

Table Preparation I

A table is used for presenting data or items row- and column-wise in a concise form. In L^AT_EX, the **tabular**, **tabularx**, and **longtable** environments are used for preparing different types of tables. However, tables produced by the **tabular** and **tabularx** environments cannot be assigned any serial number or title, which are generally required to identify a table. Moreover, they produce a table as a single object in running texts, which may cause a problem in drawing a big table in the limited space of a page. All such drawbacks can be overcome by nesting the **tabular** and **tabularx** environments with another environment, such as the **table**, **wraptable**, or **sidewaystable** environment (a serial number and a title to the table under the **longtable** environment can be assigned directly).

7.1 Table Through the **tabular** Environment

Tables are widely prepared through the **tabular** environment, where the columns of a table are generated through the mandatory argument of the environment. For example, `\begin{tabular}{|l|c|c|c|c|}` in Table 7.1 generates a five-column table

Table 7.1 A simple table through the **tabular** environment

L ^A T _E X input	Output																				
<pre> \begin{table}[!hbt] \centering \caption{Obtained marks.} \label{tab-marks} \begin{tabular}{ l c c c c } \hline Name & Math & Phy & Chem & English\ \hline Robin & 80 & 68 & 60 & 57\ \hline Julie & 72 & 62 & 66 & 63\ \hline Robert & 75 & 70 & 71 & 69\ \hline \end{tabular} \end{table} % Table~\ref{tab-marks} shows the ... </pre>	<div style="text-align: center; margin-bottom: 10px;">Table 1: Obtained marks.</div> <table border="1" style="margin: auto; border-collapse: collapse; text-align: center;"> <thead> <tr> <th>Name</th> <th>Math</th> <th>Phy</th> <th>Chem</th> <th>English</th> </tr> </thead> <tbody> <tr> <td>Robin</td> <td>80</td> <td>68</td> <td>60</td> <td>57</td> </tr> <tr> <td>Julie</td> <td>72</td> <td>62</td> <td>66</td> <td>63</td> </tr> <tr> <td>Robert</td> <td>75</td> <td>70</td> <td>71</td> <td>69</td> </tr> </tbody> </table> <p>Table 1 shows the marks obtained by three students in the final examination.</p>	Name	Math	Phy	Chem	English	Robin	80	68	60	57	Julie	72	62	66	63	Robert	75	70	71	69
Name	Math	Phy	Chem	English																	
Robin	80	68	60	57																	
Julie	72	62	66	63																	
Robert	75	70	71	69																	

(`\begin{tabular}[]{}{}` may also be used with optional provision in `[]` for vertical positioning). A column can be generated through one of the three letters of **l**, **r**, and **c** (other types of columns are discussed in §7.2 and §7.5). Each of these letters represents a column as well as the alignment of the entries in that column (**l** for left alignment, **r** for right alignment, and **c** for center alignment). The `|` symbol in the argument of `\begin{tabular}{}{}` is used either to mark a boundary or to separate two columns by a vertical line in the specified location, covering the full height of the table. Following the `\begin{tabular}{}{}` command, the column-wise entries of a row are inserted, separating two entries by an `&` and ending the row by a line break command `\`. Further, the `\hline` command is used either to mark a boundary or to separate two rows by a horizontal line in the specified location, covering the full width of the table (a `\hline` command before a row draws a horizontal line above the row). Finally, the `tabular` environment is ended by the `\end{tabular}` command (the `\hline` command just above the `\end{tabular}` command draws the lower horizontal boundary of the table).

Note that the `tabular` environment in Table 7.1 is nested inside the `table` environment for creating the table in a separate paragraph as well as for captioning and labeling it. The `table` environment is first created through the `\begin{table}[]!hbt` command (the optional argument `!hbt` is for the preferred vertical positioning of the table, which is explained in §7.3). The next command in Table 7.1 is `\centering`, which instructs for width-wise center alignment of the table (other commands could be `\flushleft` for left alignment or `\flushright` for right alignment). The `\caption{att1}`¹ command used in the `table` environment (but outside the `tabular` environment) assigns a serial number to the table preceded by the default label-word `Table` and followed by a colon, along with its argument `att1` as the title (caption) of the table (since the title usually comes on the top of a table, the `\caption{}` command is used before the `tabular` environment). Following the `\caption{}` command, the `\label{}` command is inserted with a unique reference key, which as shown in Table 7.1 can be used in the `\ref{}` command for referring the table anywhere in the document. Also note that `\label{}` is always used after `\caption{}`. Moreover, `\label{}` does not have any effect without `\caption{}`, in which case the table is not assigned any serial number.

7.2 Table Through the `tabularx` Environment

In the `tabular` environment discussed in §7.1, a column is generated by one of the options of **l**, **c**, and **r**. The width of a column under any of these options is made equal to the length of the longest entry in that column. This may extend a table even beyond the width of a page if the table has some very long entries.

The `tabularx` package provides the `tabularx` environment, which can calculate automatically the width of a column so as to restrict a table within

¹If the `\caption{att1}` command is used inside the `table`, `longtable`, or `sidewaystable` environment (but outside the `tabular` or `tabularx` environment), the table is assigned a serial number along with the argument `att1` as the title (caption) of the table.

a pre-specified horizontal width irrespective of the lengths of the entries in the table. The `tabularx` environment takes two mandatory arguments, i.e., `\begin{tabularx}{awidth}{acols}`, where *awidth* is the horizontal width of the table and *acols* is its columns. The columns in the `tabularx` environment are generated in the same way as in the `tabular` environment. A fixed-width column is generated through `l`, `c`, or `r`, while a `X` is used to generate a flexible-width column (i.e., a column whose width is to be calculated automatically)². All the flexible-width columns of a table are of equal width, which is calculated internally as the difference of the total width (*awidth*) of the table and total width of the fixed-width columns, divided by the number of flexible-width columns. Entries in a flexible-width column are made full aligned. Other alignments can be obtained using either `>{\raggedright\arraybackslash}`, `>{\centering\arraybackslash}`, or `>{\raggedleft\arraybackslash}` before `X`, which make the entries left, center, and right aligned, respectively (without the `\arraybackslash` command, the line breaking command `\` used for terminating a row may not work properly in some cases). Table 7.2 shows an application of the `tabularx` environment

Table 7.2 A simple table through the `tabularx` environment

L ^A T _E X input	Output																		
<pre> \begin{table}[!hbt] \centering \caption{Scored points.} \begin{tabularx}{0.8\linewidth} { X c >{\raggedleft\arraybackslash}X } \hline {\bf Name} & {\bf Sex} & {\bf Points}\ \hline Milan & M & 1,500\ Julie & F & 1,325\ Sekhar & M & 922\ Dipen & M & 598\ Rubi & F & 99\ \hline \end{tabularx} \end{table> </pre>	<p>Table 2: Scored points.</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th>Name</th> <th>Sex</th> <th>Points</th> </tr> </thead> <tbody> <tr> <td>Milan</td> <td>M</td> <td>1,500</td> </tr> <tr> <td>Julie</td> <td>F</td> <td>1,325</td> </tr> <tr> <td>Sekhar</td> <td>M</td> <td>922</td> </tr> <tr> <td>Dipen</td> <td>M</td> <td>598</td> </tr> <tr> <td>Rubi</td> <td>F</td> <td>99</td> </tr> </tbody> </table>	Name	Sex	Points	Milan	M	1,500	Julie	F	1,325	Sekhar	M	922	Dipen	M	598	Rubi	F	99
Name	Sex	Points																	
Milan	M	1,500																	
Julie	F	1,325																	
Sekhar	M	922																	
Dipen	M	598																	
Rubi	F	99																	

for generating a three-column table of a total width of 80% of the page width, i.e., `0.8\linewidth` (a fixed value, say `10cm` or `6in`, can also be used). Since the middle column is generated by the option `c`, its width is fixed by the longest entry in that column. The extreme two columns are generated by the option `X`, for which their widths are equal and calculated internally to accommodate all the three columns in

²In the `tabularx` environment, a fixed-width column is generated through `l`, `c`, or `r`, while a flexible-width column is generated through a `X`.

the pre-specified width (i.e., 0.8\linewidth) of the table. Moreover, the last column is made right aligned by generating it through `>\raggedleft\arraybackslash`**X**, instead of just through **X**. All other matters of Table 7.2 are same with those of Table 7.1.

7.3 Vertical Positioning of Tables

As shown in Tables 7.1 and 7.2, the preferred vertical position of a table on a page can be specified as an optional argument to the `table` environment, i.e., `\begin{table}[avp]`, where `avp` is the specifier for vertical positioning of the table. The commonly used specifiers are **h**, **b**, and **t**, which stand for here, bottom of the page, and top of the page, respectively. These specifiers can be used individually or in a combination of two or three. Moreover, for placing the table in the specified position even if enough space is not available on the current page, the specifier or the combination of the specifiers may be preceded by a **!** symbol, like **!h**, **!b**, or **!hbt**. Irrespective to the order of the specifiers in a combination, L^AT_EX always uses the following order for positioning a table:

- ▷ If **!** is used, many default or preset restrictions are ignored and a table is attempted to put in the specified position.
- ▷ If **h** is given, the table is attempted to put in the exact position. If fails and no more specifier is given, by default L^AT_EX considers the specifier **t** for placing the table on the top of the next page.
- ▷ If **t** is given, the table is attempted to put on the top of the current page.
- ▷ If **b** is given, the table is attempted to put at the bottom of the current page.

Besides **h**, **b**, and **t**, there is another specifier **H**, which is defined in the `float` package. Usually, if a table cannot be put on the current page due to space limit, it is taken to the next page and the remaining space of the current page is filled by the texts which are typed in the L^AT_EX input file after the table. However, **H** instructs to put a table *here* only. If the blank space on the current page is not sufficient to hold the table, it is taken to the top of the next page along with the texts that follow the table, by leaving the current page incomplete. The specifier **H** is used alone, i.e., it should not be combined with **!** or any of **h**, **b**, and **t**. Refer §8.9 on the page 80 for putting all the tables at the end of a document, regardless of their actual positions in the L^AT_EX input file.

7.4 Sideways (Rotated) Texts in Tables*

If a table contains some long entries, space can be saved by printing such entries in vertical direction through the `sideways` environment defined in the `rotating` package. An application of the `sideways` environment is shown in Table 7.3 on the next page.

Table 7.3 Table with entries in vertical direction

L ^A T _E X input	Output			
<pre> \begin{tabular}{ l c c c } \hline Name & \begin{sideways}Mathematics\end{sideways}& & \begin{sideways}Physics\end{sideways}& & \begin{sideways}Chemistry\end{sideways} \\ \hline Robin & 80 & 68 & 60 \\ \hline Julie & 72 & 62 & 66 \\ \hline Robert & 75 & 70 & 71 \\ \hline \end{tabular} </pre>	Name	Mathematics	Physics	Chemistry
	Robin	80	68	60
	Julie	72	62	66
	Robert	75	70	71

7.5 Adjusting Column Width in Tables*

The width of a column, generated through `l`, `r`, or `c`, is set automatically based on the length of the longest entry in that column. This may suffer from the drawback of extending a table beyond the page width if the table has some long entries (many users tackle the situation by manually splitting a long entry into multiple rows). On the other hand, the `X` option in the `tabularx` environment generates columns of equal width irrespective of the lengths of their entries. This may also suffer from the disadvantage of allocating excess width to columns having short entries only, while some columns not having sufficient width to accommodate their long entries.

Above drawbacks can be alleviated by generating columns of a table through `p{}`, `m{}`, or `b{}` command defined in the `array` package. The arguments of `p{}`, `m{}`, and `b{}` specify the width of a column, and the letters `p`, `m`, and `b` make an entry, respectively, vertically top aligned, middle aligned, and bottom aligned relative to the alignment in the previous column (entries in all the three cases are horizontally full aligned). Applications of these three options for fixing the width of a column are shown in Table 7.4 on the next page. The same vertical alignment is made to all the three columns in the first three cases. While in the fourth case, three different alignments are made to the three columns of the table. The first column in the fourth case is top aligned (`p{}`), middle one is middle aligned (`m{}`), and the last one is bottom aligned (`b{}`). As a result, the vertical alignment of a column has become relative to that of its previous column. The second column is middle aligned about the top line of the first column which is top aligned. Similarly, the middle of the second column is made the bottom of the third column which is bottom aligned.

Fixing the width of a column by an absolute value, like `p{1.5cm}`, may make a table too small or extending beyond the width of the page (or the column of a multi-column document), particularly if the page or font size is changed in a later stage. Therefore, a good practice would be to fix the width of a column as a fraction of the `\linewidth` command in a single-column document and `\columnwidth` command in a multi-column document³, e.g., `p{0.3\linewidth}` or `m{0.2\columnwidth}`.

³A good practice would be to specify a length as a fraction of `\linewidth` in a single-column document and `\columnwidth` in a multi-column document (instead of a fixed length, like `5mm`), particularly to avoid any unpleasant output upon changing the page or font size in a later stage.

Table 7.4 Fixing column widths in tables with `p{}`, `m{}`, and `b{}`

L ^A T _E X input	Output			
<pre>\begin{tabular}{ p{1.7cm} p{1.5cm} p{1.6cm} } \hline This is the first and the ...& A medium size entry& \hline This is another long entry\\ \end{tabular}</pre>	<table border="1"> <tr> <td>This is the first and the longest entry</td> <td>A medium size entry</td> <td>This is another long entry</td> </tr> </table>	This is the first and the longest entry	A medium size entry	This is another long entry
This is the first and the longest entry	A medium size entry	This is another long entry		
<pre>\begin{tabular}{ m{1.7cm} m{1.5cm} m{1.6cm} } \hline This is the first and the ...& A medium size entry& \hline This is another long entry\\ \end{tabular}</pre>	<table border="1"> <tr> <td>This is the first and the longest entry</td> <td>A medium size entry</td> <td>This is another long entry</td> </tr> </table>	This is the first and the longest entry	A medium size entry	This is another long entry
This is the first and the longest entry	A medium size entry	This is another long entry		
<pre>\begin{tabular}{ b{1.7cm} b{1.5cm} b{1.6cm} } \hline This is the first and the ...& A medium size entry& \hline This is another long entry\\ \end{tabular}</pre>	<table border="1"> <tr> <td>This is the first and the longest entry</td> <td>A medium size entry</td> <td>This is another long entry</td> </tr> </table>	This is the first and the longest entry	A medium size entry	This is another long entry
This is the first and the longest entry	A medium size entry	This is another long entry		
<pre>\begin{tabular}{ p{1.7cm} m{1.5cm} b{1.6cm} } \hline This is the first and the ...& A medium size entry& \hline This is another long entry\\ \end{tabular}</pre>	<table border="1"> <tr> <td>This is the first and the longest entry</td> <td>A medium size entry</td> <td>This is another long entry</td> </tr> </table>	This is the first and the longest entry	A medium size entry	This is another long entry
This is the first and the longest entry	A medium size entry	This is another long entry		

In the columns of a table, entries are printed leaving some blank space on both sides defined by the `\tabcolsep` command. The length of such a horizontal blank space between two columns can be changed by changing the value of `\tabcolsep` (default is 6pt), e.g., `\setlength{\tabcolsep}{2mm}`. Similarly, the blank space before or after a particular entry can be eliminated using `@{}`, e.g., `\begin{tabular}{|@{}|@{}|@{}}` will omit blank space on either side of a table, and `\begin{tabular}{|!@{}|!@{}|!}` will omit the blank space between the two columns (`@{~}` can also be used for leaving a blank space of length equal to that of `~`). On the other hand, the indentation of an entry can be increased by redefining the length of the `\parindent` command (default is 0pt), e.g., `>\setlength{\parindent}{5mm}p{}` will generate a column, in which entries will be indented by 5mm.

7.6 Additional Provisions for Customizing Columns of Tables*

Besides the provisions discussed in §7.1–7.5, the `tabular` and `tabularx` environments have many more provisions for customizing a table, some of which are outlined here (all of these provisions are defined in the `array` package).

- ▷ The style of the entries in a particular column can be altered using `>{command}` before the column-generating option `l`, `c`, `r`, `X`, `p{}`, `m{}`, or `b{}`. For example, `>\bfseries|` for printing all the entries of that column in boldface fonts, or `>\centering}p{5cm}` for making the entries center aligned.

- ▷ A column-generating option can be preceded and followed by `>{ $\}$` and `<{ $\}$` , respectively, for converting the column into math-mode, e.g., `>{ $\}$!<{ $\}$` will generate a left-aligned math-mode column so that a mathematical expression can be inserted in that column without creating any more math-mode.
- ▷ For repeated use of a particular type of column, a new column type can be defined in the preamble through the `\newcolumntype{<math><math>}` command. For example, `\newcolumntype{C}{>{ $\}$ c<{ $\}$ }` can be used for generating directly a center-aligned math-mode column with `C`, or `\newcolumntype{R}{>{\raggedleft\arraybackslash}X}` for generating a right-aligned flexible-width column with `R`.
- ▷ Instead of repeating a column type for generating multiple number of consecutive columns of the same type, `*{<math><math>}` may be used, which means `n` number of columns of type `c<math><math>`. For example, `\begin{tabular}{|l|*{5}{c|}}` will generate a left-aligned column first and then five number of center-aligned columns, with vertical lines on both sides of each column (Table 7.6 on page 67 shows an application).
- ▷ For changing the width of a column-separating vertical line (default width is 0.4pt), the `|` sign may be replaced by `!{\vrule width <math><math>}` with `aval` as the width of the vertical line, e.g., `!{\vrule width 0.9mm}` will generate a vertical line of 0.9mm width.
- ▷ The width of vertical and horizontal lines created by `|`, `\vline`⁴, `\hline` or `\cline{<math><math>}` can be controlled by setting the value of the `\arrayrulewidth` command (default value is 0.4pt), e.g., `\setlength{\arrayrulewidth}{2pt}` for obtaining 2pt thick lines.
- ▷ The `booktabs` package provides some commands for drawing horizontal lines of different widths, as well as of different spacings below or above a horizontal line. These commands include `\toprule[<math><math>]`, `\midrule[<math><math>]`, `\bottomrule[<math><math>]`, and `\addlinespace[<math><math>]`, where the width or spacing value, as applicable, is taken as the argument in `[<math><math>]`. For example, `\toprule[3pt]`, `\midrule[1pt]`, and `\bottomrule[2pt]` (instead of `\hline`) for producing the top, middle, and bottom lines of a table of widths 3pt, 1pt, and 2pt, respectively. On the other hand, `\toprule[3pt]\addlinespace[2pt]` for leaving 2pt blank space below the top line, or `\addlinespace[1pt]\bottomrule[2pt]` for a blank space of 1pt above the bottom line.
- ▷ The vertical space between a column entry and a horizontal line, produced by `\hline` or `\cline{<math><math>}` (refer §7.7 for `\cline{<math><math>}`), is controlled by the `\extrarowheight` command defined in the `tabularx` package. A suitable value can be assigned to `\extrarowheight` (default value is 0pt) for increasing such space, e.g., `\setlength{\extrarowheight}{3mm}` for creating an extra space of 3mm. The `\setlength{\extrarowheight}{<math><math>}` command is to be placed before starting the `tabular` or `tabularx` environment.

⁴The `\vline` command in the `tabular` and `tabularx` environments draws a vertical line, in the place of its application, having a height equal to that of a row.

Table 7.5 Some additional provisions for customizing a table

L ^A T _E X input	Output														
<pre> \begin{table}[hbt] \centering \setlength{\extrarowheight}{4mm} \begin{tabular}{ !{\vrule width 0.8mm}>{\bfseries} >{\\$} <{\\$}!{\vrule width 0.8mm}} \hline Length & & {\bf In Metre}\! \hline 1 Millimeter & & 10^{-3}\! \hline 1 Centimeter & & 10^{-2}\! \hline 1 Decimeter & & 10^{-1}\! \hline 1 Decameter & & 10\! \hline 1 Hectometer & & 10^2\! \hline 1 Kilometer & & 10^3\! \hline \end{tabular} \end{table} </pre>	<table border="1"> <thead> <tr> <th>Length</th> <th>In Metre</th> </tr> </thead> <tbody> <tr> <td>1 Millimeter</td> <td>10^{-3}</td> </tr> <tr> <td>1 Centimeter</td> <td>10^{-2}</td> </tr> <tr> <td>1 Decimeter</td> <td>10^{-1}</td> </tr> <tr> <td>1 Decameter</td> <td>10</td> </tr> <tr> <td>1 Hectometer</td> <td>10^2</td> </tr> <tr> <td>1 Kilometer</td> <td>10^3</td> </tr> </tbody> </table>	Length	In Metre	1 Millimeter	10^{-3}	1 Centimeter	10^{-2}	1 Decimeter	10^{-1}	1 Decameter	10	1 Hectometer	10^2	1 Kilometer	10^3
Length	In Metre														
1 Millimeter	10^{-3}														
1 Centimeter	10^{-2}														
1 Decimeter	10^{-1}														
1 Decameter	10														
1 Hectometer	10^2														
1 Kilometer	10^3														

Applications of some of the above provisions in a `tabular` environment are shown in Table 7.5. The `>{\bfseries}` command before the option `l` in the first column prints all the entries of that column in boldface fonts (while `{\bf }` in the second column prints only the heading in boldface fonts). The `>{\$}|<{\$}` command converts the second column into math-mode, for which the mathematical entries of that column (all other than its heading) could be inserted directly without creating a separate math-mode for an entry. Two vertical lines, each of width 0.8 mm, on both sides of the table have been obtained by the `!{\vrule width 0.8mm}` commands. On the other hand, the `\setlength{\extrarowheight}{4mm}` command before the `tabular` environment creates an extra vertical space of 4 mm above each row of the table.

7.7 Merging Rows and Columns of Tables

When presenting different types of information in a table, some cells are often required to be merged into a single one. The `multirow` package provides the `\multicolumn{}{}{}` and `\multirow{}{}{}` commands for merging two or more columns and rows, respectively. The applications of the commands are shown in Table 7.6 on the facing page.

In `\multicolumn{n_c}{calign}{centry}`, n_c is the number of columns to be merged, `calign` is the alignment of the merged column, and `centry` is the entry of that merged cell. Since four columns in the first row in Table 7.6 are merged into a single cell, the number of entries in that row is reduced from six to three (the `\multicolumn{}{}{}` command spanning a single column can also be used for changing the alignment in that column). The permitted `calign` in the `tabular` environment is `l` (for left alignment), `r` (for right alignment), or `c` (for center alignment). Note that the option `X` as `calign` in `\multicolumn{}{}{}` may not work properly under the `tabularx`

Table 7.6 Merging two or more cells of a table into a single one

L^AT_EX input	<pre> \begin{tabular}{ l*{5}{c} } \hline \multirow{2}{*}{Name}& \multicolumn{4}{c}{Subjects}& \multirow{2}{*}{Total}\hline \cline{2-5} & Math & Phy & Chem & English & \hline \hline \hline Robin & 80 & 68 & 60 & 57 & 265\hline Julie & 72 & 62 & 66 & 63 & 263\hline Robert & 75 & 70 & 71 & 69 & 285\hline \hline \end{tabular} </pre>																												
Output	<table border="1"> <thead> <tr> <th rowspan="2">Name</th> <th colspan="4">Subjects</th> <th rowspan="2">Total</th> </tr> <tr> <th>Math</th> <th>Phy</th> <th>Chem</th> <th>English</th> </tr> </thead> <tbody> <tr> <td>Robin</td> <td>80</td> <td>68</td> <td>60</td> <td>57</td> <td>265</td> </tr> <tr> <td>Julie</td> <td>72</td> <td>62</td> <td>66</td> <td>63</td> <td>263</td> </tr> <tr> <td>Robert</td> <td>75</td> <td>70</td> <td>71</td> <td>69</td> <td>285</td> </tr> </tbody> </table>	Name	Subjects				Total	Math	Phy	Chem	English	Robin	80	68	60	57	265	Julie	72	62	66	63	263	Robert	75	70	71	69	285
Name	Subjects				Total																								
	Math	Phy	Chem	English																									
Robin	80	68	60	57	265																								
Julie	72	62	66	63	263																								
Robert	75	70	71	69	285																								

environment. Although many people suggest to use `>{\setlength{hsize}{n_c\hsize}}X` instead of simply `X`, it also may not work properly in some L^AT_EX compilers. Hence, a good option is to use `p{}` with manually adjusted argument value (the option `X` generates a column using `p{}` internally with automatically adjusted argument value).

Similarly, in `\multirow{n_r}{cwidth}{centry}`, `n_r` is the number of rows to be merged, `cwidth` is the width of the merged cell, and `centry` is the entry of that merged cell. The value of `cwidth` can be set manually (e.g., `25mm` or `1.0in`), or can be obtained an auto-adjusted one using an `*` only. The entry in the merged cell, obtained through `\multirow{}{}{}`, is vertically center aligned. Other alignment can be obtained by assigning an optional argument after the second mandatory argument of the command, e.g., `\multirow{4}{2cm}[3mm]{centry}` for merging four rows to produce a single cell of width 2 cm and to print `centry` in it 3 mm above the vertical center line (a negative value to the optional argument will print `centry` below the vertical center line). When some rows in a column are merged, `\multirow{}{}{}` is used in the first row to be merged and the column in each of the remaining merged rows is left blank (i.e., the column is ended simply by a `&` or `\`) as shown in the first and last columns in the second row of Table 7.6.

Further, the `\cline{m-n}` command is used in Table 7.6 for drawing a horizontal line covering columns `m` to `n` only. Another noticeable thing in Table 7.6 is the double horizontal line after the heading of the table. This is done using two consecutive `\hline` commands without any line break between them. Similarly, more than one vertical line can also be drawn using additional `|` symbol in the argument of the `\begin{tabular}{}{}` command.

Note that both the `\multirow{}{}{}` and `\multicolumn{}{}{}` commands can be used together for creating a single cell by merging a number of rows and columns. One such example is `\multicolumn{3}{c}{\multirow{2}{*}{Outcome}}`, where three columns and two rows are merged into a single cell for printing ‘Outcome’ with center alignment.

7.8 Table Wrapped by Texts*

If the size of a table is very small compared to the width of a page, the `wratable` environment, supported by the `wrapfig` package, can be used to wrap around the table by texts. The `wratable` environment needs two mandatory arguments, i.e., `\begin{wratable}{aside}{asize}`, where `aside` and `asize` are, respectively, the location and size of the table. The location can be specified by `l` (left side of the page) or `r` (right side of the page), while the size is specified in units (e.g., `25mm`, `1.0in`, or `0.3\linewidth`). The `wratable` environment is similar with the `table` environment; the only difference lies in creating the environment. A self-explanatory application of this environment is shown in Table 7.7.

Table 7.7 Table wrapped by texts through the `wratable` environment

LaTeX input	Output																
<pre> \begin{wratable}{r}{5cm} \centering \caption{Obtained marks.} \label{wrap-table} \begin{tabular}{ l c c c } \hline Name & Math & Phy & Chem\ \hline Robin & 80 & 68 & 60\ \hline Julie & 72 & 62 & 66\ \hline Robert & 75 & 70 & 71\ \hline \end{tabular} \end{wratable} Marks obtained by ... are shown in Table~\ref{wrap-table}, ... </pre>	<p>Marks obtained by Robin, Julie and Robert in Math, Phy and Chem are shown in Table 3, which is wrapped around by texts for saving space. However, the total marks obtained by the students could not shown in the table due to space limit. You can find it by hand calculation only.</p> <table border="1"> <caption>Table 3: Obtained marks.</caption> <thead> <tr> <th>Name</th> <th>Math</th> <th>Phy</th> <th>Chem</th> </tr> </thead> <tbody> <tr> <td>Robin</td> <td>80</td> <td>68</td> <td>60</td> </tr> <tr> <td>Julie</td> <td>72</td> <td>62</td> <td>66</td> </tr> <tr> <td>Robert</td> <td>75</td> <td>70</td> <td>71</td> </tr> </tbody> </table>	Name	Math	Phy	Chem	Robin	80	68	60	Julie	72	62	66	Robert	75	70	71
Name	Math	Phy	Chem														
Robin	80	68	60														
Julie	72	62	66														
Robert	75	70	71														

7.9 Table with Colored Background*

In order to make some entries of a table prominent, the `colortbl` package provides the `\rowcolor{}`, `\columncolor{}`, and `\cellcolor{}` commands for coloring, respectively, any row, column, and cell of a table by the color specified as the argument of a command (refer §2.4 on page 13 for detail of colors). For `gray` color, optional provision is there for specifying its intensity also, in which case the commands take the forms of `\rowcolor[gray]{x}`, `\columncolor[gray]{x}`, and `\cellcolor[gray]{x}`, where `x` is the intensity of the `gray` color to be specified by a number between 0 and 1. The row to be colored is to be started with a `\rowcolor{}` command, while a `\cellcolor{}` command is to be entered in the particular cell to be colored. On the other hand, a colored column is to be generated using a `\columncolor{}` command, in the form of `>\columncolor{}`, in the argument of the table-generating environment `tabular` or `tabularx`, e.g., `\begin{tabular}{|l|>\columncolor{green}c|r|}` command will generate the middle column center aligned and colored by green color.

As a major drawback, `\rowcolor{}`, `\columncolor{}`, and `\cellcolor{}` may override column-separating vertical lines and row-separating horizontal lines in some cases. There is no formal rule to preserve them. Column-separating vertical lines can be preserved by controlling the amounts of color panel overhang on either side of a column. This can be done through two optional arguments to `\rowcolor{}` and `\columncolor{}` in the forms of `\rowcolor{}[lhang][rhang]` and `\columncolor{}[lhang][rhang]`, where `lhang` and `rhang` are, respectively, the amounts of overhang on the left and right sides of a column. Without these two optional arguments, a color panel overhangs by default amount of `\tabcolsep`, while `lhang` equals `rhang` if only one is present. On the other hand, for preserving a row-separating horizontal line, `\rule{0pt}{rhtg}\noindent` may be used, where `rhtg` is the height of the zero-width rule generated by the `\rule{0pt}{}` command. Based on some trials, the values of `lhang`, `rhang`, and `rhtg` may be fixed manually, e.g., `\rowcolor{green}[0.9\tabcolsep]` or `\columncolor{blue}[0pt]`, or `\rule{0pt}{2.6ex}\noindent`.

Applications of the `\rowcolor{}`, `\columncolor{}`, and `\cellcolor{}` commands, as stated above, are shown in Table 7.8. Because of the repeated application of

Table 7.8 Table with colored background through the `\rowcolor{}`, `\columncolor{}`, and `\cellcolor{}` commands

LaTeX input	Output																				
<pre> \begin{table}[hbt] \begin{tabular}{ c l r r } \hline SN& Item& Price& Amount\ \hline \rowcolor{red}[0.9\tabcolsep] \rule{0pt}{2.9ex}\noindent 1& Rice& 35& 140\ \hline \rowcolor{gray}[0.7][0.91\tabcolsep] \rule{0pt}{2.7ex}\noindent 2& Sugar& 55& 55\ \hline \rowcolor{green}[0.91\tabcolsep] \rule{0pt}{2.7ex}\noindent 3& Atta& 75& 75\ \hline \multicolumn{3}{ r }{Total}& \cellcolor{blue} 270\ \hline \end{tabular} \end{table} </pre>	<table border="1"> <thead> <tr> <th>SN</th> <th>Item</th> <th>Price</th> <th>Amount</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Rice</td> <td>35</td> <td>140</td> </tr> <tr> <td>2</td> <td>Sugar</td> <td>55</td> <td>55</td> </tr> <tr> <td>3</td> <td>Atta</td> <td>75</td> <td>75</td> </tr> <tr> <td colspan="3">Total</td> <td>270</td> </tr> </tbody> </table>	SN	Item	Price	Amount	1	Rice	35	140	2	Sugar	55	55	3	Atta	75	75	Total			270
SN	Item	Price	Amount																		
1	Rice	35	140																		
2	Sugar	55	55																		
3	Atta	75	75																		
Total			270																		
<pre> \newcolumntype{B}[2] {>{\columncolor{#1}[0.91\tabcolsep]}#2} \begin{table}[hbt] \begin{tabular}{ c B{gray}{l} B{red}{r} B{green}{r} } \hline \rowcolor{white} \rule{0pt}{2.8ex}\noindent SN& Item& Price& Amount\ \hline 1& Rice& 35& 140\ \hline \rowcolor{white} \rule{0pt}{2.8ex}\noindent 2& Sugar& 55& 55\ \hline 3& Atta& 75& 75\ \hline \rowcolor{white} \multicolumn{3}{ r }{Total}& \cellcolor{blue} 270\ \hline \end{tabular} \end{table} </pre>	<table border="1"> <thead> <tr> <th>SN</th> <th>Item</th> <th>Price</th> <th>Amount</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Rice</td> <td>35</td> <td>140</td> </tr> <tr> <td>2</td> <td>Sugar</td> <td>55</td> <td>55</td> </tr> <tr> <td>3</td> <td>Atta</td> <td>75</td> <td>75</td> </tr> <tr> <td colspan="3">Total</td> <td>270</td> </tr> </tbody> </table>	SN	Item	Price	Amount	1	Rice	35	140	2	Sugar	55	55	3	Atta	75	75	Total			270
SN	Item	Price	Amount																		
1	Rice	35	140																		
2	Sugar	55	55																		
3	Atta	75	75																		
Total			270																		

`\columncolor{}` in the second example, a new column type with two arguments, `B{ccol}{calign}`, is defined through `\newcolumntype{B}[2]`, where `ccol` is the color argument of `\columncolor{}` and `calign` is the alignment of the column to be

generated. Note that any of `\rowcolor{}`, `\columncolor{}`, and `\cellcolor{}` overrides their earlier use in a table, which is shown in the second example in Table 7.8, where `\rowcolor{}` overrides `\columncolor{}` and `\cellcolor{}` overrides `\rowcolor{}`.

Also note that the `\rowcolor{}`, `\columncolor{}`, and `\cellcolor{}` commands may not work properly along with the column margin adjusting command `@{}` discussed in §7.5 on page 63 (in such a requirement, however, the `\tabcolsep` command may be redefined).