The \texttt{makecell} package*

Olga Lapko  
Lapko.O@g23.relcom.ru  
2009/08/03

Abstract

This package helps to create common layout for tabular material. The \texttt{\thead} command, based on one-column tabular environment, is offered for creation of tabular column heads. This macro allows to support common layout for tabular column heads in whole documentation. Another command, \texttt{\makecell}, is offered for creation of multilined tabular cells. There are also command \texttt{\makecellbox} command and \texttt{mcellbox} environment usage of tabular cells inside text.

Package also offers: 1) macro \texttt{\makegapedcells}, which changes vertical spaces around all cells in tabular, like in \texttt{tabls} package, but uses code of \texttt{array} package. (Macro \texttt{\makegapedcells} redefines macro \texttt{\@classz} from \texttt{array} package. Macro \texttt{\nomakegapedcells} cancels this redefinition.); 2) macros \texttt{\multirowhead} and \texttt{\multirowcell}, which use \texttt{\multirow} macro from \texttt{multirow} package; 3) numbered rows \texttt{\nline} or skipping cells \texttt{\eline} in tabulars; also are the skipping rows command \texttt{\erows} and repeated rows command \texttt{\Xrows}; 4) diagonally divided cells (\texttt{\diaghead}) plus citation of sample file of \texttt{slashbox} package, which does the same; 5) \texttt{\hline} and \texttt{\cline} with defined thickness: \texttt{\Xhline} and \texttt{\Xcline} consequently.

\footnote{This file has version number V0.1e, last revised 2009/08/03.}
# Contents

1 Tabular Cells and Column Heads .................................................. 3
   1.1 Building Commands ......................................................... 3
   1.2 Settings For Tabular Cells ............................................... 5
   1.3 Settings For Column Heads .............................................. 6

2 Changing of Height and Depth of Boxes .................................... 7

3 How to Change Vertical Spaces Around Cells in Whole Table ............ 8

4 Multirow Table Heads and Cells ................................................ 10
   4.1 Multirow Table Heads and Cells: Second Variant .................... 13

5 Numbered Lines in Tabulars ..................................................... 14

6 Cells (One-Column Tabulars) in the Text .................................. 15

7 Diagonally Divided Cell .......................................................... 16

8 Comparison: Diagonally Divided Cell with the slashbox package ....... 18

9 Thick Rules for the hline and cline Commands .............................. 21

10 Code of package ........................................................................ 23
   10.1 Multilined cells ............................................................ 23
   10.2 Gape commands ............................................................ 28
   10.3 Modification of command from array package ...................... 29
   10.4 Rows of skipped and numbered cells ................................ 29
   10.5 Diagonally separated column heads .................................. 32
   10.6 The \hline and \cline with necessary thickness ....................... 34
1 Tabular Cells and Column Heads

1.1 Building Commands

\makecell Macro creates one-column tabular with predefined common settings of alignment, spacing and vertical spaces around (see section \ref{sec:alignment}). This will be useful for creation of multilined cells. This macro allows optional alignment settings.

\makecell[\text{(vertical or/or horizontal alignment})\text{]{(cell text)}

For vertical alignment you use t, b, or c—this letters you usually put in optional argument of \texttt{tabular} or \texttt{array} environments. For horizontal alignment you may use alignment settings like r, l, or c, or more complex, like \texttt{p{3cm}}. Since this package loads \texttt{array} package, you may use such alignment settings like \texttt{>{\parindent1em}p{3cm}}.

\begin{tabular}{|c|c|}
\hline
Cell text & 28--31\\
\hline
\makecell{Multilined \ cell text} & 28--31\\
\hline
\makecell[l]{Left aligned \ cell text} & 37--43\\
\hline
\makecell[r]{Right aligned \ cell text} & 37--43\\
\hline
\makecell[b]{Bottom aligned \ cell text} & 52--58\\
\hline
\makecell*[r\{p{3cm}\}]{Cell long text with predefined width} & 52--58\\
\hline
\makecell*[\{>{\parindent1em}p{3cm}\}]{Cell long...} & 52--58\\
\hline
\end{tabular}

Starred form of command, \texttt{\makecell*}, creates vertical \texttt{\jot} spaces around.

\textit{Note}. When you define column alignment like \texttt{p{3cm}} in optional argument of \texttt{\makecell} (or \texttt{\thead}, see below), please follow these rules: 1) if vertical alignment defined, write column alignment in group, e.g. \texttt{c{p{3cm}}}; 2) if vertical alignment is absent, write column alignment in double group—\texttt{p{3cm}}; or add empty group—\texttt{p{3cm}}]. Be also careful with vertical alignment when you define column alignment as paragraph block: e.g., use \texttt{b{3cm}} for bottom alignment (and \texttt{m{3cm}} for centered vertical alignment).

\thead Macro creates one-column \texttt{tabular} for column heads with predefined common settings (see table \ref{tab:alignment}). This macro uses common layout for column heads: font, alignment, spacing, and vertical spaces around (see section \ref{sec:alignment}.

\renewcommand\theadset{\def\arraystretch{.85}}%
\begin{tabular}{|l|c|}
\hline
\hline

Table 1. Example of multilined cells

<table>
<thead>
<tr>
<th>Cell text</th>
<th>28–31</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multilined cell text</td>
<td>28–31</td>
</tr>
<tr>
<td>Left aligned cell text</td>
<td>37–43</td>
</tr>
<tr>
<td>Right aligned cell text</td>
<td>37–43</td>
</tr>
<tr>
<td>Bottom aligned cell text</td>
<td>52–58</td>
</tr>
<tr>
<td>Cell long text with predefined width</td>
<td>52–58</td>
</tr>
</tbody>
</table>

\begin{tabular}{|l|c|}
  \hline
  Left column text & 28--31 \\
  \hline
\end{tabular}

Table 2. Example of column heads

<table>
<thead>
<tr>
<th>First column head</th>
<th>Second \multlined \ column head</th>
</tr>
</thead>
<tbody>
<tr>
<td>Long left column text</td>
<td>28–31</td>
</tr>
</tbody>
</table>

\texttt{\thead{First column head} & \thead{Second \multlined \ column head} \hline \hline \texttt{Left column text & 28--31 \hline \end{tabular}}

Starred form of command, \texttt{\thead*}, creates vertical \texttt{jot} spaces around. \texttt{\rothead} creates table heads rotated by \texttt{90$^\circ$} counterclockwise. Macro uses the same font and spacing settings as previous one, but column alignment changed to \texttt{p\{\rotheadsize\}} with \texttt{\raggedright} justification: in this case left side of all text lines “lies” on one base line.

This parameter defines the width of rotated tabular heads. You may define that like:

\texttt{\setlength\rotheadsize{3cm}}

or

\texttt{\settowidth\rotheadsize{\theadfont \{Widest head text\}}}

like in following example (table 3):

\texttt{\settowidth\rotheadsize{\theadfont Second multilined} \begin{tabular}{|l|c|}}

4
1.2 Settings For Tabular Cells

This section describes macros, which make layout tuning for multilined cells, created by \makecell macro (and also \multirowcell and \rotcell macros). The \cellset macro also is used by \thead (\rothead, \multirowthead) macro.

\cellset Spacing settings for cells. Here you could use commands like:
\renewcommand\cellset{\renewcommand\arraystretch{1}\setlength\extrarowheight{0pt}}
as was defined in current package.

\cellalign Default align for cells. Package offers vertical and horizontal centering alignment, it defined like:
\renewcommand\cellalign{cc}

\cellgape Define vertical spaces around \makecell, using \gape command if necessary. It defined like:
\renewcommand\cellgape{}

You may define this command like
\renewcommand\cellgape{\Gape[1pt]}

or
\renewcommand\cellgape{\gape[t]}
(See also section 2 about \gape and \Gape command.)

\cellrotangle The angle for rotated cells and column heads. The default value 90 (counterclockwise). This value definition is used by both \rotcell and \rothead macros.
1.3 Settings For Column Heads

This section describes macros, which make layout tuning for tabular column heads, created by \thead (\rothead, \multirowthead) macro.

- \theadfont: Sets a special font for column heads. It could be smaller size
  \renewcommand\theadfont{\footnotesize}

  as was defined in current package (here we suppose that \small command used for tabular contents itself). Next example defines italic shape
  \renewcommand\theadfont{\itshape}

- \theadset: Spacing settings for column heads. Here you could use commands like:
  \renewcommand\theadset{\renewcommand\arraystretch{1.0}
  \setlength{\extrarowheight}{0pt}}

- \theadalign: Default align for tabular column heads. Here also offered centering alignment:
  \renewcommand\theadalign{cc}

- \theadgape: Define vertical spaces around column head (\thead), using \gape command if necessary. It defined like:
  \renewcommand\theadgape{\gape}

- \rotheadgape: Analogous definition for rotated column heads. Default is absent:
  \renewcommand\rotheadgape{\gape}
2 Changing of Height and Depth of Boxes

Sometimes \texttt{tabular} or \texttt{array} cells, or some elements in text need a height/depth correction. The \texttt{\textbackslash \raisebox} command could help for it, but usage of that macro in these cases, especially inside math, is rather complex. Current package offers the \texttt{\textbackslash gape} macro, which usage is similar to \texttt{\textbackslash smash} macro. The \texttt{\textbackslash gape} macro allows to change height and/or depth of included box with necessary dimension.

This macro changes included box by \texttt{\textbackslash jot} value (usually 3 pt). It is defined with optional and mandatory arguments, like \texttt{\textbackslash smash} macro, which \texttt{(re)defined by amsmath} package. Optional argument sets change of height only (t) or depth only (b). Mandatory argument includes text.

\texttt{\textbackslash gape\{t or b\}\{text\}}

Examples of usage:

\begin{verbatim}
\gape{text} \gape[t]{text} \gape[b]{text}
\end{verbatim}

\texttt{\textbackslash Gape}

Another way of height/depth modification. This macro allows different correction for height and depth of box:

\texttt{\textbackslash Gape\{height corr\}\{depth corr\}\{text\}}

If both arguments absent, \texttt{\textbackslash Gape} command works like \texttt{\textbackslash gape\{text\}}, in other words, command uses \texttt{\textbackslash jot} as correction value for height and depth of box.

If only one optional argument exists, \texttt{\textbackslash Gape} command uses value from this argument for both height and depth box corrections.

\begin{verbatim}
\Gape{text} \Gape[\jot]{text}
\Gape[6pt]{text} \Gape[6pt][-2pt]{text}
\end{verbatim}

You may also use \texttt{\textbackslash height} and \texttt{\textbackslash depth} parameters in optional arguments of \texttt{\textbackslash Gape} macro, parameters was borrowed from \texttt{\textbackslash raisebox} command.

These three macros modify standard \texttt{\textbackslash strut} by \texttt{\textbackslash jot} value: \texttt{\textbackslash bottopstrut} changes both height and depth; \texttt{\textbackslash topstrut} changes only height; \texttt{\textbackslash botstrut} changes only depth. These commands could be useful, for example, in first and last table rows.

Note. If you use \texttt{bigstrut} package note that these macros duplicate \texttt{\textbackslash bigstrut}, \texttt{\textbackslash bigstrut[t]}, and \texttt{\textbackslash bigstrut[b]} commands consequently. Please note that value, which increases strut in \texttt{\textbackslash topstrut} etc. equals to \texttt{\textbackslash jot}, but \texttt{\textbackslash bigstrut} and others use a special dimension \texttt{\textbackslash bigstrutjot}. 

3 How to Change Vertical Spaces Around Cells in Whole Table

This section describes macros which try to emulate one of possibilities of \texttt{tabls} package: to get necessary vertical spacing around cells.

Sets the parameters for vertical spaces:

\begin{verbatim}
\setcellgapes[(t or b)]{value}
\end{verbatim}

The next examples with array and tabular use following settings:

\begin{verbatim}
\setcellgapes{5pt}
\end{verbatim}

You may also try to load negative values if you wish. This macro you may put in the preamble as common settings.

The first macro switches on vertical spacing settings. The second cancels first one.

The \texttt{\makegapedcells} macro temporarily redefines macro \texttt{\@cellsize} of \texttt{array} package, so use this mechanism carefully. Load \texttt{\makegapedcells} inside group or inside environment (see table 4):

\begin{verbatim}
\begin{table}[h]
\makegapedcells
...
\end{table}
\end{verbatim}

Please note that space defined in \texttt{\setcellgapes} and space which creates \texttt{\gape} mechanism in commands for tabular cells (usually \texttt{\thead} or \texttt{\makecell*}) are summarized.
Table 4. Example of multilined cells with additional vertical spaces

<table>
<thead>
<tr>
<th>Cell text</th>
<th>28–31</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multilined cell text</td>
<td>28–31</td>
</tr>
<tr>
<td>Left aligned cell text</td>
<td>37–43</td>
</tr>
<tr>
<td>Right aligned cell text</td>
<td>37–43</td>
</tr>
<tr>
<td>Bottom aligned cell text</td>
<td>52–58</td>
</tr>
<tr>
<td>Cell long text with predefined width</td>
<td>52–58</td>
</tr>
<tr>
<td>Cell long text with predefined width</td>
<td>52–58</td>
</tr>
</tbody>
</table>
# Multirow Table Heads and Cells

The next examples show usage of macros which use \multirow command from \texttt{multirow} package.

At first goes short repetition of arguments of \multirow macro itself:

\[
\text{\texttt{\multirow} } \langle \text{nrow} \rangle \langle \text{njot} \rangle \langle \text{width} \rangle \langle \text{vmove} \rangle \langle \text{contents} \rangle
\]

- \langle nrow \rangle sets number of rows (i.e. text lines); \langle njot \rangle is mainly used if you’ve used bigstrut package: it makes additional tuning of vertical position (see comments in \texttt{multirow} package); \langle width \rangle defines width of contents, the * sign used to indicate that the text argument’s natural width is to be used; \langle vmove \rangle is a length used for fine tuning: the text will be raised (or lowered, if \langle vmove \rangle is negative) by that length; \langle contents \rangle includes \texttt{\multirow}’ed text.

These two macros use following arguments (example uses \texttt{\multirowcell} command):

\[
\text{\texttt{\multirowcell} } \langle \text{nrow} \rangle \langle \text{vmove} \rangle \langle \text{hor alignment} \rangle \langle \text{contents} \rangle
\]

in these macros were skipped \langle njot \rangle and \langle width \rangle. Instead of tuning optional argument \langle njot \rangle for vertical correction used \langle vmove \rangle optional argument. For the \langle width \rangle argument both \texttt{\multirowcell} and \texttt{\multirowthead} macros use natural width of contents (i.e. the * argument used).

First example (table 5) with \texttt{\multirow}’ed column heads and cells:

\[
\renewcommand\theadset{\def\arraystretch{.85}}% \\
\begin{tabular}{|l|c|c|} \\
\multirowthead{4}{First ...} & \multicolumn{2}{c|}{\thead{Multicolumn head}}\hline \\
& \thead{Second ...} & \thead{Third ...}\hline \\
\makecell{Multilined\ Cell text} & A & \multirowcell{3}{28--31}\hline \\
\makecell{Left ...} & C & \multirowcell{4}{1ex}\[1ex\]{37--43\52--58}\hline \\
\makecell{Cell ...} & F & \makecell{Cell ...} & G & \end{tabular}
\]

Second example (table 6) with \texttt{\multirow}’ed column heads and cells uses \texttt{\makegapedcells} command. The \texttt{\theadgape} command does nothing:

\[
\makegapedcells \\
\renewcommand\theadset{\def\arraystretch{.85}}% \\
\renewcommand\theadgape{} \\
\end{tabular}
\]

The last example (table 7) uses \texttt{tabularx} environment with \texttt{\hspace} in the width argument.
Table 5. Example of "\multirow'ed" cells

<table>
<thead>
<tr>
<th>First Column head</th>
<th>Multicolumn head</th>
<th>Second multlined column head</th>
<th>Third column head</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cell text</td>
<td>A</td>
<td></td>
<td>28–31</td>
</tr>
</tbody>
</table>
| Multiline
Cell text    | B                |                             |                  |
| Left aligned
Cell text | C                |                             | 37–43            |
| Right aligned
Cell text | D                |                             |                  |
| Bottom aligned
Cell text | E                |                             |                  |
| Cell long long long text with predefined width | F | 37–43 |                  |
| Cell long long long text with predefined width | G | 52–58 |                  |

Table 6. Example of "\multirow'ed" cells and additional vertical spaces

<table>
<thead>
<tr>
<th>First Column head</th>
<th>Multicolumn head</th>
<th>Second multlined column head</th>
<th>Third column head</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cell text</td>
<td>A</td>
<td></td>
<td>28–31</td>
</tr>
</tbody>
</table>
| Multiline
Cell text    | B                |                             |                  |
| Left aligned
Cell text | C                |                             | 37–43            |
| Right aligned
Cell text | D                |                             |                  |
| Bottom aligned
Cell text | E                |                             |                  |
<p>| Cell long long long text with predefined width | F | 37–43 |                  |
| Cell long long long text with predefined width | G | 52–58 |                  |</p>
<table>
<thead>
<tr>
<th>First Column head</th>
<th>Multicolumn head</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cell text</td>
<td>A</td>
</tr>
<tr>
<td>Multilined</td>
<td>B</td>
</tr>
<tr>
<td>Cell text</td>
<td>28–31</td>
</tr>
<tr>
<td>Left aligned</td>
<td>C</td>
</tr>
<tr>
<td>cell text</td>
<td>37–43</td>
</tr>
<tr>
<td>Right aligned</td>
<td>D</td>
</tr>
<tr>
<td>cell text</td>
<td></td>
</tr>
<tr>
<td>Bottom aligned</td>
<td>E</td>
</tr>
<tr>
<td>cell text</td>
<td></td>
</tr>
<tr>
<td>Cell long long long long long long text with predefined width</td>
<td>F</td>
</tr>
<tr>
<td>Cell long long long long text with predefined width</td>
<td>G</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

As you may see the \makecell’s in last two rows defined as

\makecell\{p{\hsize}\}{{{...}}}

and

\makecell\{>{{parindent1em}}p{\hsize}\}{{{...}}}

consequently.
4.1 Multirow Table Heads and Cells: Second Variant

Another, simplified, variant of multirow cell: use `\makecell` and `\thead` commands, and set `\` with negative space at the end, for example

```
\thead{First Column head\\[-5ex]}
```

cells, which stay in one “multi row” will have the same value of this negative space, in spite of different number of lines in their contents.
5 Numbered Lines in Tabulars

The three commands \eline, \nline, \rline allow to skip:

\eline\{\textit{number of cells}\}

and numbering (\nline) a few/all sells in the row:

\nline[\textit{numbering type}][\langle start number\rangle]\{\textit{number of cells}\}

Command \rline does the same as \nline, but allows numbering by Russian letters (it redefines \LaTeX’s \texttt{\Alph} and \texttt{\alph} with \texttt{\Asbuk} and \texttt{\asbuk} consequently).

(see table 9)

\begin{tabular}{|*{12}{c|}}
\hline
\eline{6} \hline
\nline{6} \hline
\eline{3} & \nline[1][4]{3} \hline
\nline[\texttt{(a)}]{6} \hline
\nline[column 1]{6} \hline
\end{tabular}

Table 8. Examples of filling of cells

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(1)</td>
<td>(2)</td>
<td>(3)</td>
<td>(4)</td>
<td>(5)</td>
<td>(6)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>column 1</td>
<td>column 2</td>
<td>column 3</td>
<td>column 4</td>
<td>column 5</td>
<td>column 6</td>
<td></td>
</tr>
</tbody>
</table>

Two “lazy” commands: for empty table rows (\erows) and for rows with repeated text (\Xrows) were added in the version V0.1e.

\begin{tabular}{|*{4}{c|}}
\hline
\erows{4}{8} \hline
\end{tabular} \quad
\begin{tabular}{||c||c||c||c||}
\hline
14
Table 9. Examples of filling of cells

<table>
<thead>
<tr>
<th>No</th>
<th>First Data</th>
<th>Second Data</th>
<th>Third Data</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

6 Cells (One-Column Tabulars) in the Text

If the \makecell command appears in the text (outside outer tabular environment) it could create wrong layout. The \makecellbox command and \mcellbox environment are created for one-column tabulars in the text.
7 Diagonally Divided Cell

This variant of head’s positioning is not too popular nowadays, but in the some cases it could be used. Instead of creating of multicolumn head above a wide couple of all column heads except the very left column, the most left column head (upper left cell) divided by diagonal line. The lower head is usually head of very left column and upper head—“multicolumn” to all other column heads of table to the right.

This package offers macro based on possibilities of picture environment.

\diaghead{(H ratio,V ratio)}{(Text set for column width)}%
{(First head)}{(Second head)}

where \((H ratio,V ratio)\) sets the ratios like in \line command (digits from 1 up to 6). This argument is optional, the default ratio \line direction defined as \((5,-2)\).

The \{(Text set for column width)\} defined by hand, for example: 1) sets the width, using longest text lines from both heads—in this case you must put \theadfont macro, if you use \thead’s; 2) the longest text from the rest of column; 3) \hskip(value), even \hskip\hsize the case of \(p\) column (or \(X\) column in tabularx environment). The \{(First head)\} is head in lower corner (usually for first or very left column), \{(Second head)\}—in the upper corner (head for the all right columns).

Here is code of table [10]

\makegapedcells
\begin{tabular}{|l|c|c|} \hline
\diaghead{\theadfont Diag Column Head II}{Diag Column Head I}{Diag Column Head II}& \thead{Second column}& \thead{Third column} \\
\hline...
end{tabular}
\medskip
\begin{tabularx}{.62\hsize}{|X|c|c|} \hline
\diaghead*(-4,1){\hskip\hsize}{Diag Column Head I}{Diag Column Head II}& \thead{Second column}& \thead{Third column} \\
\hline...
end{tabularx}
\medskip
\nomakegapedcells
\begin{tabular}{|l|c|c|} \hline
\diaghead\(4,1\){\hskip4.2cm}{Diag Column Head I}{Diag Column Head II}& \thead{Second column}& \thead{Third column} \\
\hline...
\end{tabular}

The correct position of diagonal ends depends of width of column. If cell width is narrower then necessary column ends of diagonal don’t touch corners of cell.
Table 10. Examples of tabulars with diagonally divided cells

<table>
<thead>
<tr>
<th>Diag Column Head I</th>
<th>Second column</th>
<th>Third column</th>
</tr>
</thead>
<tbody>
<tr>
<td>Left aligned cell text</td>
<td>A</td>
<td>37–43</td>
</tr>
<tr>
<td>Right aligned cell text</td>
<td>B</td>
<td>37–43</td>
</tr>
<tr>
<td>Bottom aligned cell text</td>
<td>C</td>
<td>52–58</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Diag Column Head I</th>
<th>Second column</th>
<th>Third column</th>
</tr>
</thead>
<tbody>
<tr>
<td>Left aligned cell text</td>
<td>A</td>
<td>37–43</td>
</tr>
<tr>
<td>Right aligned cell text</td>
<td>B</td>
<td>37–43</td>
</tr>
<tr>
<td>Bottom aligned cell text</td>
<td>C</td>
<td>52–58</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Diag Column Head I</th>
<th>Second column</th>
<th>Third column</th>
</tr>
</thead>
<tbody>
<tr>
<td>Left aligned cell text</td>
<td>A</td>
<td>37–43</td>
</tr>
<tr>
<td>Right aligned cell text</td>
<td>B</td>
<td>37–43</td>
</tr>
<tr>
<td>Bottom aligned cell text</td>
<td>C</td>
<td>52–58</td>
</tr>
</tbody>
</table>
8 Comparison: Diagonally Divided Cell with the \slashbox package

I didn’t know about the \slashbox package in time when the code for the diagonal cells was created. This package uses also picture \line mechanism, but calculates slopes automatically. It creates the \backslash direction of diagonal lines only. (The \slash direction seems to be not correct. Or direction depends to traditions of current publishing house?)

Here is used citation of the sample file distributed with the \slashbox package[1].

The verbatim of the table examples are added only in citation.

The usage is pretty straightforward, such as

\begin{tabular}{|l||*{5}{c|}}
\hline
\backslashbox{Room}{Date} & 5/31 & 6/1 & 6/2 & 6/3 & 6/4 \\
\hline
Meeting Room &\&\&\& & \\
Auditorium &\&\&\& & \\
Seminar Room &\&\&\& & \\
\hline
\end{tabular}

You may include a newline (\\) in ‘Room’ and/or ‘Date’. Note that you will get spaces aside the slash line if there is a wider column in the same column of a different line. In such a case, you need to specify the width of the slashed column by saying

\begin{tabular}{|l||*{4}{c|}}
\hline
\backslashbox[48mm]{Room}{Date} & 5/31 & 6/1 & 6/2 & 6/3 & 6/4 \\
\hline
Room With a Long Name &\&\&\& & \\
Auditorium &\&\&\& & \\
Seminar Room &\&\&\& & \\
\hline
\end{tabular}

The specified width is neglected if it is narrower than the natural width of the column.

`\backslashbox` assumes by default that there is a blank space of width `\tabcolsep` on both sides of the column. Thus the slash line might exceed the boundary when you use `@{}` etc.

You can avoid it by specifying

\begin{tabular}{|@{ $\bullet$ \hspace*{3mm}}l||*{5}{c|}}
\hline
\multicolumn{1}{|@{}l||}{\backslashbox[0pt][l]{Room}{Date}} & \makebox[3em]{5/31} & \makebox[4em]{6/1} & \makebox[3em]{6/2} & \makebox[3em]{6/3} & \makebox[3em]{6/4} \\
\hline
Meeting Room & & & & & \\
Auditorium & & & & & \\
Seminar Room & & & & & \\
\end{tabular}

Here `[l]` tells the command that there is no extra space on the left of this column. You can use `[r]` and `[lr]` likewise. You have to also specify the width of the column in this case, but it can be 0pt.

The addition for the sample with `\slashbox` command. This example demonstrates that you may combine the `makecell` commands and `\backslashbox`.

\begin{tabular}{|l||*{5}{c|}}
\hline
\theadset\theadfont\backslashbox{Room}{Date} & \makebox[3em]{\thead{5/31}} & \makebox[3em]{\thead{6/1}} & \makebox[3em]{\thead{6/2}} & \makebox[3em]{6/3} & \makebox[3em]{\thead{6/4}} \\
\hline
Meeting Room & & & & & \\
Auditorium & & & & & \\
Seminar Room & & & & & \\
\end{tabular}
<table>
<thead>
<tr>
<th>Room With a Long Name</th>
<th>5/31</th>
<th>6/1</th>
<th>6/2</th>
<th>6/3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Auditorium</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Seminar Room</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Room

Date

5/31

6/1

6/2

6/3
9 Thick Rules for the hline and cline Commands

For horizontal rules in tabular there were added two commands \Xhline and \Xcline. They use additional mandatory argument with defined rule width.

The example, with result in table 11.

\begin{table}
\renewcommand\theadset{\def\arraystretch{.85}}\%
\renewcommand\theadgape{}
ttabbox
{\caption{...}\label{...}}\%
{\begin{tabular}{!{\vrule width1.2pt}c
!{\vrule width1.2pt}c|c
!{\vrule width1.2pt}}
\Xhline{1.2pt}
\multirowthead{4}{First Column head}&
\multicolumn{2}{c!{\vrule width1.2pt}}{\thead{Multicolumn head}}\
\Xcline{2-3}{1.2pt}
& \thead{Second \multlined \ column head} &
\thead{Third \ column head}\
\Xhline{1.2pt}
Cell text & A &\multirowcell{4}{28--31}\
\Xhline{1.2pt}
\end{tabular}}
\end{table}
Table 11. Example of `tabular` environment with thick lines

<table>
<thead>
<tr>
<th>First Column head</th>
<th>Multicolumn head</th>
<th>Second multiline column head</th>
<th>Third column head</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cell text</td>
<td>A</td>
<td></td>
<td>28–31</td>
</tr>
<tr>
<td>Multilined cell text</td>
<td>B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Left aligned cell text</td>
<td>C</td>
<td></td>
<td>37–43</td>
</tr>
<tr>
<td>Right aligned cell text</td>
<td>D</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bottom aligned cell text</td>
<td>E</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cell long long long long long text with predefined width</td>
<td>F</td>
<td>37–43</td>
<td>52–58</td>
</tr>
<tr>
<td>Cell long long long long long text with predefined width</td>
<td>G</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
10 Code of package

10.1 Multilined cells

First goes request of \texttt{array} package.

\begin{verbatim}
\RequirePackage{array}
\end{verbatim}

\texttt{\makecell} The definition of command for multilined cells. At first defined \texttt{\gape} stuff. Non-star form loads special setting for vertical space around (if it used). Star form always creates additional vertical \texttt{\jot}-spaces.

\begin{verbatim}
2 \newcommand\makecell{\@ifstar{\let\tabg@pe\gape\makecell@}%
3 {\let\tabg@pe\cellgape\makecell@}}
\end{verbatim}

Next macro loads vertical and horizontal common alignment for cells and loads redefined spacing parameters \texttt{\arraystretch} and \texttt{\extrarowheight} if these parameters were redefined.

\begin{verbatim}
4 \newcommand\makecell@{\def\t@bset{\cellset}%
5 \let\mcell@align\cellalign
6 \@ifnextchar[\mcell@tabular
7 {\expandafter\mcell@@tabular\cellalign\@nil}}
\end{verbatim}

\texttt{\thead} The macro for tabular column heads. At first defined \texttt{\gape} stuff. Non-star from loads special setting for vertical space around (if it used). Star form always creates additional vertical \texttt{\jot}-spaces.

\begin{verbatim}
8 \newcommand\thead{\@ifstar{\let\tabg@pe\gape\thead@}%
9 {\let\tabg@pe\theadgape\thead@}}
\end{verbatim}

Next macro loads vertical and horizontal common alignment for column heads and loads redefined spacing parameters \texttt{\arraystretch} and \texttt{\extrarowheight} if these parameters were redefined. (First go settings for cells, as for \texttt{\makecell}, then special settings for column heads.)

For column heads also loaded font settings.

\begin{verbatim}
10 \newcommand\thead@{\def\t@bset{\cellset\theadfont\theadset}%
11 \let\mcell@align\theadalign
12 \@ifnextchar[\mcell@tabular
13 {\expandafter\mcell@@tabular\theadalign\@nil}}
\end{verbatim}

\texttt{\rotheadsize} The width dimension for rotated cells.

\begin{verbatim}
14 \@ifdefinable\rotheadsize{\newdimen\rotheadsize}
\end{verbatim}

\texttt{\rotcell} The macro for rotated cell. If no \texttt{rotating} package loaded this macro works like \texttt{\makecell}.

\begin{verbatim}
15 \newcommand\rotcell{\@ifundefined{\string\rotcell}{\PackageWarning{makecell}{{\texttt{rotcell}} space needs rotating package}%%%%%%%%%%%%%%%%%%}{% }% \let\tabg@pe\empty\let\t@bset{\cellset\makecell@}
16 {\@ifnextchar[}{\@@rotcell}}
\end{verbatim}
The macro for rotated tabular column heads. If no rotating package loaded this macro works like \thead.

\rothead

The macro for multirow cells. If no multirow package loaded this macro works like \makecell.

\multirowcell

These macros define settings for \multirow arguments.

\multirowthead

The macro for multirow column heads. If no multirow package loaded this macro works like \thead.
These macros define settings for \texttt{\multirow} arguments.

\begin{verbatim}
\texttt{\mcell@mrowhead} \texttt{\multirow}

\texttt{\mcell@multirow} \texttt{\multirow}

\texttt{\mcell@tabular} \texttt{\multirow}
\end{verbatim}

\texttt{\mcell@multirow} \texttt{\texttt{\multirow}}

\texttt{\mcell@tabular} \texttt{\texttt{\multirow}}

Definitions for horizontal and vertical alignments, which use by \texttt{\tabular} and \texttt{array} environments.

For \texttt{l}, \texttt{r}, \texttt{t}, and \texttt{b} alignments commands set \texttt{c}-argument as vertical or horizontal centering alignment if necessary. For \texttt{l} and \texttt{r} alignments also redefined alignment settings for \texttt{\makecell} (\texttt{\thead}) blocks.

\begin{verbatim}
\texttt{\mcell@l} \texttt{\mcell@r} \texttt{\mcell@t} \texttt{\mcell@b}
\end{verbatim}

Default vertical and horizontal alignment is centered.

\begin{verbatim}
\texttt{\mcell@i} \texttt{\mcell@ij} \texttt{\mcell@ic}
\end{verbatim}

Default horizontal alignment of \texttt{\makecell} (\texttt{\thead}) blocks is centered.

\begin{verbatim}
\texttt{\mcell@left} \texttt{\mcell@right}
\end{verbatim}

The core macros for \texttt{\tabular} building.

The code for this macro borrowed from \texttt{\caption} 3.x package (AS).
\@for\mcell@Tmp:={#2}\do{%
  \ifx\mcell@tmp\mcell@Tmp
    \let\next\@firstoftwo
  \fi
}
\next}

The \mcell@tabular macro at first calls \mcell@setalign macro for sorting of alignment arguments, then calls \mcell@@tabular macro, which created tabular cell or column head.

\def\mcell@@tabular#1#2\@nil#3{%
  \expandafter\mcell@setalign\mcell@align\@nil
  \mcell@setalign{#1}{#2}\@nil
  \expandafter\mcell@@@tabular\expandafter\mcell@i\mcell@ii\@nil{#3}%
}

\mcell@setalign
This macro sorts arguments for vertical and horizontal alignment.

First argument has second check at the end of macro for the case if it is c-argument.

\@ifdefinable\mcell@setalign{}
\def\mcell@setalign#1#2\@nil{
  \def\@tempa{#1}
  \def\@tempc{c}
  Restore default alignment for \makecell and \thead blocks.

  \global\let\mcell@left\hfill
  \global\let\mcell@right\hfill

  If in optional argument appears alone c-argument it defines horizontal centering only.

  \def\mcell@c{
    \def\mcell@ii{c}%
    \mcell@ifinlist{#1}{l,r,t,b,c,}{\@nameuse{mcell@#1}}%
    \mcell@ifinlist{#2}{l,r,t,b,c,}{\@nameuse{mcell@#2}}%
    Here goes repeated check for first argument, if it is c-argument we call \mcell@c command, which can be now redefined.

    \ifx\@tempa\@tempc\mcell@c\fi
  }%
}

\mcell@@@tabular
This macro builds tabular itself. First (and last) go commands which align \makecell and \thead blocks like l, r, or c (if they loaded). Then goes check whether math mode exists. The \mcell@multirow emulation macro transforms to \multirow when necessary.

\def\mcell@multirow#1#2\@nil#3\%{
  \hbox{\t@bset\$\array[#1]{@{}#2@{}}#3\endarray$}}%
The layout macros for tabular building settings.

Spacing settings for tabular spacing inside cells (like \arraystretch or \extrarowheight).

Vertical space around cells (created by \gape stuff).

Vertical and horizontal alignment of cell text.

Angle for rotated column heads and cells.

Font for column heads

Special spacing settings for tabular spacing in column heads (like \arraystretch or/and \extrarowheight).

Vertical space around column heads (created by \gape stuff).

Vertical space around rotated column heads.

Vertical and horizontal alignment of column head text.
10.2 Gape commands

The macro itself. It uses analogous to \smash macro from amsmath package.

\begin{verbatim}
\@ifdefinable\gape{}
\DeclareRobustCommand\gape{\@ifnextchar[\@gape{\@gape[tb]}}
\end{verbatim}

The \setcellgapes defines settings used by \makegapedcells command.
First goes check for optional argument.

\begin{verbatim}
\newcommand\setcellgapes{\@ifnextchar[\%}{\@gape\jot}{\@gape\jot}
\end{verbatim}

The body of settings.

\begin{verbatim}
\@ifdefinable\@setcellgapes{}
\newcommand\setcellgapes{\@ifnextchar[\%}{\@gape\jot}{\@gape\jot}
\end{verbatim}

Negative compensate inside \makegapedcells area.

\begin{verbatim}
\newcommand\negjot[1]{{\jot\mcell@MBnegjot\gape\jot[\mcell@MBnegtb]{#1}}}
\end{verbatim}

The macros which count advanced height and depth of boxes.

\begin{verbatim}
\newcommand\mcell@mb@t[2]{\@tempdima\ht#1\advance\@tempdima#2\ht#1\@tempdima}
\newcommand\mcell@mb@b[2]{\@tempdimb\dp#1\advance\@tempdimb#2\dp#1\@tempdimb}
\newcommand\mcell@mb@tb[2]{\mcell@mb@t{#1}{#2}\mcell@mb@b{#1}{#2}}
\end{verbatim}

The body of \gape macros.

\begin{verbatim}
\@ifdefinable\@gape{}
\@ifdefinable\@@gape{}
\def\@gape\jot{\@gape\jot}
\def\@@gape{\ifmmode\expandafter\mathpalette\expandafter\mathg@pe
\else\expandafter\makeg@pe\fi}
\end{verbatim}

The macros which put box with necessary parameters in text and math mode.

\begin{verbatim}
\newcommand\makeg@pe[1]{\setbox\z@\hbox{\color@begingroup#1\color@endgroup}}
\newcommand\mathg@pe[2]{\setbox\z@\hbox{$\m@th#1{#2}$}}
\end{verbatim}

\begin{verbatim}
\@ifdefinable\Gape{}\@ifdefinable\@@Gape{}
\def\@Gape\jot{\@Gape\jot}\def\@@Gape{\ifnum8\ht\z@#1\@tempdima\ht\z@\advance\@tempdima#1\ht\z@\@tempdima
\@tempdimb\dp\z@\advance\@tempdimb#2\dp\z@\@tempdimb}
\end{verbatim}

The macros which put box with necessary parameters in text and math mode.

\begin{verbatim}
\DeclareRobustCommand\Gape{\@ifnextchar[\@Gape{\@Gape[\jot]}}
\end{verbatim}
The macros abbreviations for \strut which changed by value of \jot. First enlarges both depth and height.

Second enlarges only height.

Third enlarges only depth.

10.3 Modification of command from array package

At first is saved \@classz macro.

This macros redefine and restore the \@classz macro from array package.

Following macro creates tabular/array cells with changed vertical spaces.

Redefined \@classz macro from array package.

The row of empty cells.

10.4 Rows of skipped and numbered cells

The row of empty cells.
\DeclareRobustCommand\eline[1]{\@temptokena{}\count@ #1\%}
\advance\count@\m@ne
\loop\@temptokena\expandafter{\the\@temptokena&}\%\advance\count@\m@ne\ifnum\count@>\z@\repeat
\the\@temptokena\ignorespaces}

\erows Lazy macros for filling few rows.
\Xrows
\% newtoks\@temptokena\newtoks\@temptokenb
\% newcommand\erows[3][\line\hline]{\relax@temptokenb{}\@temptokenc{}\@tempcnta#3\relax
\% \count@#2\advance\count@\m@ne\loop\showthe\@temptokenb
\% \@temptokenb\expandafter\the\@temptokenb&\%
\% \advance\count@\m@ne\ifnum\count@>\z@\repeat
\% \the\@temptokenb\showthe\@temptokenb}
\% newcommand\erows[3][\line\hline]{\@temptokena{}\count@ #3\%
\% \loop\@temptokena\expandafter\the\@temptokena\eline{#2}#1\%
\% \advance\count@\m@ne\ifnum\count@>\z@\repeat
\% \the\@temptokena}
\% newcommand\Xrows[2][\line\hline]{\@temptokena{}\count@ #2\%
\% \loop\@temptokena\expandafter\the\@temptokena\eline{#1}\%
\% \advance\count@\m@ne\ifnum\count@>\z@\repeat
\% \the\@temptokena}

\rnline The rows of numbered cells. The \rnline command replaces \Alph and \alph
\nline counter by \Asbuk and \asbuk consequently.
\nlne

\newcounter{nlinenum}
\@ifdefinable\rnline{}
\DeclareRobustCommand\rnline{\gdef\TeXr@rus{\let\@Alph\@Asbuk\let\@alph\@asbuk}\@nline}
\@ifdefinable\nline{}
\DeclareRobustCommand\nline{\gdef\TeXr@rus{}\@nline}
\newcommand\@nline{\@ifnextchar[\]{\@@nline}{\@@nline[1]}}
\@ifdefinable\@@nline{}
\def\@@nline[#1]{\@ifnextchar[\]{\@@@nline[#1]}{\@@@nline[#1][1]}}
\@ifdefinable\@@@nline{}
\def\@@@nline[#1][#2]#3{\count@#3\def\TeXr@label{\TeXr@label@{nlinenum}}%\expandafter\TeXr@loop\@gobble{}#1\@@@
\xdef\Num{\the\TeXr@lab}%%\c@nlinenum#2\relax%\expandafter\@temptokena\expandafter{\Num %\global\advance\c@nlinenum\@ne}%%\advance\count@\m@ne%\loop\@temptokena\expandafter{\the\@temptokena&\Num %\global\advance\c@nlinenum\@ne%\advance\count@\m@ne\ifnum\count@>\z@\repeat
\% global@\@emptokenb\expandafter\the\@emptokenb\expandafter\the\@emptokenb#1%\% \advance\count@\m@ne\ifnum\count@>\z@\repeat
\% \the@emptokenb\showthe@emptokenb}
\% newcommand\erows[3][\line\hline]{\@emptokena{}\count@ #3%\% \loop\@emptokena\expandafter\the@emptokena\eline{#2}#1%\% \advance\count@\m@ne\ifnum\count@>\z@\repeat
\% \the@emptokena}
\% newcommand\Xrows[2][\line\hline]{\@emptokena{}\count@ #2%\% \loop\@emptokena\expandafter\the@emptokena\eline{#1}%\% \advance\count@\m@ne\ifnum\count@>\z@\repeat
\% \the@emptokena}

\the\@emptokena\ignorespaces

[Borrowed code stuff and explanation from \texttt{enumerate/paralist} packages just with changes of command names.]

Internal token register used to build up the label command from the optional argument.

\newtoks\TeXr@lab
This just expands to a ‘?’. \texttt{\ref} will produce this, if no counter is printed.

\def\TeXr@qmark{?}
The next four macros build up the command that will print the item label. They each gobble one token or group from the optional argument, and add corresponding tokens to the register \texttt{@enLab}. They each end with a call to \texttt{@enloop}, which starts the processing of the next token.

\TeXr@label Add the counter to the label. \#2 will be one of the ‘special’ tokens A a I i 1, and is thrown away. \#1 will be a command like \texttt{\Roman}.\def\TeXr@label@#1#2#3{\edef\TeXr@the{\noexpand#2{#1}}\TeXr@lab\expandafter{\the\TeXr@lab\TeXr@rus\csname the#1\endcsname}\advance\@tempcnta1\TeXr@loop}

The only foreign command in this stuff. It indicates whether the list has numeration by Russian letters.

\def\TeXr@rus{}
\TeXr@space\TeXr@sp@ce
Add a space to the label. The tricky bit is to gobble the space token, as you cannot do this with a macro argument.
\def\TeXr@space{\afterassignment\TeXr@sp@ce\let\@tempa=}\def\TeXr@sp@ce{\TeXr@lab\expandafter{\the\TeXr@lab\space}\TeXr@loop}

\TeXr@group Add a \{ \} group to the label.
\def\TeXr@group#1{\TeXr@lab\expandafter{\the\TeXr@lab{#1}}\TeXr@loop}

\TeXr@other Add anything else to the label
\def\TeXr@other#1{\TeXr@lab\expandafter{\the\TeXr@lab#1}\TeXr@loop}

\TeXr@loop The body of the main loop. Eating tokens this way instead of using \texttt{@tfor} lets you see spaces and all braces. \texttt{@tfor} would treat a and \{a\} as special, but not \{{a}\}.
\def\TeXr@loop{\futurelet\TeXr@temp\TeXr@loop@}
\def\TeXr@loop@{%
  \IfExistsA{\TeXr@temp}{\def\@tempa{\TeXr@label\Alph}\} 
  \Else{
    \IfExistsA{\TeXr@temp}{\def\@tempa{\TeXr@label\alph}\} 
    \Else{
      \IfExists\{\TeXr@temp\} {\def\@tempa{\TeXr@label\roman}\} 
      \Else{
        \IfExists\{I\}\{\TeXr@temp\} {\def\@tempa{\TeXr@label\Roman}\} 
        \Else{
          \@tempa}}}}
Hook for possible extensions

\TeX\hook

\providecommand\TeX\hook{}

10.5 Diagonally separated column heads

\diaghead Macro for diagonally separated column heads.

\newcommand\diaghead[3]{\hbox\bgroup\expandafter\mcell@getcelldiagratios\celldiagratio\relax\@tempswafalse\ifnum\mcell@Hratio<0\count@-\mcell@Hratio\relax\edef\mcell@Hratio{\the\count@}\relax\ifnum\mcell@Vratio<0\count@-\mcell@Vratio\relax\edef\mcell@Vratio{\the\count@}\@tempswatrue\fi\else\ifnum\mcell@Vratio<0\count@-\mcell@Vratio\relax\edef\mcell@Vratio{\the\count@}\@tempswatrue\else\fi\fi\settowidth\@tempdima{#1}\advance\@tempdima2\tabcolsep\edef\mcell@diagH{\the\@tempdima}\divide\@tempdima\mcell@Hratio\edef\mcell@diagV{\the\@tempdima}\relax\let\mcell@oriunitlength\unitlength\let\unitlength\relax\kern-\tabcolsep\kern-\@wholewidth\setbox\z@\hbox{\theadfont\strut}\@tempdima\dp\z@\advance\@tempdima.8\p@%2\@wholewidth\advance\@tempdima.8\p@%2\@wholewidth

The value of compensate vertical spacing defined here experimentally and equals to 2 default line thickness.
If \texttt{\textbackslash makedgapedcells} switched on for the table there is compensate spacing.

\begin{verbatim}
\ifx\@classz\mcell@classz
\setbox\z@\hbox{#1}\ht\z@\z@\dp\z@\z@
\mcell@MB@\z@\mcell@MB@\jot
\global\dimen@\@tempdima\global\@tempdimb\@tempdimb
\else\global\dimen@\z@\global\@tempdimb\z@\fi
\advance\@tempdima\dimen@
\edef\mcell@diagVoffset{\the\@tempdima}
\@tempdima\mcell@diagV\advance\@tempdima-\mcell@diagVoffset
\advance\@tempdima-\@tempdimb
\edef\mcell@diagVcorr{\the\@tempdima}
\noindent\nomakegapedcells\hbox{\begin{tabular}{@{}c@{}}%
At least a \texttt{\textbackslash normallineskip} vertical space from top and bottom of cell.
\ifdim\jot<2\p@\jot2\p@\fi
\if@tempswa
For South-East or North-West directions.
\begin{picture}(\mcell@diagH,\mcell@diagVcorr)(\z@,\mcell@diagVoffset)
\put(\z@,\mcell@diagV){\makebox(\z@,\z@)[tl]{%
{\edef\tempa{((\mcell@Hratio,-\mcell@Vratio))}\expandafter
\line\tempa{\mcell@diagH}}}
\put(\tabcolsep,\jot){\makebox(\z@,\z@)[tl]{\theadfont
\let\cellset\theadset\makecell[tl]{\strut#2}}}
\@tempdima\mcell@diagV\advance\@tempdima-\jot
\@tempdimb\mcell@diagH\advance\@tempdimb-\tabcolsep
\put(\@tempdimb,\@tempdima){%
{\makebox(\z@,\z@)[tr]{\theadfont
\let\cellset\theadset{\makecell[tr]{#3\strut}}}}
\end{picture}%
\else
\end{tabular}%
\end{verbatim}

For South-West or North-East directions.

\begin{verbatim}
\begin{picture}(\mcell@diagH,\mcell@diagVcorr)(\z@,\mcell@diagVoffset)
\put(\z@,\mcell@diagV){\makebox(\z@,\z@)[tl]{%
{\edef\tempa{((\mcell@Hratio,\mcell@Vratio))}\expandafter
\line\tempa{\mcell@diagH}}}
\put(\tabcolsep,\z@){\makebox(\z@,\z@)[bl]{\theadfont
\let\cellset\theadset\makecell[bl]{\strut#2}}}
\@tempdima\mcell@diagH\advance\@tempdima-\tabcolsep
\@tempdimb\mcell@diagV\advance\@tempdimb-\jot
\put(\@tempdima,\@tempdimb){%
{\makebox(\z@,\z@)[tr]{\theadfont
\let\cellset\theadset{\makecell[tr]{#3\strut}}}}
\end{picture}%
\else
\end{verbatim}

\end{tabular}
Macro used by previous one. Extracts ratios for defining of height of cell.

\iffalse
\if@edefinable\mcell@getcelldiagratios{}
\def\mcell@getcelldiagratios(#1,#2){\def\mcell@Hratio{#1}\def\mcell@Vratio{#2}}

10.6 The $\texttt{\hline}$ and $\texttt{\cline}$ with necessary thickness

\texttt{\bf \hline} The commands for $\texttt{\hline}$ and $\texttt{\cline}$ with necessary thickness. Added code for the \texttt{longtable} environment.

\newcommand{\Xhline}[1]{\noalign{\ifnum0='}\fi\arrayrulewidth#1\ifx\hline\LT@hline\let\@xhline\LT@@hline\fi\hrule\@height\arrayrulewidth\futurelet\reserved@a\@xhline}

\texttt{\bf \Xcline}

\newcommand{\Xcline}{\def\Xcline#1#2;#3\@nil}{\omit\@multicnt#1\advance\@multispan\m@ne\ifnum\@multicnt=\@ne\@firstofone{&\omit}\fi\@multicnt#2\advance\@multicnt-#1\advance\@multispan\@ne\leaders\hrule\@height#3\hfill\cr\noalign{\vskip-#3}}