The ctable package*

Wybo Dekker
wybo@servalys.nl

October 30, 2010

Abstract

The ctable package provides a ctable command for the typesetting of table and figure floats. You will not need to type the usual nested begin...end sequences, as ctable is a command, not an environment. ctable has only 4 arguments, but the optional first one may hold many *key=value* pairs and makes ctable very flexible and extensible. It uses Simon Fear's booktabs package for better vertical spacing around horizontal rules and it provides facilities for making table footnotes.

1 Purpose

The ctable package lets you easily typeset captioned table and figure floats with optional footnotes. Both caption and footnotes will normally be forced within the width of the table. If the width of the table is specified, then tabularx will be used to typeset it, and one or more X column specifiers should be specified. Otherwise tabular will be used.

This package defines the commands \ctable, \tnote and \tmark, and four \tabularnewline generating commands. The latter generate reasonable amounts of whitespace around horizontal rules and are also useful for tabulars outside this package.

Since the ctable package imports the array and booktabs packages, all commands from those packages are available as well.

Note that, in line with the comments that Simon Fear made describing his booktabs package, vertical rules for column separation can be produced with \ctable, but no provisions are made to have them make contact with horizontal rules.

2 Usage

\ctable

\ctable is called with 4 parameters, of which the first is optional:

```
\ctable[options]       % key=value,...
       {coldefs}       % for \begin{tabular}
       {foottable}     % zero or more \tnote commands (see below)
       {table rows}    % rows for the table
```

Options are given as key=value pairs, separated by comma's. Extra comma's, including one behind the last pair, don't hurt. Arguments to option should be put between braces if they contain comma's or equals signs. Currently the following option keys have been defined:

*This document corresponds to ctable v1.17, dated 2010/10/30.

`caption={...}`	table caption; the braces are needed only if your caption contains a comma or an equals sign.
`cap={...}`	for a short caption to go to the `\listoftables`. Without the `cap` option, the full caption will go into the `\listoftables`. If `cap` is given an empty value, no entry in the `\listoftables` will be made. This may be useful, for example, with the `continued` option.
`continued[=...]`	if used, the table will be numbered the same as the previous table. If used without an argument, the caption will be suffixed with ‘ (continued)’, if used with an argument, the suffix will be the argument.
`captionskip=...`	moves the caption relative to the table; the default is `0ex`, which puts captions at their default LaTeX positions: a top caption’s baseline at `1ex` above the top rule position of the table and a bottom caption’s baseline at `4ex` below the bottom rule position.
`mincapwidth=...`	sets the minimum width of the float. Without this option, the width is set to that of the tabular, and the caption and footnotes are typeset within that width. This may be a problem with very narrow tables; `mincapwidth` can then be used to give the float a minimum width. The tabular will be centered in it.
`doinside=...`	command to be run inside, just before the tabular or tabularx environment. You can use this, for example, for the adjustment of the font size with `\small`.
`pos=...`	float position, default: `tbp`.
`label=...`	for `\label`
`width=...`	`tabularx` will be used to typeset the table at the specified width — one or more `X` column specifiers must be provided.
`maxwidth=...`	like the *width* option, but any `X` column specifiers will be replaced with `l` if the resulting table width would thus stay within the specified maximum width. This is especially useful where the LaTeX source is generated by a script.
`center`	center the table in the available text width; this is the default.
`left`	left align the table in the available text width.
`right`	right align the table in the available text width.
`figure`	produce a figure float instead of a table float.
`botcap`	put the caption at the bottom of the float instead of on top of it.
`sideways`	rotate table or figure by 90 degrees anticlockwise and put it on a separate page. With the twoside option for the standard LaTeX document classes, rotation will be -90 on even pages. If you use this option, the `pos` option is not allowed.
`star`	use the starred versions of the `table` and `figure` environments, which place the float over two columns when the `twocolumn` option or the `\twocolumn` command is active.
`nosuper`	in the footnote table, typeset footnote markers on the line, instead of superscripted.
`notespar`	typeset footnotes in a parapgraph instead of in a table.
`framerule=...`	draw a frame around the table with the given rule thickness. The default is `0pt`, so that no frame will be seen.
`framesep=...`	set the distance between the frame and the table to the given dimension. The default is `0pt`.
`framefg=`*r g b*	set the foreground color of the frame (the rule color) to the given triplet of *rgb*-values. The values should be numbers between 0 and 1. The default is `0 0 0` (black).
`framebg=`*r g b*	set the background color of the frame (the color inside the frame) to the given triplet of *rgb*-values. The values should be numbers between 0 and 1. The default is 1 1 1 (white).

The footnotes are placed under the table, without a rule. You therefore probably will want to use the

`\tnote` `\LL` (last line) command if you use footnotes. `\tnote[label]{footnote text}` places [label] footnote text under the table. Can only be used in the foottable parameter described above. The label is optional, the default label is a single *a*. For more detailed control, you can also replace this command with something like `labeltext&footnotetext\NN`.

`\tmark` `\tmark[label]` this command places the superscripted label in the table. It is equivalent with `$^{label}$`. The label is optional, the default label is a single *a*.

The newline generating commands are a combination of `\tabularnewline` and zero or one of booktabs `\toprule`, `\midrule` or `\bottomrule`. These combinations have been made, and short names have been defined, because source texts for complex tables often become very crowded:

`\NN` Normal Newline, generates just a normal new line. An optional dimen parameter inserts extra vertical space under the line.

`\FL` First Line, generates a new line and a thick rule with some extra space under it. An optional dimen parameter sets the line width; the default is 0.08em.

`\ML` Middle Line: generates a new line and a thin rule with some extra space over and under it. An optional dimen parameter sets the line width; the default is 0.05em.

`\LL` Last Line: generates a new line and a thick rule with some extra space over it. An optional dimen parameter sets the line width; the default is 0.08em.

These macros can be used outside `\ctable` constructs.

Finally, for completeness, here are some of booktabs' commands that may be useful:

`\toprule` `\toprule[<wd>]` where <wd> is the optional thinkness of the rule.

`\midrule` `\midrule[<wd>]`.

`\bottomrule` `\bottomrule[<wd>]`.

`\cmidrule` `\cmidrule[<wd>](<trim>){a-b}` where <trim> can be `r`, `l`, or `rl` and the rule is drawn over columns `a` through `b`.

`\morecmidrules` `\morecmidrules` must be used to separate two successive cmidrules.

`\addlinespace` `\addlinespace[<wd>]` inserts extra space between rows.

`\specialrule` `\specialrule{<wd>}{<abovespace>}{<belowspace>}`.

See the booktabs documentation for details.

2.1 The width and maxwidth options

When LaTeX-sources containing tables are generated automatically by a script, it is often not known in advance what the maximum size of an `l` column will be. A good solution for this is to use an `X` specifier, typesetteing the table at the text width with the tabularx package. However, this will result in too much white space in cases where the column contains small texts only. This problem can be solved by using the maxwidth option instead of the width option. The `X` specifiers will then be replaced with `l` as long as the width of the resulting table stays with the specified maximum width.

3 Tips

- When you a table wider than `\textwidth`, use the geometry package and surround your ctable call with `\newgeometry{width=...,margin=...}` and `\restoregeometry`.

4 Examples

Table 1 is an example taken from the related package threeparttable by Donald Arseneau, with an extra footnote. It was typeset with:

```
\ctable[
```

Table 1: The Skewing Angles (β) for Mu(H) + X_2 and Mu(H) + HX [a]

	H(Mu) + F_2	H(Mu) + Cl_2
β(H)	80.9° [b]	83.2°
β(Mu)	86.7°	87.7°

[a] for the abstraction reaction, Mu + HX → MuH + X.
[b] 1 degree = $\pi/180$ radians.
[c] this is a particularly long note, showing that footnotes are set in raggedright mode as we don't like hyphenation in table footnotes.

```
   cap     = The Skewing Angles,
   caption = The Skewing Angles ($\beta$) for
             $\fam0 Mu(H)+X_2$ and $\fam0 Mu(H)+HX$~\tmark,
   label   = nowidth,
]{rlcc}{
   \tnote{for the abstraction reaction,
          $\fam0 Mu+HX \rightarrow MuH+X$.}
   \tnote[b]{1 degree${} = \pi/180$ radians.}
   \tnote[c]{this is a particularly long note, showing that
             footnotes are set in raggedright mode as we don't like
             hyphenation in table footnotes.}
}{                                                          \FL
  &            & $\fam0 H(Mu)+F_2$     & $\fam0 H(Mu)+Cl_2$ \ML
  &$\beta$(H)  & $80.9^\circ\tmark[b] & $83.2^\circ$       \NN
  &$\beta$(Mu) & $86.7^\circ$          & $87.7^\circ$       \LL
}
```

Table 2 is an example with a width specification, taken from the tabularx documentation, with the vertical rules removed. By using the trimming parameters of the booktabs \cmidrule command, some of the horizontal splitting was regained. The left option left aligns the table. It was typeset with:

```
\ctable[
   caption = Example with a specified width of 100mm,
   label   = width,
   width   = 100mm,
   pos     = ht,
   left
]{c>{\raggedright}Xc>{\raggedright}X}{
   \tnote{footnotes are placed under the table}
}{                                                          \FL
   \multicolumn{4}{c}{Example using tabularx}               \ML
   \multicolumn{2}{c}{Multicolumn entry!} & THREE & FOUR   \NN
       \cmidrule(r){1-2}\cmidrule(rl){3-3}\cmidrule(l){4-4}
   one&
   The width of this column depends on the width of the
       table.\tmark &
   three&
   Column four will act in the same way as
   column two, with the same width.                         \LL
}
```

Figures, even single ones, are always put in tabular cells. This is not particularly handy for single pictures, but it eases the construction of arrays of pictures, including sub-captions, delineation, and spacing. For a small example, which also shows how you can simplify the construction of figure arrays, see subsection 5.9 on page 9.

Table 2: Example with a specified width of 100mm

Example using tabularx			
Multicolumn entry!		THREE	FOUR
one	The width of this column depends on the width of the table.[a]	three	Column four will act in the same way as column two, with the same width.

[a] footnotes are placed under the table

5 Option examples

In the following, small examples will be shown illustrating the effect of options. In the left column the relevant part of the source is shown, in the right column you see the result. In most cases you see a standard example on a light yellow background, followed by one or more variations on a light blue background. Where necessary, the example will show boxes to indicate the page and the text body.

5.1 `center, left, right`

These options align the float in the page; the default is `center`:

```
\ctable[
  caption = Centered,
]{c}{}{\FL Table's first row\LL}
```

Table 1: Centered

Table's first row

```
\ctable[
  caption = Left,
     left
]{c}{}{\FL Table's first row\LL}
```

Table 1: Left

Table's first row

```
\ctable[
  caption = Right,
     right
]{c}{}{\FL Table's first row\LL}
```

Table 1: Right

Table's first row

5.2 `nosuper`

Footnote markers in `ctable` are typeset superscripted by default. Use the `nosuper` option to place them on the base line:

```
\ctable{c}{
  \tnote{First footnote}
  \tnote[b]{Second footnote}
}{\FL Table's\tmark\ first\tmark[b]\ row\LL}
```

Table's[a] first[b] row

[a] First footnote
[b] Second footnote

```
\ctable[nosuper]{c}{
  \tnote[a.]{First footnote}
  \tnote[b.]{Second footnote}
}{\FL Table's\tmark\ first\tmark[b]\ row\LL}
```

Table's[a] first[b] row

a. First footnote
b. Second footnote

5.3 notespar

Footnotes in `ctable` are typeset in a paragraph, instead of a table:

```
\ctable{c}{
  \tnote{First note}
  \tnote[b]{Second note}
  \tnote[c]{Third note}
}{\FL Table's\tmark\ first\tmark[b]\ row
      with footnotes\tmark[c]\LL}
```

Table's[a] first[b] row with footnotes[c]

[a] First note
[b] Second note
[c] Third note

```
\ctable[notespar]{c}{
  \tnote[a]{First note.}
  \tnote[b]{Second note.}
  \tnote[c]{Third note, this one is a
            little longer and forces a
            new line at the end.\\}
  \tnote[d]{And here is e very long note:
            \input{thuan}}
}{\FL Table's\tmark\ first\tmark[b]\ row
      with footnotes\tmark[c]\LL}
```

Table's[a] first[b] row with footnotes[c]

[a] First note. [b] Second note. [c] Third note, this one is a little longer and forces a new line at the end.
[d] And here is e very long note: Had our solar system included two suns, the problem would have involved three bodies (the two suns and each planet), and chaos would have been immediately obvious. Planets would have had erratic and unpredictable orbits, and creatures living on one of these planets would never have been able to percieve the slightest harmony. Nor would it have occurred to them that the universe might be ruled by laws and that it is up to man's intellect to discover them. Besides, it is not at all obvious that life and conscience could even emerge in such a chaotic system.

5.4 continued

The `continued` option suffixes the caption with ' (continued)', and lowers the table number by one, so that it obtains the same number as the previous table. This option can be given an argument to replace the default suffix:

```
\ctable[
  caption = Caption,
  mincapwidth = 50mm,
]{c}{}{\FL Table's first row\LL}
```

Table 1: Caption

Table's first row

```
\ctable[
  caption = Caption,
  mincapwidth = 50mm,
  continued
]{c}{}{\FL Table's first row\LL}
```

Table 1: Caption (continued)

Table's first row

```
\ctable[
  caption = Caption,
  mincapwidth = 50mm,
  continued = \textit{(contd)}
]{c}{}{\FL Table's first row\LL}
```

Table 1: Caption *(contd)*

Table's first row

5.5 mincapwidth

`ctable` forces caption and footnotes to stay within the width of the table. Sometimes, however, tables are so narrow, that this is not really what you want. In such cases, use the `mincapwidth` option to give caption and footnotes some extra room:

```
\ctable[
  caption = a lengthy caption
]{c}{}{\FL row1\LL}
```

Table 1: a lengthy caption

row1

```
\ctable[
mincapwidth = 55mm,
    caption = a more or less lengthy caption
]{c}{\tnote[x]{a more or less lengthy note}}{\FL row1\tmark[x]\LL}
```

Table 1: a more or less lengthy caption

row1[x]

[x] a more or less lengthy note

You can set `mincapwidth` to a large value, say `\hsize`, if you want a one-line caption. Note, however, that this may influence the horizontal positioning of the table: values larger than `\hsize` will move a centered table out of the center, a value of `\hsize` will prevent the `left` and `right` options to do their work, because the table is already captured between the left and right margins.

5.6 maxwidth

When LaTeX-sources containing tables are generated automatically by a script, it is often not known in advance what the maximum size of an `l` column will be. A good solution for this is to use an `X` specifier, typesetting the table at the text width with the `tabularx` package. However, this will result in too much white space in cases where the column contains small texts only. This problem can be solved by using the maxwidth option instead of the width option. The `X` specifiers will then be replaced with `l` as long as the width of the resulting table stays with the specified maximum width.

```
\ctable[
  framerule = .1pt,
  maxwidth=3cm
]{lX}{}{\FL 1 & first row\LL}
```

1	first row

```
\ctable[
  framerule = .1pt,
  maxwidth=3cm
]{lX}{}{\FL 1 & test\LL}
```

1	test

5.7 framerule

The following examples show the use of frames and backgrounds. Every table is typeset by `ctable` with a frame around it, but the frame is, by default, drawn with a zero width line, and is therefore invisible. You can make it visible by either changing the linewidth to a positive value or by giving it a background color, which will be used to fill the frame.
Here is a simple table without a frame, followed by one with a red, `1pt` thick frame:

```
\ctable[
  caption = Frame,
]{c}{}{\FL Table's first row\LL}
```

Table 1: Frame

Table's first row

```
\ctable[
  caption = Frame,
  framerule = 2pt,
  framefg = .8 0 0
]{c}{}{\FL Table's first row\LL}
```

Table 1: Frame

Table's first row

As you see, the frame fits closely to the first (\FL) and last (\LL) table lines. This can be a reason to either remove those lines, or to introduce some whitespace between the frame and the table with the `framesep` option:

```
\ctable[
  caption = Frame,
  framerule = 1pt,
  framefg = .8 0 0,
  framesep=10pt
]{c}{}{\FL Table's first row\LL}
```

Table 1: Frame

Table's first row

And finally, we could also frame the table by giving it a, say, yellow backgound instead of a red frame line, or even do both:

```
\ctable[
  caption = Frame,
  framebg = 1 1 0,
  framesep=10pt
]{c}{}{\FL Table's first row\LL}
```

Table 1: Frame

Table's first row

```
\ctable[
  caption = Frame,
  framerule = 2pt,
  framesep = 5pt,
  framebg = 1 1 0,
  framefg = 1 0 0,
  framesep=10pt
]{c}{}{\FL Table's first row\LL}
```

Table 1: Frame

Table's first row

5.8 captionskip

The distance between a top caption and the table is 2ex, but it can be varied with captionskip:

```
\ctable[
  caption = Caption,
]{c}{}{\FL Table's first row\LL}
```

Table 1: Caption

Table's first row

```
\ctable[
    caption =  Caption,
captionskip = 1ex,
]{c}{}{\FL Table's first row\LL}
```

Table 1: Caption

Table's first row

This works for bottom caption, too:

```
\ctable[
  caption = Caption,
   botcap
]{c}{}{\FL Table's first row\LL}
```

Table's first row

Table 1: Caption

```
\ctable[
    caption =  Caption,
captionskip = -2ex,
     botcap
]{c}{}{\FL Table's first row\LL}
```

Table's first row

Table 1: Caption

5.9 figure, botcap

By default, ctable generates a table float, but with the figure option, a figure float is generated instead. The caption stays on top, so if you are accustomed to have bottom caption for your figures, you will probably also need the botcap option:

```
\ctable[caption = a table]{c}{
}{\FL Table's first row\LL}
```

Table 1: a table

Table's first row

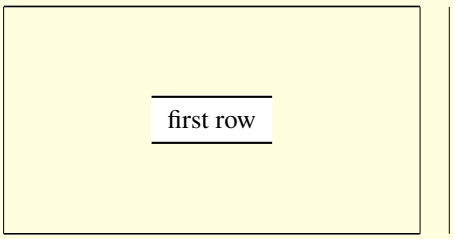
```
\newcommand{\F}[1]{
  \includegraphics[width=\hsize]{#1}
}
\newcolumntype{H}[1]{>{\hsize=#1\hsize}X}
\ctable[
  caption = a figure,
  figure, botcap,
  width=.4\hsize,
]{H{.4}H{.6}}{}{\FL
  \F{penguin}& \F{lion}\LL
}
```

Figure 1: a figure

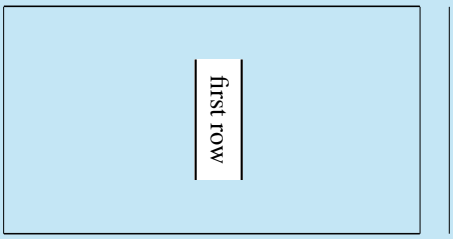
5.10 `sideways`

The `sideways` option creates a landscape table with its head pointing at the spine — when the documentclass' twoside option has been used, that is. The following examples show the effect of the `sideways` option, first on page one, then on page 2. Note that the `caption` option has not been used, so no caption appears:

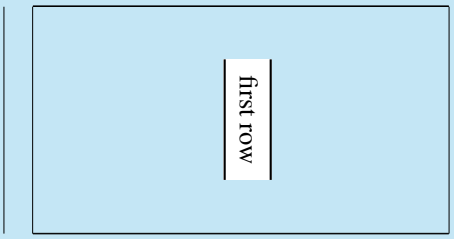
```
\ctable[
]{c}{}{\FL first row\LL}
```




```
\ctable[
  sideways
]{c}{}{\FL first row\LL}
```




```
\setcounter{page}{2}
\ctable[
  sideways
]{c}{}{\FL first row\LL}
```



6 Implementation

```
\RequirePackage{color,xkeyval,array,tabularx,booktabs,rotating}
\def\NN{\tabularnewline}
\def\FL{\toprule}
\def\ML{\NN\midrule}
\def\LL{\NN\bottomrule}
\definecolor{@ctblframefg}{rgb}{0,0,0}%
\definecolor{@ctblframebg}{rgb}{1,1,1}%
\def\@ctblfgcolor#1 #2 #3={\definecolor{@ctblframefg}{rgb}{#1,#2,#3}}
\def\@ctblbgcolor#1 #2 #3={\definecolor{@ctblframebg}{rgb}{#1,#2,#3}}
\def\@ctbltextsuperscript#1{%
  \ifx\@ctblnosuper\empty\@textsuperscript{#1}\else{\footnotesize#1}\fi
}
\def\@ctbldoinside{\relax}
\newdimen\@ctblframesep
\newdimen\@ctblframerule
\newdimen\@ctblwidth
\newdimen\@ctblcaptionskip
\newdimen\@ctblmaxwidth
\newdimen\@ctblmincapwidth
\newdimen\@ctblw % the final width
\newdimen\@ctblfloatwidth
\newdimen\@ctbloldsep
\newdimen\@ctbloldrule
```

Allocate box registers so that we can determine the widths of the tables

```
\newbox\ctbl@t          % tabular saved and measured here
```

Option setting commands from keyval. The table position (here, top, bottom, page) gets a special treatment, since LaTeX does not expand commands there. So instead of putting things like `tbp` in a command like `\@ctblbegin` we put `\begin{table}[tbp]` in it.

```
\define@key{ctbl}{caption}{\def\@ctblcaption{#1}}
\define@key{ctbl}{cap}{\def\@ctblcap{#1}}
\define@key{ctbl}{label}{\def\@ctbllabel{#1}}
\define@key{ctbl}{continued}[(continued)]{\def\@ctblcontinued{#1}}
\define@key{ctbl}{pos}{\def\@ctblpos{#1}\def\@ctblbegin{\@ctblbeg[#1]}}
\define@key{ctbl}{width}{\@ctblwidth=#1}
\define@key{ctbl}{maxwidth}{\@ctblmaxwidth=#1}
\define@key{ctbl}{mincapwidth}{\@ctblmincapwidth=#1}
\define@key{ctbl}{botcap}[]{\def\@ctblbotcap{1}}
\define@key{ctbl}{sideways}[]{\def\@ctblsideways{sideways}}
\define@key{ctbl}{rotate}[]{\def\@ctblsideways{sideways}
                            \PackageWarning{ctable}{
                              using obsolete option 'rotate', use 'sideways' instead}
                            }
\define@key{ctbl}{figure}[]{\def\@ctbltaborfig{figure}}
\define@key{ctbl}{center}[]{\let\@ctblalign\centering}
\define@key{ctbl}{right}[]{\let\@ctblalign\raggedleft}
\define@key{ctbl}{left}[]{\let\@ctblalign\raggedright}
\define@key{ctbl}{star}[]{\def\@ctblstarred {*}}
\define@key{ctbl}{framerule}{\@ctblframerule=#1}
\define@key{ctbl}{framesep}{\@ctblframesep=#1}
\define@key{ctbl}{framefg}{\@ctblfgcolor#1=}
\define@key{ctbl}{framebg}{\@ctblbgcolor#1=}
\define@key{ctbl}{captionskip}{\@ctblcaptionskip=#1}
\define@key{ctbl}{nosuper}[]{\def\@ctblnosuper{1}}
\define@key{ctbl}{notespar}[]{\def\@ctblnotespar{1}}
\define@key{ctbl}{doinside}{\def\@ctbldoinside{#1}}
```

A caption will only be generated if the *caption* option was used, with a non-empty value. If so, it goes in the lot/lof, unless the *cap* option specified a different (probably shorter) value for it. A *cap* option with an empty value inhibits a tof/lof entry.

```
\def\@ctblCaption{
  \ifx\@ctblcap\undefined\let\@ctblcap\@ctblcaption\fi
```

```
   \ifx\@ctblcaption\empty\else
      \def\@ctblcaptionarg{\ifx\@ctbllabel\empty\else\label{\@ctbllabel}\fi
         \@ctblcaption\ \@ctblcontinued\strut}
      \ifx\@ctblcap\empty
         \caption[]{\@ctblcaptionarg}
      \else
         \caption[\@ctblcap]{\@ctblcaptionarg}
      \fi
   \fi
}
```

Need to redefine X columntype, but the array package would generate a warning. So first set the type to be redefined to `\undefined` to suppress the warning. Save the standard X type once in the new type Y

```
\newcolumntype{Y}{X}
\def\@ctblXcolumntype#1{%
  \let\NC@find@X\undefined
  \newcolumntype{X}{#1}%
}
\def\@ctblframe#1#2#3{%
   \@ctbloldsep\fboxsep\fboxsep\@ctblframesep%
   \@ctbloldrule\fboxrule\fboxrule\@ctblframerule%
   \fcolorbox{#1}{#2}{\fboxsep\@ctbloldsep\fboxrule\@ctbloldrule #3}%
}
\newcommand{\tnote}[2][a]{%
   \ifx\@ctblnotespar\empty%
     \hbox{\@ctbltextsuperscript{\normalfont\textit{#1}}}&#2\NN
   \else%
     \@ctbltextsuperscript{\normalfont\textit{#1}}\,#2
   \fi
}
\newcommand{\tmark}[1][a]{%
   \hbox{\textsuperscript{\normalfont\textit{#1}}}}
\newcommand{\ctable}[4][]{%
   \def\@ctbltaborfig{table}%
   \let\@ctblalign\centering%
   \def\@ctblsideways{}%
   \def\@ctblcontinued{}%
   \def\@ctblpos     {}%
   \def\@ctblcaption {}%
   \let\@ctblcap\undefined%
   \def\@ctbllabel   {}%
   \def\@ctblbeg     {\begin{\@ctblsideways\@ctbltaborfig\@ctblstarred}}%
   \def\@ctblbegin   {\@ctblbeg}%
   \def\@ctblend     {\end{\@ctblsideways\@ctbltaborfig\@ctblstarred}}%
   \def\@ctblbotcap  {}%
   \def\@ctblstarred {}%
   \def\@ctblnosuper {}%
   \def\@ctblnotespar{}%
   \def\@ctbldoinside{}%
   \@ctblframerule0pt
   \@ctblcaptionskip=0ex
   \@ctblframesep0pt
   \@ctblwidth=0pt
   \@ctblmaxwidth=0pt
   \@ctblmincapwidth=0pt
   \setkeys{ctbl}{#1}%
```

It makes no sense to use *width* together with *maxwidth* or *pos* together with *sideways*

```
   \ifdim\@ctblwidth=0pt\else
      \ifdim\@ctblmaxwidth=0pt\else
         \PackageError{ctable}{
            You may not use the width and maxwidth options together}{
            Use either width or maxwidth}
      \fi
```

```
   \fi
   \ifx\@ctblpos\empty\else
      \ifx\@ctblsideways\empty\else
      \PackageError{ctable}{
         You may not use the pos and sideways options together}{
         Rotated tables and figures are always typeset on a separate page}
      \fi
   \fi
```

It makes no sense to label a captionless table, because the label can't be placed, leaving the user wondering why references to the table get a ??

```
   \ifx\@ctblcaption\empty
      \ifx\@ctbllabel\empty\else
         \PackageError{ctable}{
            You may not label a captionless table}{
            Such a label can't be referenced}
      \fi
   \fi
```

save the table contents in a box, so we can determine its width, initially, save the table typeset with the tabular environment:

```
   \sbox\ctbl@t{%
      \@ctblXcolumntype{l}% temporarily make  type X = l
      \@ctblframe{@ctblframefg}{@ctblframebg}{%
         \@ctbldoinside
         \begin{tabular}{#2}
            #4%
         \end{tabular}%
      }%
   }%
```

then look if we'll need the tabularx environment:

```
   \newif\if@ctblusex\@ctblusexfalse
   \ifdim\@ctblmaxwidth=0pt
      \ifdim\@ctblwidth=0pt
      \else
         \@ctblusextrue
      \fi
   \else
      \ifdim\wd\ctbl@t>\@ctblmaxwidth
         \@ctblusextrue
      \fi
   \fi
%
% if so, replace tabular with tabularx:
%
   \if@ctblusex
      \sbox\ctbl@t{%
         \@ctblXcolumntype{Y}% restore X
         \@ctblframe{@ctblframefg}{@ctblframebg}{%
            \@ctbldoinside
            \begin{tabularx}{\ifdim\@ctblwidth>0pt\@ctblwidth\else\@ctblmaxwidth\fi}{#2}
               #4%
            \end{tabularx}%
         }%
      }%
   \fi
```

the `ctbl@t` box now contains the table as we want to typeset it; determine its width:

```
   \@ctblw=\wd\ctbl@t
```

Now find the width of the float, `\@ctblfloatwidth`; everything in it will be centered within that width. Normally we'll use the width of the table, `\@ctblw`, but if the mincapwidth, `\@ctblmincapwidth` was set wider than the table, that will be used:

```
   \@ctblfloatwidth=\ifdim\@ctblmincapwidth>\@ctblw
      \@ctblmincapwidth
   \else
```

```
      \@ctblw
   \fi
```

\@ctblbegin is now defined as something like \begin{table}[tbp].

```
   \@ctblbegin
      \ifx\@ctblcontinued\empty\else\addtocounter{\@ctbltaborfig}{-1}\fi
      \@ctblalign
      \begin{minipage}{\@ctblfloatwidth}\parindent0pt
         \ifx\@ctblbotcap\empty\@ctblCaption\vskip\@ctblcaptionskip\fi
         \centering{\usebox\ctbl@t} % insert the tabular
         \def\@ctblfootnotes{#3}%
         \ifx#3\empty\else{% append footnotes, if any
            \footnotesize
            \ifx\@ctblnotespar\empty%
               \\
               \begin{tabularx}{\hsize}{r@{\,}>{\raggedright}X}
                  #3%
               \end{tabularx}%
            \else%
               \\[.2ex]
               \begin{minipage}{\hsize}%
                  #3%
               \end{minipage}%
            \fi
         }
         \fi
         \ifx\@ctblbotcap\empty\else\vskip\@ctblcaptionskip\@ctblCaption\fi
      \end{minipage}
   \@ctblend
}
```

Change History