.dvi`, `.ps`, `.pdf`, `.doc`, `.sty`, `.htm`, `.html`, `.tar` and `.tgz` are not converted.

.MAKEFITS SCRIPT

When a `.MakeFITS` file exists, it is called from **tofits** (via the *Bourne shell* `/bin/sh`) to generate *all* FITS files: this can be executed only when **tofits** is called without any *file* argument.

This script may use (and change) the following numeric environment variables which are set by **tofits** before calling `.MakeFITS`:

- `toFITSd`=0 to generate only the headers (option **–nodata**)
- `toFITSh`=0 to skip the SIMPLE header (option **–noh**)
- `toFITSm`=1 to use the `FILEMARK` extension (which separates two FITS files)
- `toFITSr`=1 is the recursivity count, incremented each time **tofits** is called.

The `.MakeFITS` script may (recursively) call **tofits** .

RETURNED STATUS

The **tofits** command returns 0 in case of success, and non-zero when errors are found.

EXAMPLES

- Convert the files in current directory which are documented in the standard way:
  `tofits > catalog.fits`
- Convert the files of catalogue I/219 into a single compressed FITS file:
  `tofits -z /ftp/cats/I/219 > 1219.fgz`
- Extract the correlation coefficients (indicated by the `CORR` string) from the Hipparcos catalogue stored on a CD-ROM (the description file is named `tofits.dat`, the correlation coefficients table is described by `h_dm_cor.dat`):
  ```
  grep CORR /cd/cats/hip_dm_c.dat \
  | tofits -i /cd/fits/tofits.dat -t h_dm_cor.dat -o h_dm_cor.fit
  ```

SEE ALSO

anafile(1) compress(1) gzip(1) "*Standards for Astronomical Catalogues*"