The \texttt{longtable} package\footnote{This file has version number v4.09, last revised 1998/05/13.}

David Carlisle\footnote{The new algorithm for aligning ‘chunks’ of a table used in version 4 of this package was devised coded and documented by David Kastrup, dak@neuroinformatik.ruhr-uni-bochum.de.}

1998/05/13

Abstract

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

List of Tables

\begin{tabular}{ll}
1 & An optional table caption (used in the list of tables) \\
2 & A floating table \\
3 & A difficult \texttt{\textbackslash multicolumn} combination: pass 1 \\
4 & A difficult \texttt{\textbackslash multicolumn} combination: pass 2 \\
5 & A difficult \texttt{\textbackslash multicolumn} combination: pass 3 \\
6 & A difficult \texttt{\textbackslash multicolumn} combination: pass 4 \\
7 & A summary of \texttt{longtable} commands
\end{tabular}

1 Introduction

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 \LaTeX{}’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{\textbackslash 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 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>This part appears at the top of the table</td>
<td></td>
</tr>
<tr>
<td>longtable columns are specified in the tabular environment.</td>
<td></td>
</tr>
<tr>
<td>@{}r</td>
<td>p{1in}@{}</td>
</tr>
<tr>
<td>Each row ends with a \ command.</td>
<td></td>
</tr>
<tr>
<td>The \ command has an optional argument, just as in the tabular environment.</td>
<td></td>
</tr>
<tr>
<td>See the effect of [10pt] ?</td>
<td></td>
</tr>
<tr>
<td>lots of lines like this.</td>
<td></td>
</tr>
<tr>
<td>lots of lines like this.</td>
<td></td>
</tr>
<tr>
<td>lots of lines like this.</td>
<td></td>
</tr>
<tr>
<td>lots of lines like this.</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>
</tbody>
</table>

This is a \multicolumn{2}{||c||} {|
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. |
The [t] [b] [c] argument of tabular can not be used. |
The optional argument may be one of [l] [r] [c] to specify whether the table should be adjusted to the left, right or centrally. |
Lots of lines like this. |
Lots of lines like this. |
Lots of lines like this. |
Lots of lines like this. |
Lots of lines like this. |
Lots of lines like this. |
Lots of lines like this. |
Lots of lines like this. |
Lots of lines like this. |
Lots of lines like this. |
Lots of lines like this. |
| |
| This goes at the bottom. |
Table 1: (continued)

<table>
<thead>
<tr>
<th>First</th>
<th>Second</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lots of lines</td>
<td>like this.</td>
</tr>
<tr>
<td>Lots of lines</td>
<td>like this.</td>
</tr>
<tr>
<td>Lots of lines</td>
<td>like this.</td>
</tr>
<tr>
<td>*Some lines may take up a lot of space, like this: This last column is a “p” column so this “row” of the table can take up several lines. Note however that \TeX will never break a page within such a row. Page breaks only occur between rows of the table or at \hline commands.</td>
<td></td>
</tr>
<tr>
<td>Lots of lines</td>
<td>like this.</td>
</tr>
<tr>
<td>Lots of lines</td>
<td>like this.</td>
</tr>
<tr>
<td>Lots of lines</td>
<td>like this.</td>
</tr>
<tr>
<td>Lots of lines</td>
<td>like this.</td>
</tr>
<tr>
<td>Lots of lines</td>
<td>like this.</td>
</tr>
<tr>
<td>Lots of lines</td>
<td>like this.</td>
</tr>
<tr>
<td>Lots of lines</td>
<td>like this.</td>
</tr>
<tr>
<td>Lots of lines</td>
<td>like this.</td>
</tr>
<tr>
<td>Lots of lines</td>
<td>like this.</td>
</tr>
<tr>
<td>Lots of lines</td>
<td>like this.</td>
</tr>
<tr>
<td>Lots of lines</td>
<td>like this.</td>
</tr>
<tr>
<td>Lots of lines</td>
<td>like this.</td>
</tr>
<tr>
<td>Lots of lines</td>
<td>like this.</td>
</tr>
<tr>
<td>These lines will appear in place of the usual foot at the end of the table.</td>
<td></td>
</tr>
</tbody>
</table>

\footnote{This is a footnote.}
\footnote{\texttt{longtable} takes special precautions, so that footnotes may also be used in ‘p’ columns.}
2 Chunk Size

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}}.\footnote{You can also use the plain \TeX\ syntax \texttt{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}. However, if necessary, \texttt{longtable} can work with \texttt{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 \texttt{LTchunksize} must be at least as large as the number of rows in each of the head or foot sections.

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

3 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 \texttt{\}\ command is replaced by a \texttt{\endhead} command. If the first page should have a different heading, then this should be entered in the same way, and terminated with the \texttt{\endfirsthead} command. The \texttt{LTchunksize} should be at least as large as the number of rows in the heading.

There are also \texttt{\endfoot} and \texttt{\endlastfoot} commands which are used in the same way (at the start of the table) to specify rows (or an \texttt{\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 \texttt{\firsthead}, or the beginning of the \texttt{\lastfoot}. This helps to control which lines appear on the first and last page of the table.

The \texttt{\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 \texttt{\setlength{\LTcapwidth}{2in}} in the preamble of your document. The default is 4in. \texttt{\caption} also writes the information to produce an entry in the list of tables. As with the \texttt{\caption} command in the \texttt{figure} and \texttt{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 \texttt{\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 \texttt{\caption} command in the first heading, and put a subsidiary caption in a \texttt{\caption[]} command in the main heading. If the optional argument to \texttt{\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 \texttt{\caption*} command.

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

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

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

4 Multicolumn entries

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

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

Consider Table 3. In the second chunk, \texttt{longtable} sees the wide \texttt{\multicolumn} entry. At this point it thinks that the first two columns are very narrow. All the width of the \texttt{\multicolumn} entry is assumed to be in the third column. (This is a ‘feature’ of \texttt{\TeX}’s primitive \texttt{\halign} command.) \texttt{longtable} then passes the information that there is a wide third column to the later chunks, with the result that the first pass over the table is too wide.
Table 3: A difficult \texttt{\multicolumn} combination: pass 1

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

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

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

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

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

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

\begin{tabular}{|c|c|c|}
\hline
1 & 2 & 3 \\
wide multicolumn spanning 1–3 & &  \\
multicolumn 1–2 & 3 &  \\
wide 1 & 2 & 3 \\
\hline
\end{tabular}
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 too 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
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 occuring 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 `\LaTeX`-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 \texttt{longtable} controls the horizontal alignment of the table. The possible options are \texttt{[c]}\texttt{, [r]} and \texttt{[l]}, for centring, right and left adjustment, respectively. Normally centring is the default, but this document specifies
\LTleft \parindent \LTright \fill

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 \texttt{\extracolsep} command. For instance
\begin{tabular*}{\textwidth}{@{\extracolsep{...}}...}
produces a full width table, to get a similar effect with \texttt{longtable} specify
\LTleft{0pt} \LTright{0pt}
\begin{longtable}{@{\extracolsep{...}}...}

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 \texttt{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, \texttt{longtable} does not need to issue a \texttt{\clearpage} at the start of the table, and so the table may start half way down a page. Also the \texttt{\endlastfoot} command which could not safely be implemented under the old scheme, has been added.
- \texttt{longtable} now issues an error if started in the scope of \texttt{\twocolumn}, or the \texttt{multicols} environment.
- The separate documentation file \texttt{longtable.tex} has been merged with the package file, \texttt{longtable.dtx} using Mittelbach’s \texttt{doc} package.
- Support for footnotes has been added. Note however that \texttt{\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 \texttt{\footnotemark} at that point, and \texttt{\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 \setlongtable 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’ multicolumn 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 \setlongtable was activated.)

• //* and \nopagebreak commands may be used to control page breaking.

7 Summary

Table 7: A summary of longtable commands

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>\LTleft</td>
<td>Glue to the left of the table. \fill</td>
<td></td>
</tr>
<tr>
<td>\LTright</td>
<td>Glue to the right of the table. \fill</td>
<td></td>
</tr>
<tr>
<td>\LTpre</td>
<td>Glue before the table. \bigskipamount</td>
<td></td>
</tr>
<tr>
<td>\LTpost</td>
<td>Glue after the table. \bigskipamount</td>
<td></td>
</tr>
<tr>
<td>\LTcapwidth</td>
<td>The width of a parbox containing the caption. 4in</td>
<td></td>
</tr>
<tr>
<td>\LTchunksize</td>
<td>The number of rows per chunk. 20</td>
<td></td>
</tr>
</tbody>
</table>

Optional arguments to \begin{longtable}

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td>Position as specified by \LTleft and \LTright.</td>
</tr>
<tr>
<td>[c]</td>
<td>Centre the table.</td>
</tr>
<tr>
<td>[l]</td>
<td>Place the table flush left.</td>
</tr>
<tr>
<td>[r]</td>
<td>Place the table flush right.</td>
</tr>
</tbody>
</table>
Commands to end table rows

\\ Specifies the end of a row
\[\langle \text{dim} \rangle\] 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 of 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{\langle \text{caption} \rangle} Caption ‘Table \(\text{\langle \text{caption} \rangle}\)' and a ‘\langle \text{caption} \rangle’ entry in the list of tables.
\caption[\langle \text{lot} \rangle]{\langle \text{caption} \rangle} Caption ‘Table \(\text{\langle \text{caption} \rangle}\)’, and a ‘\langle \text{\text{lot}} \rangle’ entry in the list of tables.
\caption[]\{\langle \text{caption} \rangle\} Caption ‘Table \(\text{\langle \text{caption} \rangle}\)’, but no entry in the list of tables.
\caption*\{\langle \text{caption} \rangle\} Caption ‘\(\langle \text{caption} \rangle\)’, but no entry in the list of tables.

Commands available at the start of a row

\pagebreak Force a page break.
\pagebreak[\langle \text{val} \rangle] A ‘hint’ between 0 and 4 of the desirability of a break.
\nopagebreak Prohibit a page break.
\nopagebreak[\langle \text{val} \rangle] 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}}\\
\hline\hline
\multicolumn{2}{@{*}c@{*}}{This part appears at the top of the table}\\
\textsc{First}&\textsc{Second}\\
\hline\hline
{This part appears at the top of every other page}\\
\textbf{First}&\textbf{Second}\\
\hline\hline
This goes at the\&bottom.\\
\hline
\endfoot
These lines will\&appear\\
in place of the \& usual foot\\
at the end\& of the table\\
\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...&
\raggedleft This last column is a \‗p‘ column...
\tabularnewline
... Lots of lines\& like this.\\
... \hline
Lots\footnote{...} of lines\& like this.\\
Lots of lines\& like this\footnote{...}\\
\hline
Lots of lines\& 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
\def\LT@err{\PackageError{longtable}}

\LT@warn The warning generating command
\def\LT@warn{\PackageWarning{longtable}}

\LT@final@warn If any \longtable{s} have not aligned, generate a warning at the end of the run at \AtEndDocument.
\def\LT@final@warn{\AtEndDocument{\LT@warn{Table \@width s have changed. Rerun LaTeX.\@gobbletwo}}}\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.
\DeclareOption{errorshow}{\def\LT@warn{\PackageInfo{longtable}}}\global\let\LT@warn\relax

pausing Make every warning message into an error so \TeX stops. May be useful for debugging.
\DeclareOption{pausing}{\def\LT@warn#1{\LT@err{#1}{This is not really an error}}}\global\let\LT@err\relax

set final The next options are just alternative syntax for the \setlongtables declaration.
\DeclareOption{set}{}\DeclareOption{final}{}

\ProcessOptions

9.3 User Setable Parameters

\LTleft Glue to the left and right of the table, default \fill (ie centred).
\newskip\LTleft \LTleft=\fill
\LTright \newskip\LTright \LTright=\fill
\LTpre Glue before and after the longtable. \bigskip by default.
\LTpost
18 \newskip\LTpre \LTpre=\bigskipamount
19 \newskip\LTpost \LTpost=\bigskipamount
\LTchunksize Chunk size (The number of rows taken per \halign). Default 20.
20 \newcount\LTchunksize \LTchunksize=20
\c@LTchunksize Added in V3.07 to allow the \LaTeX\ syntax \setcounter{LTchunksize}{10}.
21 \let\c@LTchunksize\LTchunksize
\LTcapwidth Width of the \parbox containing the caption. Default 4in.
22 \newdimen\LTcapwidth \LTcapwidth=4in

9.4 Internal Parameters
\LT@head Boxes for the table head and foot.
\LT@firsthead 23 \newbox\LT@head
\LT@foot 24 \newbox\LT@firsthead
\LT@lastfoot 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 \table, but this fails if \table is reset during a document, eg report class resets it every chapter.
This was changed from \newcount\LT@tables in V3.04. \LaTeX\ counters are preserved correctly when \includeonly is used. In the rest of the file \LT@tables has been replaced by \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 \table counter is not defined (eg in 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
\begin{longtable}

\ \end{longtable}
Set up these internal commands for `longtable`.

\global\let\LT@mcw\relax
\let\LT@mcol\multicolumn

Now redefine \@tabarray to restore \hline and \multicolumn so that arrays and tabulars nested in `longtable` (or in page headings on longtable pages) work out OK. Saving the original definitions done here so that you can load the `array` package before or after `longtable`.

\let\LT@@tabarray\@tabarray
\let\LT@@hl\hline
\def\@tabarray{%
\let\hline\LT@@hl
\let\multicolumn\LT@mcol
\LT@@tabarray}
\let\LT@LL@FM@cr\@tabularcr
\let\LT@LL@FM@cr\@tabularcr
\let\LT@LL@FM@cr\@tabularcr
\else
Initialise these commands as in `array`. \dollar replaced by \dollarbegin \dollarend in V3.03 to match `array` V2.0h. We do not need to set \dollarbegin and \dollarend as the `array` package gives them the correct values at the top level.

\advance\@tempdim\extrarowheight
\col@sep\tabcolsep

More or less standard definitions, but first start a \noalign.
\def\pagebreak\noalign{\ifnum'\relax}
\def\nopagebreak\noalign{\ifnum'\relax}
\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
\else
Initialise these commands as in `array`. \dollar replaced by \dollarbegin \dollarend in V3.03 to match `array` V2.0h. We do not need to set \dollarbegin and \dollarend as the `array` package gives them the correct values at the top level.
\advance\@tempdim\extrarowheight
\col@sep\tabcolsep
The rest of this macro is mainly based on \texttt{array} package, but should work for the standard \texttt{tabular} too.

\begin{verbatim}
\let\@startpbox\LT@startpbox\let\LT@LL@FM@cr\@arraycr
\fi
\\setbox\@arstrutbox\hbox{\vrule
\@height \arraystretch \@tempdima
\@depth \arraystretch \dp \strutbox
\@width \z@}%
\let\@sharp##\let\protect\relax
\begingroup
\@mkpream{#2}\
\xdef\LT@bchunk{\global\advance\c@LT@chunks\@ne
\global\LT@rows\z@\setbox\z@\vbox\bgroup
\LT@setprevdepth
\tabskip\LTleft\halign to\hsize\bgroup
\tabskip\z@ \@arstrut \@preamble \tabskip\LTRight \cr}\
\endgroup
\expandafter\LT@nofcols\LT@bchunk&\LT@nofcols
\LT@make@row
\m@th\let\par\@empty
\everycr{}\lineskip\z@\baselineskip\z@
\LT@bchunk}
\end{verbatim}

The following line was added in v4.05. In order to get the \texttt{\penalties} to work at chunk boundaries Need to take more care about where and when \texttt{\lineskip} glue is added. The following does nothing at top of table, and in header chunks, but in normal body chunks it sets \texttt{\prevdepth} (to 0pt, but any value would do) so that \texttt{\lineskip} glue will be added. the important thing to note is that the glue will be added after any vertical material coming from \texttt{\noalign}.

\begin{verbatim}
\LT@setprevdepth
\tabskip\LTleft\halign to\hsize\bgroup
\tabskip\z@ \@arstrut \@preamble \tabskip\LTRight \cr}\
\endgroup
\expandafter\LT@nofcols\LT@bchunk\LT@nofcols
\LT@make@row
\m@th\let\par\@empty
\everycr{}\lineskip\z@\baselineskip\z@
\LT@bchunk}
\end{verbatim}

A few more internal commands for \texttt{longtable}.

\begin{verbatim}
\m@th\let\par\@empty
\everycr{}\lineskip\z@\baselineskip\z@
\LT@bchunk}
\end{verbatim}

\texttt{\LT@no@pgbk} Can simplify the standard \texttt{\no@pgbk} as this is vmode only but then need to close the \texttt{\noalign}.

\begin{verbatim}
\def\LT@no@pgbk#1[#2]{\penalty #1\@getpen{#2}\ifnum'='0\fi}}
\end{verbatim}
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.

\LT@start

\def\LT@start{%
\let\LT@start\endgraf
\endgraf\penalty\z@\vskip\LTpre

Start a new page if there is not enough room for the table head, foot, and one extra line.

\dimen0\pagetotal
\advance\dimen0 \ht\ifvoid\LT@firsthead\LT@head\else\LT@firsthead\fi
\advance\dimen0 \dp\ifvoid\LT@firsthead\LT@head\else\LT@firsthead\fi
\advance\dimen0 \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.

\dimen0\iivfuzz
\vfuzz\maxdimen
\setbox\tw@\copy\z@
\setbox\tw@\vsplit\tw@ to \ht\@arstrutbox
\setbox\tw@\vbox{\unvbox\tw@}%
\vfuzz\dimen0\iivfuzz
\dimen0\ii\vfuzz\maxdimen
\ifdim\ht\@arstrutbox>\ht\tw@\@arstrutbox\else\tw@\fi
\advance\dimen0\dp\ifdim\dp\@arstrutbox>\dp\tw@\@arstrutbox\else\tw@\fi

\advance\dimen0 \ht\ifdim\ht\@arstrutbox>\ht\tw@\@arstrutbox\else\tw@\fi
\advance\dimen0 \dp\ifdim\dp\@arstrutbox>\dp\tw@\@arstrutbox\else\tw@\fi

\advance\dimen0 \ht\ifdim\ht\@arstrutbox>\ht\tw@\@arstrutbox\else\tw@\fi
\advance\dimen0 \dp\ifdim\dp\@arstrutbox>\dp\tw@\@arstrutbox\else\tw@\fi

\advance\dimen0 \ht\ifdim\ht\@arstrutbox>\ht\tw@\@arstrutbox\else\tw@\fi
\advance\dimen0 \dp\ifdim\dp\@arstrutbox>\dp\tw@\@arstrutbox\else\tw@\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
\dimen0\pagegoal\advance\dimen0\ht\LT@foot\pagegoal\dimen0
\maxdepth\z@\fi

\ifvoid\LT@firsthead\copy\LT@head\else\box\LT@firsthead\fi
\output{\LT@output}
\endlongtable Called by \end{longtable}.

\def\endlongtable{%

Essentially add a final \\. 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.

\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.

\expandafter\noexpand\csname LT@\romannumeral\c@LT@tables\endcsname{\LT@save@row}}}%\fi

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?}
\global\@mparbottom\z@
\pagegoal\vsize
\endgraf
\penalty\z@
\addvspace\LTpost
Footnotes. As done in the multicol package.
\ifvoid\footins\else\insert\footins{}\fi
\begin{longtable}{l}


\noindent 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.
\def\LT@@nofcols#1&{%
\futurelet\@let@token\LT@@nofcols}
\LT@@nofcols 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.
\def\LT@@nofcols{%
\advance\LT@cols\@ne
\ifx\@let@token\LT@@nofcols
\expandafter\@gobble
\else
\expandafter\LT@@nofcols
\fi}
\LT@\tabularcr The internal definition of \tabular. 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.

\end{longtable}
\{\texttt{ifnum0='}\}\texttt{fi} added in version 3.01, required if the first entry is empty. The above in fact is not good enough, as with \texttt{array} package it can introduce a \{\} group in math mode, which changes the spacing. So use the following variant.

\begin{verbatim}
\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 \texttt{\ prevdepth} at the start of a chunk.
\LT@setprevdepth\relax
\LT@t@bularcr
\def\LT@t@bularcr{%
Increment the counter, and do \texttt{tabular}'s \texttt{\} or finish the chunk.
The \texttt{\expandafter} trick was added in Version 3. Set the \texttt{\ prevdepth} at the start
of a new chunk. (Done here so not set in header chunks).
\global\advance\LT@rows\@one
\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 justs looks for an optional argument.
\def\LT@xtabularcr{%
\@ifnextchar[\LT@argtabularcr\LT@ntabularcr}
\LT@ntabularcr
The version with no optional argument. \texttt{\texttt{ifnum0='\{\fi}} added in version 3.01.

\begin{verbatim}
\def\LT@ntabularcr{%
\ifnum0='{\fi
\LT@echunk
\LT@start
\unvbox\z@
\LT@get@widths
\LT@echunk}
\end{verbatim}

{\normalfont longtable.sty}
\LT@argtabularcr The version with an optional argument. \ifnum0='{\fi} added in version 3.01.
Changed in 3.14.
199 \def\LT@argtabularcr[#1]{% 
200 \ifnum0='{}\fi 
201 \ifdim #1>\z@
202 \unskip\@xargarraycr{#1}%
203 \else
204 \@yargarraycr{#1}%
205 \fi 
Add the dummy row, and finish the \halign.
206 \LT@echunk
207 \LT@start
208 \unvbox\z@
209 \LT@get@widths
210 \LT@bchunk}
\LT@echunk This ends the current chunk, and removes the dummy row.
211 \def\LT@echunk{% 
212 \crcr\LT@save@row\cr\egroup 
213 \global\setbox\@ne\lastbox 
The following line was added in v4.05. longtable relies on \lineskip glue (which is 0pt) to provide break points between each row so the table may be split into pages.

Previous releases left the \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 \lineskip glue at the top of a box. This meant that normally the chunks fitted together perfectly, however \nolimits material at a chunk boundary came before the first row of the next chunk but after the lineskip glue at the end of this chunk. This is the wrong place, e.g., it means even a \penalty10000 does not stop a break as the \lineskip glue in the previous item on the list provides a legal breakpoint. So now remove the \lineskip glue that was before the dummy row and introduce \LT@setprevdepth to set the \prevdepth at the start of the next chunk, to make sure \lineskip glue is added later.
214 \unskip
215 \egroup}
\LT@entry We here give the ‘basic’ definition of \LT@entry, namely that used in alignment templates. It has a \kern only if the maximum is imposed from a different chunk. The \ifhmode test reveals the first entry, when we don’t want to add an \&.
216 \def\LT@entry#1#2{% 
217 \ifhmode@firstofone\&\fi\omit 
218 \ifnum#1=\c@LT@chunks 
219 \else 
220 \kern#2\relax 
221 \fi}
This definition for the argument of \LT@save@row 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.

\LT@entry@chop
\def\LT@entry@chop#1#2{%\noexpand\LT@entry\ifnum#1>\c@LT@chunks\{Opt\}#1}{#2\fi}

\LT@entry@write
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...).

\def\LT@entry@write{%\noexpand\LT@entry^^J\@spaces}

\LT@kill
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.

\def\LT@kill{%\LT@echunk\LT@get@widths\expandafter\LT@rebox\LT@bchunk}

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

\def\LT@rebox#1\bgroup{%\#1\bgroup\unvbox\z@\unskip\setbox\z@\lastbox}

9.8 The Dummy Row
The dummy row is kept inside of the macro \LT@save@row.

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

\def\LT@blank@row{%\xdef\LT@save@row{\expandafter\LT@save@row\LT@build@blank\romannumeral\number\LT@cols 001}}
Whoops! What's that supposed to be? A drop-in replacement for the first task of Appendix D in the \TeXbook. The \texttt{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}
\LT@make@row
\end{verbatim}

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{\let\LT@entry\or\if!\ifcase\expandafter\expandafter\expandafter\LT@cols\expandafter\@gobble\LT@save@row\or\else\relax\fi}\fi}\fi}
\end{verbatim}

Now a slightly difficult part comes. Before we decide making the template from the .aux file info we check that the number of fields has remained the same. If it hasn’t, either the table format has changed, or we have the wrong table altogether. In both cases, we decide to better drop all gathered information and start over.

The expansion between !...! below will be empty if the number of \texttt{\LT@entry} macros including arguments in \texttt{\LT@save@row} is equal to \texttt{\LT@cols}. If it is not empty, we throw the row away and start from scratch.

\begin{verbatim}
\else{\let\LT@entry\or\
\if!\ifcase\expandafter\expandafter\expandafter\LT@cols\expandafter\@gobble\LT@save@row\or\else\relax\fi}\fi!\else\aftergroup\LT@blank@row\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 \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}
This is the heart of longtable. If it were not for the table head and foot, this macro together with the modified `\` command would form the basis of quite a simple little package file for long tables. It is closely modelled on the `\endvrulealign` macro of appendix D of the \TeXbook.

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

Add a column to the dummy row. Name changed from `\def\LT@save@row` in Version 3, to preserve the `\LT@` naming convention.

\begin{verbatim}
def\LT@def@row{%
  \let\LT@entry\or
  \edef\@tempa{\ifcase\expandafter\count@\LT@old@row
    \else
      {1}{0pt}\fi}
  \let\LT@entry\relax
  \xdef\LT@save@row{\LT@entry\expandafter\LT@max@sel\@tempa\LT@save@row}}
\end{verbatim}

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

\begin{verbatim}
def\LT@max@sel
\end{verbatim}
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 \texttt{\the\wd\tw@} construct instead of \texttt{#2} because \texttt{#2} might be read in from the file, and so could have \texttt{\catcode 11} versions of \texttt{p} and \texttt{t} in it which we want to be replaced by their ‘proper’ \texttt{\catcode 12} versions.

\begin{verbatim}
\def\LT@max@sel#1#2{\ifdim#2=\wd\tw@ #1\else \number\c@LT@chunks \fi}\the\wd\tw@}
\end{verbatim}

9.9 The \texttt{\hline} Command

\texttt{\hline} and \texttt{\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 \texttt{\hline} produces a line on both pages.\footnote{\texttt{longtable} has always done this, but perhaps it would be better if \texttt{hlines} were \texttt{omitted} at a page break, as the head and foot usually put a \texttt{hline} here anyway.} Also this \texttt{\hline} is more like a \texttt{\cline{1-\LT@cols}}. \texttt{tabular}'s \texttt{\hline} would draw lines the full width of the page.

\begin{verbatim}
\def\LT@hline{%
  \noalign{\ifnum0='}\fi
  \penalty\@M
  \futurelet\@let@token\LT@@hline}

\def\LT@@hline{%
  \ifx\@let@token\hline
    \global\let\@gtempa\@gobble
    \gdef\LT@sep{\penalty-\@medpenalty\vskip\doublerulesep}%
  \else
    \global\let\@gtempa\@empty
    \gdef\LT@sep{\penalty-\@lowpenalty\vskip-\arrayrulewidth}%
  \fi
  \ifnum0='\fi
  \multispan\LT@cols
  \unskip\leaders\hrule\@height\arrayrulewidth\hfill\cr
  \noalign{\LT@sep}
  \multispan\LT@cols
  \unskip\leaders\hrule\@height\arrayrulewidth\hfill\cr
  \noalign{\penalty\@M}
  \@gtempa}
\end{verbatim}
9.10 Captions

\LT@caption The caption is \multicolumn{\LT@cols}{c}{\langle a parbox with the table's caption \rangle}.
\begin{verbatim}
def\LT@caption\
noalign\bgroup
@ifnextchar[\egroup\LT@c@ption@firstofone]\LT@caption\endverbatim
\end{verbatim}

\LT@c@ption Caption command (with [optional argument]). \protect added in Version 3. \fnum@table added at V3.05.
\begin{verbatim}
def\LT@c@ption#1[#2]#3{\LT@makecaption#1\fnum@table{#3}\
def\@tempa{#2}\
@ifx\@tempa\@empty\else\let\\space\addcontentsline{lot}{table}{\protect\numberline{\thetable}{#2}}\fi}
\end{verbatim}

\LT@capt@n Caption command (no [optional argument])
\begin{verbatim}
def\LT@capt@n%\@ifstar{\egroup\LT@c@ption\@gobble}\{\egroup\@xdblarg\LT@c@ption@firstofone}\endverbatim
\end{verbatim}

\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.
\begin{verbatim}
def\LT@makecaption#1#2#3{\LT@mcol\LT@cols c{\hbox to\z@{\hss\parbox[t]\LTcapwidth{#1\@gobble\
@firstofone\else#2\fi\addcontentsline{lot}{table}{\protect\numberline{\thetable}{#2}}}\fi}}
\end{verbatim}

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.
\begin{verbatim}
def\LT@output%\ifnum\outputpenalty <-\@Mi\ifnum\outputpenalty > -\LT@endopen\endgraf\vskip\baselineskip\fi}
\end{verbatim}

............................ longtable.sty ............................
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:
\dimen@\pagegoal
\advance\dimen@-\ht\LT@lastfoot
\ifdim\dimen@<\ht\z@
\setbox\@cclv\vbox{\unvbox\z@\copy\LT@foot\vss}\
\@makecol
\@outputpage
\setbox\z@\vbox{\box\LT@head}\
End of \ifdim\dimen@<\ht\@cclc.
\fi
End of \ifdim\ht\LT@lastfoot>\ht\LT@foot.
\fi
Reset \@colroom.
\global\@colroom@includeht
\global\vsize@includeht
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}\
End of \ifnum\outputpenalty > -\LT@end@pen.
\fi
Else \outputpenalty > -@Mi.
\else
If we have not reached the end of the table,
\setbox\@cclv\vbox{\unvbox\@cclv\copy\LT@foot\vss}\
\@makecol
\@outputpage
Reset \vsize.
\global\vsize@includecolroom
Put the head at the top of the next page.
\copy\LT@head
End of \ifnum\outputpenalty <-@Mi.
\fi

\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.}
9.12 Commands for the the table head and foot

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.

376 \def\LT@end@hd@ft#1{%
377 \LT@echunk
Changed from \relax to \endgraf at V3.04, see \LT@start.
378 \ifx\LT@start\endgraf
379 \LT@err
380 {Longtable head or foot not at start of table}%
381 {Increase LTchunksize}%
382 \fi
383 \setbox#1\box\z@
384 \LT@get@widths
385 \LT@bchunk}

Call \LT@end@hd@ft with the appropriate box.

386 \def\endfirsthead{\LT@end@hd@ft\LT@firsthead}
387 \def\endhead{\LT@end@hd@ft\LT@head}
388 \def\endfoot{\LT@end@hd@ft\LT@foot}
389 \def\endlastfoot{\LT@end@hd@ft\LT@lastfoot}

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.

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.

Add extra code to switch the definition of \@footnotetext.

390 \def\LT@startpbox#1{%
391 \bgroup
392 \let\@footnotetext\LT@p@ftntext
393 \setlength\hsize{#1}%
394 \@arrayparboxrestore
395 \vrule \@height \@arstrutbox \@width \z@}

After the parbox is closed, expand \LT@p@ftn 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.

\begin{verbatim}
def\LT@endpbox{\@finalstrut\@arstrutbox\
\egroup\the\LT@p@ftn\global\LT@p@ftn{}}
\LT@p@ftntext Inside the ‘p’ column, just save up the footnote text in a token register.
def\LT@p@ftntext#1{\edef\@tempa{\the\LT@p@ftn\noexpand\footnotetext[\the\c@footnote]{#1}}\global\LT@p@ftn{\expandafter{\@tempa{#1}}}}
\end{verbatim}