The \texttt{longtable} package\footnote{This file has version number v4.13, last revised 2020/01/07.}\footnote{\ They new algorithm for aligning ‘chunks’ of a table used in version 4 of this package was devised coded and documented by David Kastrup.}

David Carlisle\footnotemark[2]

2020/01/07

This file is maintained by the \LaTeX{} Project team.
Bug reports can be opened (category \texttt{tools}) at \url{https://latex-project.org/bugs.html}.

Abstract

This package defines the \texttt{longtable} environment, a multi-page version of \texttt{tabular}.

List of Tables

\begin{itemize}
\item[1.] An optional table caption (used in the list of tables) \hspace{1cm} 2
\item[2.] A floating table \hspace{1cm} 4
\item[3.] A difficult \texttt{\textbackslash multicolumn} combination: pass 1 \hspace{1cm} 6
\item[4.] A difficult \texttt{\textbackslash multicolumn} combination: pass 2 \hspace{1cm} 6
\item[5.] A difficult \texttt{\textbackslash multicolumn} combination: pass 3 \hspace{1cm} 6
\item[6.] A difficult \texttt{\textbackslash multicolumn} combination: pass 4 \hspace{1cm} 6
\item[7.] A summary of \texttt{longtable} commands \hspace{1cm} 9
\end{itemize}

1 Introduction

\texttt{longtable} The \texttt{longtable} package defines a new environment, \texttt{longtable}, which has most of the features of the \texttt{tabular} environment, but produces tables which may be broken by \TeX{}'s standard page-breaking algorithm. It also shares some features with the \texttt{table} environment. In particular it uses the same counter, \texttt{table}, and has a similar \texttt{\caption} command. Also, the standard \texttt{\textbackslash listoftables} command lists tables produced by either the \texttt{table} or \texttt{longtable} environments.

The following example uses most of the features of the \texttt{longtable} environment. An edited listing of the input for this example appears in Section 8.

\textbf{Note:} Various parts of the following table will \textbf{not} line up correctly until this document has been run through \LaTeX{} several times. This is a characteristic feature of this package, as described below.
Table 1: A long table

<table>
<thead>
<tr>
<th>FIRST</th>
<th>SECOND</th>
</tr>
</thead>
<tbody>
<tr>
<td>longtable</td>
<td>columns are specified</td>
</tr>
<tr>
<td>same way as in the tabular</td>
<td></td>
</tr>
<tr>
<td>@{(*)}r</td>
<td></td>
</tr>
<tr>
<td>Each row ends with a \</td>
<td></td>
</tr>
<tr>
<td>The \ command has an argument, just as in tabular</td>
<td></td>
</tr>
<tr>
<td>[\text{[10pt]} \</td>
<td></td>
</tr>
<tr>
<td>Lots of lines</td>
<td></td>
</tr>
<tr>
<td>Lots of lines</td>
<td></td>
</tr>
<tr>
<td>Lots of lines</td>
<td></td>
</tr>
<tr>
<td>Lots of lines</td>
<td></td>
</tr>
<tr>
<td>Lots of lines</td>
<td></td>
</tr>
<tr>
<td>Also \hline may be used, as in tabular.</td>
<td></td>
</tr>
<tr>
<td>That was a \hline</td>
<td></td>
</tr>
<tr>
<td>\hline \hline</td>
<td></td>
</tr>
<tr>
<td>\multicolumn{2}{</td>
<td></td>
</tr>
<tr>
<td>If a page break occurs at a \hline then a line is drawn at the bottom of one page and at the top of the next.</td>
<td></td>
</tr>
<tr>
<td>The [t] [b] [c] argument of \text{tabular} can not be used.</td>
<td></td>
</tr>
<tr>
<td>The optional argument may be one of [l] [r] [c] to specify whether the table should be adjusted or centrally.</td>
<td></td>
</tr>
<tr>
<td>Lots of lines</td>
<td></td>
</tr>
<tr>
<td>Lots of lines</td>
<td></td>
</tr>
<tr>
<td>Lots of lines</td>
<td></td>
</tr>
<tr>
<td>Lots of lines</td>
<td></td>
</tr>
<tr>
<td>Lots of lines</td>
<td></td>
</tr>
<tr>
<td>Lots of lines</td>
<td></td>
</tr>
<tr>
<td>Lots of lines</td>
<td></td>
</tr>
<tr>
<td>Lots of lines</td>
<td></td>
</tr>
<tr>
<td>Lots of lines</td>
<td></td>
</tr>
<tr>
<td>Lots of lines</td>
<td></td>
</tr>
<tr>
<td>Lots of lines</td>
<td></td>
</tr>
<tr>
<td>Lots of lines</td>
<td></td>
</tr>
<tr>
<td>Lots of lines</td>
<td></td>
</tr>
<tr>
<td>Lots of lines</td>
<td></td>
</tr>
<tr>
<td>Lots of lines</td>
<td></td>
</tr>
<tr>
<td>Lots of lines</td>
<td></td>
</tr>
<tr>
<td>Lots of lines</td>
<td></td>
</tr>
<tr>
<td>Lots of lines</td>
<td></td>
</tr>
<tr>
<td>Lots of lines</td>
<td></td>
</tr>
<tr>
<td>Lots of lines</td>
<td></td>
</tr>
<tr>
<td>Lots of lines</td>
<td></td>
</tr>
<tr>
<td>Lots of lines</td>
<td></td>
</tr>
<tr>
<td>Lots of lines</td>
<td></td>
</tr>
<tr>
<td>Lots of lines</td>
<td></td>
</tr>
<tr>
<td>Lots of lines</td>
<td></td>
</tr>
<tr>
<td>Lots of lines</td>
<td></td>
</tr>
<tr>
<td>Lots of lines</td>
<td></td>
</tr>
<tr>
<td>Lots of lines</td>
<td></td>
</tr>
<tr>
<td>Lots of lines</td>
<td></td>
</tr>
<tr>
<td>Lots of lines</td>
<td></td>
</tr>
</tbody>
</table>

This goes at the bottom.
Table 1: (continued)

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>This part appears at the top of every other page *</td>
</tr>
<tr>
<td>*</td>
<td>First</td>
</tr>
</tbody>
</table>

*Some lines may take up a lot of space, like this:*

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>Lots of lines like this. *</td>
</tr>
<tr>
<td>*</td>
<td>Lots of lines like this. *</td>
</tr>
<tr>
<td>*</td>
<td>Lots of lines like this. *</td>
</tr>
<tr>
<td>*</td>
<td>Lots of lines like this. *</td>
</tr>
<tr>
<td>*</td>
<td>Lots of lines like this. *</td>
</tr>
<tr>
<td>*</td>
<td>Lots of lines like this. *</td>
</tr>
<tr>
<td>*</td>
<td>Lots of lines like this. *</td>
</tr>
<tr>
<td>*</td>
<td>Lots of lines like this. *</td>
</tr>
<tr>
<td>*</td>
<td>Lots of lines like this. *</td>
</tr>
</tbody>
</table>

| * | Lots* of lines like this. * |
| * | Lots of lines like this like this* * |
| * | Lots of lines like this. * |
| * | Lots of lines like this. * |

| * | These lines will appear in place of the usual foot * |
| * | at the end of the table * |

2 Chunk Size

\texttt{LTchunksize}

In order to \TeX multi-page tables, it is necessary to break up the table into smaller chunks, so that \TeX does not have to keep everything in memory at one time. By default \texttt{longtable} uses 20 rows per chunk, but this can be set by the user, with e.g., \texttt{\setcounter{LTchunksize}{10}}. These chunks do not affect page breaking, thus if you are using a \TeX with a lot of memory, you can set \texttt{LTchunksize} to be several pages of the table. \TeX will run faster with a large \texttt{LTchunksize}.

\footnote{This is a footnote.}

\footnote{\texttt{longtable} takes special precautions, so that footnotes may also be used in ‘p’ columns.}

\footnote{You can also use the plain \TeX syntax \texttt{\LTchunksize=10}.}
A tabular environment within a floating table

Table 2: A floating table

However, if necessary, longtable can work with \LTchunksize set to 1, in which case the memory taken up is negligible. Note that if you use the commands for setting the table head or foot (see below), the \LTchunksize must be at least as large as the number of rows in each of the head or foot sections.

This document specifies \setcounter{LTchunksize}{10}. If you look at the previous table, after the first run of \LaTeX{} you will see that various parts of the table do not line up. \LaTeX{} will also have printed a warning that the column widths had changed. longtable writes information onto the .aux file, so that it can line up the different chunks. Prior to version 4 of this package, this information was not used unless a \setlongtables command was issued, however, now the information is always used, using a new algorithm\footnote{Due to David Kastrup.} and so \setlongtables is no longer needed. It is defined (but does nothing) for the benefit of old documents that use it.

\section{Captions and Headings}

At the start of the table one may specify lines which are to appear at the top of every page (under the headline, but before the other lines of the table). The lines are entered as normal, but the last \ command is replaced by a \endhead command. If the first page should have a different heading, then this should be entered in the same way, and terminated with the \endfirsthead command. The \LTchunksize should be at least as large as the number of rows in the heading.

There are also \endfoot and \endlastfoot commands which are used in the same way (at the start of the table) to specify rows (or an \hline) to appear at the bottom of each page. In certain situations, you may want to place lines which logically belong in the table body at the end of the firsthead, or the beginning of the lastfoot. This helps to control which lines appear on the first and last page of the table.

The \caption{...} command is essentially equivalent to \multicolumn{n}{c}{\parbox{\LTcapwidth}{...}} where \(n\) is the number of columns of the table. You may set the width of the caption with a command such as \setlength{\LTcapwidth}{2in} in the preamble of your document. The default is 4in. \caption also writes the information to produce an entry in the list of tables. As with the \caption command in the figure and table environments, an optional argument specifies the text to appear in the list of tables if this is different from the text to appear in the caption. Thus the caption for table 1 was specified as \caption[An optional table caption (used in the list of tables)]{A long table\label{long}}.

You may wish the caption on later pages to be different to that on the first page. In this case put the \caption command in the first heading, and put a subsidiary caption in a \caption\* command in the main heading. If the optional argument to \caption is empty, no entry is made in the list of tables. Alternatively, if
you do not want the table number to be printed each time, use the `\caption*` command.

The captions are set based on the code for the `article` class. If you have re-defined the standard `\makecaption` command to produce a different format for the captions, you may need to make similar changes to the `longtable` version, `\LT\makecaption`. See the code section for more details.

A more convenient method of customising captions is given by the `caption(2)` package, which provides commands for customising captions, and arranges that the captions in standard environments, and many environments provided by packages (including `longtable`) are modified in a compatible manner.

You may use the `\label` command so that you can cross reference `longtable`s with `\ref`. Note however, that the `\label` command should not be used in a heading that may appear more than once. Place it either in the `firsthead`, or in the body of the table. It should not be the `first` command in any entry.

## 4 Multicolumn entries

The `\multicolumn` command may be used in `longtable` in exactly the same way as for `tabular`. So you may want to skip this section, which is rather technical, however coping with `\multicolumn` is one of the main problems for an environment such as `longtable`. The main effect that a user will see is that certain combinations of `\multicolumn` entries will result in a document needing more runs of `\LaTeX` before the various ‘chunks’ of a table align.

The examples in this section are set with `LTchunksize` set to the minimum value of one, to demonstrate the effects when `\multicolumn` entries occur in different chunks.

Consider Table 3. In the second chunk, `longtable` sees the wide multicolumn entry. At this point it thinks that the first two columns are very narrow. All the width of the multicolumn entry is assumed to be in the third column. (This is a ‘feature’ of `\TeX`’s primitive `\halign` command.) `longtable` then passes the information that there is a wide column to the later chunks, with the result that the first pass over the table is too wide.

If the ‘saved row’ from this first pass was re-inserted into the table on the next pass, the table would line up in two passes, but would be much two wide.

The solution to this problem used in Versions 1 and 2, was to use a `\kill` line. If a line is `\kill`ed, by using `\kill` rather than `\` at the end of the line, it is used in calculating column widths, but removed from the final table. Thus entering `\kill`ed copies of the last two rows before the wide multicolumn entry would mean that `\halign` ‘saw’ the wide entries in the first two columns, and so would not widen the third column by so much to make room for the multicolumn entry.

In Version 3, a new solution was introduced. If the saved row in the `.aux` file was not being used, `longtable` used a special ‘draft’ form of `\multicolumn`, this modified the definition, so the spanning entry was never considered to be wider than the columns it spanned. So after the first pass, the `.aux` file stored the widest normal entry for each column, no column was widened due to `\spanned` columns. By default `longtable` ignored the `.aux` file, and so each run of `\LaTeX` was considered a first pass. Once the `\setlongtables` declaration was given, the saved row in the `.aux` file, and the proper definition of `\multicolumn` were...
Table 3: A difficult \texttt{\textbackslash multicolumn} combination: pass 1

\begin{tabular}{c|c|c}
1 & 2 & 3 \\
wide multicolumn spanning 1–3 & & \\
multicolumn 1–2 & 3 & \\
wide 1 & 2 & 3 \\
\end{tabular}

Table 4: A difficult \texttt{\textbackslash multicolumn} combination: pass 2

\begin{tabular}{c|c|c}
1 & 2 & 3 \\
wide multicolumn spanning 1–3 & & \\
multicolumn 1–2 & 3 & \\
wide 1 & 2 & 3 \\
\end{tabular}

Table 5: A difficult \texttt{\textbackslash multicolumn} combination: pass 3

\begin{tabular}{c|c|c}
1 & 2 & 3 \\
wide multicolumn spanning 1–3 & & \\
multicolumn 1–2 & 3 & \\
wide 1 & 2 & 3 \\
\end{tabular}

Table 6: A difficult \texttt{\textbackslash multicolumn} combination: pass 4

\begin{tabular}{c|c|c}
1 & 2 & 3 \\
wide multicolumn spanning 1–3 & & \\
multicolumn 1–2 & 3 & \\
wide 1 & 2 & 3 \\
\end{tabular}
used. If any `\multicolumn` entry caused one of the columns to be widened, this information could not be passed back to earlier chunks, and so the table would not correctly line up until the third pass. This algorithm always converged in three passes as described above, but in examples such as the ones in Tables 3–6, the final widths were not optimal as the width of column 2, which is determined by a `\multicolumn` entry was not known when the final width for column 3 was fixed, due to the fact that both `\multicolumn` commands were switched from ‘draft’ mode to ‘normal’ mode at the same time.

Version 4 alleviates the problem considerably. The first pass of the table will indeed have the third column much too wide. However, on the next pass `longtable` will notice the error and reduce the column width accordingly. If this has to propagate to chunks before the `\multicolumn` one, an additional pass will, of course, be needed. It is possible to construct tables where this rippling up of the correct widths takes several passes to ‘converge’ and produce a table with all chunks aligned. However in order to need many passes one needs to construct a table with many overlapping `\multicolumn` entries, all being wider than the natural widths of the columns they span, and all occurring in different chunks. In the typical case the algorithm will converge after three or four passes, and, the benefits of not needing to edit the document before the final run to add `\setlongtables`, and the better choice of final column widths in the case of multiple `\multicolumn` entries will hopefully more than pay for the extra passes that may possibly be needed.

So Table 3 converges after 4 passes, as seen in Table 6.

You can still speed the convergence by introducing judicious `\kill` lines, if you happen to have constellations like the above. If you object even to \TeX-ing a file twice, you should make the first line of every `longtable` a `\kill` line that contains the widest entry to be used in each column. All chunks will then line up on the first pass.

### 5 Adjustment

The optional argument of `longtable` controls the horizontal alignment of the table. The possible options are `[c]`, `[r]` and `[l]`, for centring, right and left adjustment, respectively. Normally centring is the default, but this document specifies

\begin{verbatim}
\setlength\LTleft\parindent
\setlength\LTright\fill
\end{verbatim}

in the preamble, which means that the tables are set flush left, but indented by the usual paragraph indentation. Any lengths can be specified for these two parameters, but at least one of them should be a rubber length so that it fills up the width of the page, unless rubber lengths are added between the columns using the `\extracolsep` command. For instance

\begin{verbatim}
\begin{tabular*}{\textwidth}{@{\extracolsep{...}}...}
\end{verbatim}

produces a full width table, to get a similar effect with `longtable` specify

\begin{verbatim}
\setlength\LTleft{0pt}
\setlength\LTright{0pt}
\begin{longtable}{@{\extracolsep{...}}...}
\end{verbatim}
6 Changes

This section highlights the major changes since version 2. A more detailed change log may be produced at the end of the code listing if the \ltxdoc.cfg file specifies

\AtBeginDocument{\RecordChanges}
\AtEndDocument{\PrintChanges}

Changes made between versions 2 and 3.

- The mechanism for adding the head and foot of the table has been completely rewritten. With this new mechanism, longtable does not need to issue a \clearpage at the start of the table, and so the table may start half way down a page. Also the \endlastfoot command which could not safely be implemented under the old scheme, has been added.

- longtable now issues an error if started in the scope of \twocolumn, or the \multicols environment.

- The separate documentation file longtable.tex has been merged with the package file, longtable.dtx using Mittelbach's doc package.

- Support for footnotes has been added. Note however that \footnote will not work in the ‘head’ or ‘foot’ sections of the table. In order to put a footnote in those sections (e.g., inside a caption), use \footnotemark at that point, and \footnotetext anywhere in the table body that will fall on the same page.

- The treatment of \multicolumn has changed, making \kill lines unnecessary, at the price of sometimes requiring a third pass through \LaTeX.

- The \newpage command now works inside a longtable.

Changes made between versions 3 and 4.

- A new algorithm is used for aligning chunks. As well as the widest width in each column, longtable remembers which chunk produced this maximum. This allows it to check that the maximum is still achieved in later runs. As longtable can now deal with columns shrinking as the file is edited, the \setlongtables system is no longer needed and is disabled.

- An extra benefit of the new algorithm's ability to deal with ‘shrinking’ columns is that it can give better (narrower) column widths in the case of overlapping \multicolumn entries in different chunks than the previous algorithm produced.

- The ‘draft’ multicol system has been removed, along with related commands such as \LTmulticolumn.

- The disadvantage of the new algorithm is that it can take more passes. The theoretical maximum is approximately twice the length of a ‘chain’ of columns with overlapping \multicolumn entries, although in practice it usually converges as fast as the old version. (Which always converged in three passes once \setlongtables was activated.)

- \* and \nopagebreak commands may be used to control page breaking.
## 7 Summary

Table 7: A summary of `longtable` commands

### Parameters

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\LTleft</td>
<td>Glue to the left of the table.</td>
</tr>
<tr>
<td>\LTright</td>
<td>Glue to the right of the table.</td>
</tr>
<tr>
<td>\LTpre</td>
<td>Glue before the table.</td>
</tr>
<tr>
<td>\LTpost</td>
<td>Glue after the table.</td>
</tr>
<tr>
<td>\LTcapwidth</td>
<td>The width of a parbox containing the caption. (4in)</td>
</tr>
<tr>
<td>LTchunksize</td>
<td>The number of rows per chunk.</td>
</tr>
</tbody>
</table>

### Optional arguments to `\begin{longtable}`

- `none`: Position as specified by \LTleft and \LTright.
- `[c]`: Centre the table.
- `[l]`: Place the table flush left.
- `[r]`: Place the table flush right.

### Commands to end table rows

- `\\`: Specifies the end of a row
- `\\[⟨dim⟩]`: Ends row, then adds vertical space (as in the tabular environment).
- `\\*`: The same as `\\` but disallows a page break after the row.
- `\tabularnewline`: Alternative to `\\` for use in the scope of \raggedright and similar commands that redefine `\\`.
- `\kill`: Row is ‘killed’, but is used in calculating widths.
- `\endhead`: Specifies rows to appear at the top of every page.
- `\endfirsthead`: Specifies rows to appear at the top the first page.
- `\endfoot`: Specifies rows to appear at the bottom of every page.
- `\endlastfoot`: Specifies rows to appear at the bottom of the last page.

### Longtable caption commands

- `\caption{⟨caption⟩}`: Caption ‘Table ?: ⟨caption⟩’, and a ‘⟨caption⟩’ entry in the list of tables.
- `\caption[⟨lot⟩]{⟨caption⟩}`: Caption ‘Table ?: ⟨caption⟩’, and a ‘⟨lot⟩’ entry in the list of tables.
- `\caption[]{⟨caption⟩}`: Caption ‘Table ?: ⟨caption⟩’, but no entry in the list of tables.
- `\caption*{⟨caption⟩}`: Caption ‘⟨caption⟩’, but no entry in the list of tables.

### Commands available at the start of a row

- `\pagebreak`: Force a page break.
- `\pagebreak[⟨val⟩]`: A ‘hint’ between 0 and 4 of the desirability of a break.
- `\nopagebreak`: Prohibit a page break.
- `\nopagebreak[⟨val⟩]`: A ‘hint’ between 0 and 4 of the undesirability of a break.
- `\newpage`: Force a page break.

### Footnote commands available inside `longtable`

- `\footnote`: Footnotes, but may not be used in the table head & foot.
- `\footnotemark`: Footnotemark, may be used in the table head & foot.
- `\footnotetext`: Footnote text, use in the table body.

### Setlongtables

- `\setlongtables`: Obsolete command. Does nothing now.
8 Verbatim highlights from Table 1

\begin{longtable}{@{*}r||p{1in}@{*}}
KILLED & LINE!!!! \kill
\caption[An optional table caption ...]{A long table\label{long}}\\\n\hline\hline
\multicolumn{2}{@{*}c@{*}}{This part appears at the top of the table}\\\n\textsc{First}&\textsc{Second}\\
\hline\hline
\endfirsthead\caption[]{(continued)}\\\n\hline\hline
\multicolumn{2}{@{*}c@{*}}{This part appears at the top of every other page}\\\n\textbf{First}&\textbf{Second}\\
\hline\hline
\endhead\hline
This goes at the bottom.\\
\hline\hline
These lines will appear\\\nin place of the & usual foot\\nat the end of the table\\
\hline\hline
\endlastfoot
\env{longtable} columns are specified in the same way as in the \env{tabular} environment.\\
\multicolumn{2}{||c||}{This is a ...}\\
Some lines may take...&
\raggedlast This last column is a ‘‘p’’ column...
\tabularnewline

Lots of lines\footnote{...} of lines\footnote{...} like this.\\

Lots of lines\footnote{...} like this.\\
\hline
Lots of lines\footnote{...} like this.\\

\end{longtable}
9 The Macros

9.1 Initial code

Before declaring the package options, we must define some defaults here.

\LT@err The error generating command
1 \def\LT@err{\PackageError{longtable}}
\LT@warn The warning generating command
2 \def\LT@warn{\PackageWarning{longtable}}
\LT@final@warn If any longtables have not aligned, generate a warning at the end of the run at \AtEndDocument.
3 \def\LT@final@warn{\AtEndDocument{\LT@warn{Table widths have changed. Rerun LaTeX. \@gobbletwo}}}
4 \global\let\LT@final@warn\relax

9.2 Options

The first two options deal with error handling. They are compatible with the options used by the tracefnt package.

errorshow Only show errors on the terminal. ‘warnings’ are just sent to the log file.
5 \DeclareOption{errorshow}{\def\LT@warn{\PackageInfo{longtable}}}
6 \DeclareOption{pausing}{\def\LT@warn#1{\LT@err{#1}{This is not really an error}}}
7 \ProcessOptions

9.3 User Settable Parameters

\LTleft Glue to the left and right of the table, default \fill (ie centred).
8 \newskip\LTleft \LTleft=\fill
9 \newskip\LTright \LTright=\fill
\LTpre Glue before and after the longtable. \bigskip by default.
10 \newskip\LTpre \LTpre=\bigskipamount
11 \newskip\LTpost \LTpost=\bigskipamount
\LTchunksize Chunk size (The number of rows taken per \halign). Default 20.
12 \newcount\LTchunksize \LTchunksize=20

\LTleft \LTright \LTpre \LTpost \LTchunksize

................. longtable.sty ........................
\c@LTchunksize Added in V3.07 to allow the \LaTeX{} syntax \texttt{\setcounter{LTchunksize}{10}}.
21 \let\c@LTchunksize\LTchunksize
\LTcapwidth Width of the \texttt{\parbox} containing the caption. Default 4in.
22 \newdimen\LTcapwidth \LTcapwidth=4in

9.4 Internal Parameters
\LT@head Boxes for the table head and foot.
23 \newbox\LT@head
\LT@firsthead
24 \newbox\LT@firsthead
\LT@foot
25 \newbox\LT@foot
26 \newbox\LT@lastfoot
\LT@cols Counter for number of columns.
27 \newcount\LT@cols
\LT@rows Counter for rows up to chunksize.
28 \newcount\LT@rows
\c@LT@tables Counter for the tables, added in V3.02. Previous versions just used the \LaTeX{} counter \texttt{table}, but this fails if \texttt{table} is reset during a document, eg \texttt{report} class resets it every chapter.
This was changed from \texttt{\newcount\LT@tables} in V3.04. \LaTeX{} counters are preserved correctly when \texttt{\includeonly} is used. In the rest of the file \texttt{\LT@tables} has been replaced by \texttt{\c@LT@tables} without further comment.
29 \newcounter{LT@tables}
\c@LT@chunks We need to count through the chunks of our tables from Version 4 on.
30 \newcounter{LT@chunks}[\LT@tables]
\c@table If the \texttt{table} counter is not defined (eg in \texttt{letter} style), define it. (Added in V3.06.)
31 \ifx\c@table\undefined
32 \newcounter{table}
33 \def\fnum@table{\tablename\thetable}
34 \fi
35 \ifx\tablename\undefined
36 \def\tablename{Table}
37 \fi
\LT@out In a normal style, \texttt{longtable} uses the \texttt{.aux} file to record the column widths. With \texttt{letter.sty}, use a separate \texttt{.lta} file. (Added in V3.06.) Not needed for new letter class.
38 \ifx\startlabels\undefined
39 \let\@auxout\@auxout
40 \else
41 \% \texttt{\input{\jobname.lta}}\%
42 \newwrite\@auxout
43 \immediate\openout\@auxout=\jobname.lta
44 \fi
9.5 The \texttt{longtable} environment

\texttt{longtable} Called by \texttt{\begin{longtable}}. This implementation does not work in multiple column formats. \texttt{\par} added at V3.04.

\texttt{\def longtable{%}
\par
\ifx\multicols@undefined
\else
\ifnum\col@number>1
@twocolumntrue
\fi
\fi
\if@twocolumn
\LT@end@pen{longtable not in 1-column mode}\@ehc
\fi
\fi
\begingroup
\Check for an optional argument.
\@ifnextchar[\LT@array{\LT@array[\]}]

\texttt{\LT@array Start setting the alignment. Based on \texttt{\@array} from the \LaTeX\ kernel and the array package.}

Since Version 3.02, longtable has used the internal counter \texttt{\c@LT@tables}. The \LaTeX\ counter \texttt{table} is still incremented so that \texttt{\caption} works correctly.

\texttt{\def\LT@array[#1]#2{%}
\refstepcounter{table}\stepcounter{LT@tables}%
\texttt{Set up the glue around the table if an optional argument given.}
\if l#1%
\LTleft\z@ \LTright\fill
\else\if r#1%
\LTleft\fill \LTright\z@
\else\if c#1%
\LTleft\fill \LTright\fill
\fi\fi\fi
\LT@end@pen{longtable not in 1-column mode}\@ehc
\fi
\fi
\begingroup
\Check for an optional argument.
\@ifnextchar[\LT@array{\LT@array[\]}]

\texttt{\LT@@tabarray} Start setting the alignment. Based on \texttt{\@array} from the \LaTeX\ kernel and the array package.

Since Version 3.02, longtable has used the internal counter \texttt{\c@LT@tables}. The \LaTeX\ counter \texttt{table} is still incremented so that \texttt{\caption} works correctly.

\texttt{\def\LT@array[#1]#2{%}
\refstepcounter{table}\stepcounter{LT@tables}%
\texttt{Set up the glue around the table if an optional argument given.}
\if l#1%
\LTleft\z@ \LTright\fill
\else\if r#1%
\LTleft\fill \LTright\z@
\else\if c#1%
\LTleft\fill \LTright\fill
\fi\fi\fi
\LT@end@pen{longtable not in 1-column mode}\@ehc
\fi
\fi
\begingroup
\Check for an optional argument.
\@ifnextchar[\LT@array{\LT@array[\]}]
More or less standard definitions, but first start a \noalign.\pagebreak{\noalign{\ifnum'}=0\fi\@testopt{LT@no@pgbk-}4} %
\def\nopagebreak{\noalign{\ifnum'}=0\fi\@testopt\LT@no@pgbk4} %
\let\hline\LT@hline \let\kill\LT@kill \let\caption\LT@caption
\@tempdima\ht\strutbox
\let\@endpbox\LT@endpbox
Set up internal commands according to Lamport or Mittelbach.
\ifx\extrarowheight\@undefined
Initialise these commands as in tabular from the LATEX kernel.
\let\@acol\@tabacol
\let\@classz\@tabclassz \let\@classiv\@tabclassiv
\def\@startpbox{\vtop\LT@startpbox} %
\let\@@startpbox\@startpbox
\let\@@endpbox\@endpbox
\let\LT@LL@FM@cr\@tabularcr
\else
Initialise these commands as in array. \$ replaced by \$begin
\$end in V3.03 to match array V2.0h. We do not need to set \$begin
and \$end as the array package gives them the correct values at the top
level.
\advance\@tempdima\extrarowheight %
\col@sep\tabcolsep %
\let\@startpbox\LT@startpbox\let\LT@LL@FM@cr\@arraycr
\fi
The rest of this macro is mainly based on array package, but should work for the
standard tabular too.
\setbox\@arstrutbox\vbox{\hrule %
\@height \arraystretch \@tempdima
\@depth \arraystretch \@tempdima
\@width \z@}
\let\@sharp##\let\protect\relax
Interpret the preamble argument.
\begingroup
\@mkpream{#2} %
We need to rename \@preamble here as F.M.'s scheme uses \global, and we may
need to nest \@mkpream, eg for \multicolumn or an array. We do not need to
worry about nested longtables though!
\def\LT@bchunk{\global\advance\c@LT@chunks\@ne
\global\LT@rows\z@\setbox\z@\vbox{bgroup
}
The following line was added in v4.05. In order to get the \penalties to work at
cunk boundaries Need to take more care about where and when \lineskip glue
is added. The following does nothing at top of table, and in header chunks, but in
ormal body chunks it sets \prevdepth (to 0pt, but any value would do) so that
\lineskip glue will be added. the important thing to note is that the glue will
be added after any vertical material coming from \noalign.

\LT@setprevdepth
\tabskip\LTleft \noexpand\halign to\hsize\bgroup
\% \tabskip\LTleft\halign to\hsize\bgroup
\tabskip\z@ \@arstrut \@preamble \tabskip\LTRight \cr%}
\endgroup

Find out how many columns we have (store in \LT@cols).
\expandafter\LT@nofcols\LT@bchunk&\LT@nofcols
Get the saved row from \LT@i...\LT@ix (from the .aux file), or make a new blank
row.
\LT@make@row
A few more internal commands for longtable.
\m@th\let\par\@empty
\everycr{}\lineskip\z@\baselineskip\z@
Start the first chunk.
\LT@bchunk}
\LT@no@pgbk Can simplify the standard \@no@pgbk as this is vmode only but then need to close
the \noalign.
\def\LT@no@pgbk#1[#2]{\penalty #1\@getpen{#2}\ifnum'={0\fi}}
\LT@start This macro starts the process of putting the table on the current page. It is not
called until either a \ or \endlongtable command ends a chunk, as we do not
know until that point which of the four possible head or foot sections have been
specified.
It begins by redefining itself, so that the table is only started once! Until V3.04,
was redefined to \relax, now use \endgraf to force the page-breaker to wake up.
The second \endgraf is there so that \pagetotal is updated and so takes \LTpre
into account.
\def\LT@start{%
\let\LT@start\endgraf
\endgraf\penalty\z@\vskip\LTpre\endgraf
Start a new page if there is not enough room for the table head, foot, and one
extra line.
\dimen@\pagetotal
\advance\dimen@ \ht\ifvoid\LT@firsthead\LT@head\else\LT@firsthead\fi
\advance\dimen@ \dp\ifvoid\LT@firsthead\LT@head\else\LT@firsthead\fi
\advance\dimen@ \ht\LT@foot
At this point I used to add \ht\@arstrutbox and \dp\@arstrutbox as a measure
of a row size. However this can fail spectacularly for p columns which might be
much larger. Previous versions could end up with the table starting with a foot,
then a page break then a head then a ‘first head’! So now measure the first line
of the table accurately by \vsplitting it out of the first chunk.
\dimen@ii \vfuzz
\vfuzz \maxdimen
\setbox\tw@ \copy \z@
\setbox\tw@ \vsplit \tw@ to \ht \arstrutbox
\setbox\tw@ \vbox { \unvbox \tw@ } \%
\vfuzz \dimen@ii
\advance \dimen@ \ht
\ifdim \ht \arstrutbox > \ht \tw@ \@arstrutbox \else \tw@ \fi
\advance \dimen@ \dp
\ifdim \dp \arstrutbox > \dp \tw@ \@arstrutbox \else \tw@ \fi
\advance \dimen@ - \pagegoal
\ifdim \dimen@ > \z@ \vfil \break \fi
Store height of page minus table foot in @colroom.
\global @colroom \@colht
If the foot is non empty, reduce the \vsize and @colroom accordingly.
\ifvoid \LT@foot \else
\advance \vsize \- \ht \LT@foot
\global \advance \@colroom \- \ht \LT@foot
\dimen@ \pagegoal \advance \dimen@ \- \ht \LT@foot \pagegoal \dimen@
\maxdepth \z@
\fi
Put the table head on the page, and then switch to the new output routine.
\ifvoid \LT@firsthead \copy \LT@head \else \box \LT@firsthead \fi \nobreak
\output { \LT@output }
\endlongtable Called by \end{longtable}.
\def \endlongtable { \%
Essentially add a final \textbackslash \textbackslash . But as we now know the number of actual chunks, we first strip away all entries referring to a maximum entry beyond the table (this can only happen if a table has been shortened, or the table numbering has gone awry). In that case we at least start collecting valid new information with the last chunk of this table, by removing the width constraint.
\crcr
\noalign { 
\let \LT@entry \LT@entry@chop
\xdef \LT@save@row { \LT@save@row }
\LT@echunk
\LT@start
\unvbox \z@
\LT@get@widths
Write the dummy row to the .aux file. Since V3.06, use .lta for letter.sty.
\if@filesw
{ \let \LT@entry \LT@entry@write \immediate \write \@auxout { %
Since Version 3.02, longtable has used the internal counter \c@LT@tables rather than the \LaTeX counter table. This information looks entirely different from version 3 information. Still, we don’t need to rename the macro name because later code will consider the information to have no columns, and thus will throw the old data away.
\gdef \expandafter \noexpand \endlongtable {
\endlongtable}
At this point used to issue a warning if a \multicolumn has been set in draft mode.

\LT@mcw@rn

If the last chunk has different widths than the first, warn the user. Also trigger a warning to rerun \LaTeX at the end of the document.

\ifx\LT@save@row\LT@@save@row
\else
\LT@warn{Column \@width s have changed\MessageBreak in table \thetable}%
\LT@final@warn
\fi

Force one more go with the longtable output routine.
\endgraf\penalty -\LT@end@pen

Now close the group to return to the standard routine.
\endgroup

Reset \@mparbottom to allow marginpars close to the end of the table.\footnote{This can not be the correct. However if it is omitted, there is a problem with marginpars, for example on page 3 of this document. Any Output Routine Gurus out there?}

\LT@nofcols Find the next \&, then look ahead to see what is next.
\begin{Verbatim}
def\LT@nofcols\#1\&{%
\futurelet\@let@token\LT@n@fcols
\end{Verbatim}

\LT@n@fcols Add one, then stop at an \LT@nofcols or look for the next \&. The \expandafter trick was added in Version 3, also the name changed from \@LT@nofcols to preserve the \LT@ naming convention.
\begin{Verbatim}
def\LT@nofcols\%
\advance\LT@cols\@ne
\ifx\@let@token\LT@nofcols
\fi
\end{Verbatim}

9.6 Counting Columns

Columns are counted by examining \@preamble, rather than simply getting \@mkpream to increment the counter as it builds the preamble so that this package works with many of the packages which add extra column specifiers to \LaTeX's standard ones.

Version 1 counted \@sharp's to calculate the number of columns, this was changed for Version 2 as it does not work with the NFSS. Now count \&'s. (lfonts.new (and now the Standard \LaTeX definition) defines \@tabclassz so that \@sharp is inside a group.)

\LT@nofcols Find the next \&, then look ahead to see what is next.
\begin{Verbatim}
def\LT@nofcols\#1\&{%
\futurelet\@let@token\LT@n@fcols
\end{Verbatim}

\LT@n@fcols Add one, then stop at an \LT@nofcols or look for the next \&. The \expandafter trick was added in Version 3, also the name changed from \@LT@nofcols to preserve the \LT@ naming convention.
\begin{Verbatim}
def\LT@nofcols\%
\advance\LT@cols\@ne
\ifx\@let@token\LT@nofcols
\fi
\end{Verbatim}
9.7 The « and \kill Commands

The internal definition of `. In the * form, insert a \nobreak after the next \cr (or \crcr).

This star form processing was finally added in v4.05. For the previous six or seven years the comment at this point said

This definition also accepts `\*', which acts in the same way as `. tabular does this, but `longtable' probably ought to make `\*' prevent page breaking.

{\ifnum0='}fi added in version 3.01, required if the first entry is empty.

The above in fact is not good enough, as with `array' package it can introduce a {\} group in math mode, which changes the spacing. So use the following variant. Added in v3.14.

\def\LT@tabularcr{%
\relax\iffalse{\fi\ifnum0='}\fi
\@ifstar{%\def\crcr{\LT@crcr\noalign{\nobreak}}\let\cr\crcr
\LT@t@bularcr}%\LT@t@bularcr}%

\LT@crcr
\let\LT@crcr\crcr

\LT@setprevdepth This will be redefined to set the \prevdepth at the start of a chunk.

\let\LT@setprevdepth\relax

\LT@t@bularcr
\def\LT@t@bularcr{%
Increment the counter, and do \tabular’s `. or finish the chunk.

The \expandafter trick was added in Version 3. Set the \prevdepth at the start of a new chunk. (Done here so not set in header chunks).

\global\advance\LT@rows\@ne
\@ifnum\LT@rows=\LTchunksize
{\gdef\LT@setprevdepth{%\prevdepth\z@\global
\LT@setprevdepth\relax}%
\expandafter\LT@xtabularcr
\else
{\ifnum0='}fi
\expandafter\LT@LL@FM@cr
\fi}%

\LT@xtabularcr This just looks for an optional argument.

\def\LT@xtabularcr{%
\@ifnextchar[\LT@argtabularcr\LT@ntabularcr}

\ifnum0='}fi added in version 3.01, required if the first entry is empty.

The above in fact is not good enough, as with `array' package it can introduce a {\} group in math mode, which changes the spacing. So use the following variant. Added in v3.14.
\LT@ntabularcr The version with no optional argument. \texttt{\ifnum0='{\fi} added in version 3.01.
Changed in 3.14.
193 \def\LT@ntabularcr{% 194  \ifnum0='{}\fi 195  \LT@echunk 196  \LT@start 197  \unvbox\z@ 198  \LT@get@widths 199  \LT@bchunk}
\LT@argtabularcr The version with an optional argument. \texttt{\ifnum0='{\fi} added in version 3.01.
Changed in 3.14.
200 \def\LT@argtabularcr[#1] {% 201  \ifnum0='{}\fi 202  \ifdim #1>\z@ 203   \unskip\@xargarraycr{#1} 204  \else 205   \@yargarraycr{#1} 206  \fi
Add the dummy row, and finish the \texttt{\halign}.
207  \LT@echunk 208  \LT@start 209  \unvbox\z@ 210  \LT@get@widths 211  \LT@bchunk}
\LT@echunk This ends the current chunk, and removes the dummy row.
212 \def\LT@echunk{% 213  \crcr\LT@save@row\cr\egroup 214  \global\setbox\@ne\lastbox
The following line was added in v4.05. \texttt{longtable} relies on \texttt{\lineskip} glue (which
is 0pt) to provide break points between each row so the table may be split into
pages. Previous releases left the \texttt{\lineskip} glue at the end of each chunk that had
been added when the dummy row was added. There was no glue at the start of
the next chunk as \TeX normally does not put \texttt{\lineskip} glue at the top of a box.
This meant that normally the chunks fitted together perfectly, however \texttt{\noalign}
material at a chunk boundary came before the first row of the next chunk but
after the \texttt{\lineskip} glue at the end of this chunk. This is the wrong place, e.g.,
it means even a \texttt{\penalty10000} does not stop a break as the \texttt{\lineskip} glue
in the previous item on the list provides a legal breakpoint. So now remove the
\texttt{\lineskip} glue that was before the dummy row and introduce \texttt{\LT@setprevdepth}
to set the \texttt{\prevdepth} at the start of the next chunk, to make sure \texttt{\lineskip}
glue is added later.
215 \unskip 216 \egroup
\LT@entry We here give the ‘basic’ definition of \texttt{\LT@entry}, namely that used in alignment
templates. It has a \texttt{\kern} only if the maximum is imposed from a different chunk.
The \texttt{\ifhmode} test reveals the first entry, when we don’t want to add an \&.
217 \def\LT@entry[#1] {%
This definition for the argument of \LTsaveRow is used to scrap all those maxima which could not be verified because they occur after the end of the table. This can happen only if a table has been shortened (or the sequencing got mixed up) since the previous run. Note that this is premature: the last chunk still is going to be set, and with the chopped limits.

To write an entry for the aux file, we use a slightly surprising definition which has the sole purpose of avoiding overfull lines (which might break \TeX’s limits when reading the aux file, probably you’d need to have a few hundred columns before this happened but...).

This ends the current chunk as above, but strips off two rows, the ‘dummy row’ and the ‘killed row’ before starting the next chunk. Since V3.04, the old chunk is reboxed at the start of the box containing the next chunk. This allows \kill to be used in headers, which must be processed in a single box.

Drop the old chunk (box0) back at the top of the new chunk, removing the killed row. This macro added at V3.04.

Create a blank row if we are not using the info in the .aux file.

9.8 The Dummy Row
The dummy row is kept inside of the macro \LTsaveRow.
Whoops! What’s that supposed to be? A drop-in replacement for the first task of Appendix D in the TeXbook. The \texttt{\textbackslash romannumeral} produces \texttt{\LT@cols} instances of \texttt{m} followed by \texttt{i}. The below macro then replaces the \texttt{ms} by appropriate entries.

\begin{verbatim}
\def\LT@build@blank#1{\%
  \if#1m\%
  \noexpand\LT@entry{1}{0pt}\%
  \expandafter\LT@build@blank\fi}
\end{verbatim}

\texttt{\LT@make@row} Prior to version 4, by default did not use information in the .aux file but now we can define \texttt{\LT@make@row} to use the .aux file, even on the ‘draft’ passes.

\begin{verbatim}
\def\LT@make@row{\%
  \global\expandafter\let\expandafter\LT@save@row
  \csname LT@\romannumeral\c@LT@tables\endcsname
  \ifx\LT@save@row\relax
  \LT@blank@row
  \else\%
  \if!\%
  \ifcase\expandafter\expandafter\expandafter\LT@cols\%
  \expandafter\@gobble\LT@save@row\%
  \or\%
  \else\%
  \relax\%
  \fi\%
  \else\%
  \aftergroup\LT@blank@row\%
  \fi\%
  \fi\%
\end{verbatim}

\texttt{\setlongtables} Redefine \texttt{\LT@make@row} to use information in the .aux file, if there is a saved row for this table with the right number of columns.

Since Version 3.02, \texttt{longtable} has used the internal counter \texttt{\c@LT@tables} rather than the \texttt{\LaTeX} counter \texttt{table}. The warning message was added at V3.04, as was the \texttt{\global}, to stop save-stack overflow.

Since Version 4.01 \texttt{\setlongtables} does nothing as it is not needed, but is defined as \texttt{\relax} for the benefit of old documents.

\begin{verbatim}
\let\setlongtables\relax
\end{verbatim}

\texttt{\LT@get@widths} This is the heart of \texttt{longtable}. If it were not for the table head and foot, this macro together with the modified \texttt{\LaTeX} command would form the basis of quite a simple little package file for long tables. It is closely modelled on the \texttt{\endvrulealign} macro of appendix D of the \texttt{\LaTeX}book.

\begin{verbatim}
\def\LT@get@widths{\%
\end{verbatim}
\global added at V3.04, to stop save-stack overflow.

Loop through the last row, discarding glue, and saving box widths. At V3.04 changed the scratch box to 2, as the new \kill requires that \box0 be preserved.

\setbox\tw@\hbox{%
\unhbox\@ne
\let\LT@old@row\LT@save@row
\global\let\LT@save@row\@empty
\count@\LT@cols
\loop
\unskip
\setbox\tw@\lastbox
\ifhbox\tw@
\LT@def@row
\advance\count@\m@ne
\repeat}
Remember the widths if we are in the first chunk.
\ifx\LT@@save@row\@undefined
\let\LT@@save@row\LT@save@row
\fi}
\LT@def@row Add a column to the dummy row. Name changed from \def\LT@save@row in Version 3, to preserve the \LT@ naming convention.
\def\LT@def@row{% We start by picking the respective entry from our old row. These redefinitions of \LT@entry are local to the group started in \LT@get@widths.
\let\LT@entry\or
\edef\@tempa{%\ifcase\expandafter\count@\LT@old@row
\else
{1}{0pt}%
\fi}%
Now we tack the right combination in front of \LT@save@row:
\let\LT@entry\relax
\xdef\LT@save@row{% \LT@def@row
\expandafter\LT@max@sel\@tempa
\LT@save@row}}
\LT@max@sel And this is how to select the right combination. Note that we take the old maximum information only if the size does not change in either direction. If the size has grown, we of course have a new maximum. If the size has shrunk, the old maximum (which was explicitly not enforced because of being in the current chunk) is invalid, and we start with this chunk as the new size. Note that even in the case of equality we must use the \the\wd\tw@ construct instead of \#2 because \#2 might be read in from the file, and so could have \catcode 11 versions of p and t in it which we want to be replaced by their ‘proper’ \catcode 12 versions.
\def\LT@max@sel@#2{% \ifdim#2=\wd\tw@ #1% \else
9.9 The \hline Command

\LT@hline \hline and \hline\hline both produce two lines. The only difference being the glue and penalties between them. This is so that a page break at a \hline produces a line on both pages. Also this \hline is more like a \cline{1-\LT@cols}. \tabular’s \hline would draw lines the full width of the page.

\LT@@hline This code is based on \cline. Two copies of the line are produced, as described above.

9.10 Captions

\LT@caption The caption is \multicolumn{\LT@cols}{c}{{\parbox{\linewidth}{\caption{\protect\parbox\captionfnum\captionfirst}{\captionfnum\captionfnum}\captionfnum\captionfnum}{\captionfnum\captionfnum}}}\captionfnum

\LT@cOption Caption command (with [optional argument]). \protect added in Version 3. \fnum@table added at V3.05.

\LT@cOption#1[#2]#3 The \LT@cOption has always done this, but perhaps it would be better if hlines were omitted at a page break, as the head and foot usually put a hline here anyway.
\LT@capti@on
Caption command (no [optional argument])
\def\LT@capti@on{% 
\@ifstar
{"\egroup\LT@c@ption\@gobble\[\]%}%
{"\egroup\@xdblarg{\LT@c@ption\@firstofone}}}

\LT@makecaption
Put the caption in a box of width 0pt, so that it never affects the column widths. Inside that is a \parbox of width \LTcapwidth.
\def\LT@makecaption#1#2#3{% 
\LT@mcol\LT@cols c{\hbox to\z@{\hss\parbox[t]\LTcapwidth{#1\ifdim\wd\@tempboxa \hsize \else \hbox to\hsize{\hfil\box\@tempboxa\hfil}\fi\endgraf\vskip\baselineskip}}}% 
\hss}}}

9.11 The Output Routine
The method used here for interfacing a special purpose output routine to the standard \LaTEX routine is lifted straight out of F. Mittelbach’s multicol package.

\LT@output
Actually this is not so bad, with FM leading the way.
\def\LT@output{% 
\ifnum\outputpenalty < -\@Mi 
\ifnum\outputpenalty > -\LT@end@pen
If this was a float or a marginpar we complain.
\LT@err{floats and marginpars not allowed in a longtable}\@ehc
\else
We have reached the end of the table, on the scroll at least,
\setbox\z@\vbox{\unvbox\@cclv}\ifdim \ht\LT@lastfoot > \ht\LT@foot
The last foot might not fit, so:\footnote{An alternative would be to vsplit off a bit of the last chunk, so that the last page did not just have head and foot sections, but it is hard to do this in a consistent manner.} 
\dimen@\pagegoal 
\advance\dimen@ -\ht\LT@lastfoot 
\ifdim\dimen@ < \ht\z@ 
\setbox\z@\vbox{\unvbox\z@\copy\LT@foot\vss}\@makecol
\@outputpage
\setbox\z@\vbox{\box\LT@head}\fi
\@cclv}}

End of \ifdim\dimen@ < \ht\@cclv.
End of \ifdim \ht\LT@lastfoot > \ht\LT@foot. \fi
Reset \@colroom.
\global\@colroom\@colht
\global\vsize\@colht
Put the last page of the table on to the main vertical list.
\vbox \{
\unvbox\z@\box\ifvoid\LT@lastfoot\LT@foot\else\LT@lastfoot\fi\%
\endgraf
\LT@end@hd@ft #1{
\LT@echunk
Changed from \relax to \endgraf at V3.04, see \LT@start.
\ifx\LT@start\endgraf
\LT@err
\fi
\setbox#1\box\z@
\LT@get@widths
\LT@bchunk}
\endfirsthead
The core of \endhead and friends. Store the current chunk in the box specified by #1. Issue an error if the table has already started. Then start a new chunk.
\def\LT@end@hd@ft#1{
\LT@echunk
Changed from \relax to \endgraf at V3.04, see \LT@start.
\ifx\LT@start\endgraf
\LT@err
\fi
\setbox#1\box\z@
\LT@get@widths
\LT@bchunk}
\endhead
Call \LT@end@hd@f with the appropriate box.
\endhead
\endfoot
\endlastfoot
9.12 Commands for the table head and foot
9.13 The `\multicolumn` command

Earlier versions needed a special ‘draft’ form of `\multicolumn`. This is not needed in version 4, and so these commands have been removed.

\LTmulticolumn
\LT@mcwarn

9.14 Footnotes

The standard `\footnote` command works in a c column, but we need to modify the definition in a p column to overcome the extra level of boxing. These macros are based on the `array` package, but should be OK for the standard `tabular`.

\LT@startpbox Add extra code to switch the definition of `\@footnotetext`.
\LT@endpbox After the parbox is closed, expand `\LT@p@ftntext` which will execute a series of `\footnotetext{⟨num⟩}{⟨note⟩}` commands. After being lifted out of the parbox, they can migrate on their own from here.

\LT@p@ftntext Inside the ‘p’ column, just save up the footnote text in a token register.