This lesson shows how you can build tables in LaTeX, influence the alignment of the cells, add rules to the table, and merge cells.

Tables in LaTeX are set using the `tabular` environment. This lesson will assume you load the `array` package, which adds more functionality to LaTeX tables, and which is not built into the LaTeX kernel only for historic reasons. So put the following in your preamble and we're good to go:

```
\usepackage{array}
```

In order to typeset a tabular we have to tell LaTeX how many columns will be needed and how they should be aligned. This is done in a mandatory argument – often referred to as the table preamble – to the `tabular` environment, in which you specify the columns by using **single-letter names**, called preamble-tokens. The available column types are:

- `l`: left aligned column
- `c`: centered column
- `r`: right aligned column
- `p{width}`: a column with fixed width width; the text will be automatically line wrapped and fully justified
- `m{width}`: like p, but vertically centered compared to the rest of the row
- `b{width}`: like p, but bottom aligned
- `w{align}{width}`: prints the contents with a fixed width, silently overprinting if things get larger. You can choose the horizontal alignment using l, c, or r.
- `W{align}{width}`: like w, but this will issue an overfull box warning if things get too wide.

In addition, a few other **preamble-tokens** are available which don't define a column but might be useful as well:

- `*{num}{string}`: repeats string for num times in the preamble. With this you can define multiple identical columns.
- `{decl}`: this will put decl before the contents of every cell in the following column (this is useful, e.g., to set a different font for this column)
- `<{decl}`: this will put decl after the contents of each cell in the previous column
- `|`: add a vertical rule
- `@{decl}`: replace the space between two columns with decl
- `!{decl}`: add decl in the center of the existing space

The columns l, c, and r will have the natural width of the widest cell. Each column has to be declared, so if you want three centered columns, you'd use ccc in the table preamble. Spaces are ignored, so c c c is the same.

In a table body columns are separated using an ampersand `&` and a new row is started using `\\`.

We have everything we need for our first table. In the following code the `&` and `\\` are aligned. This isn't necessary in LaTeX, but helps reading the source.

```
\begin{tabular}{lll}
  Animal & Food  & Size   \\
  dog    & meat  & medium \\
  horse  & hay   & large  \\
  frog   & flies & small  \\
\end{tabular}
```

If a table column contains a lot of text you will have issues to get that right with only l, c, and r. See what happens in the following example:

```
\begin{tabular}{cl}
  Animal & Description \\
  dog    & The dog is a member of the genus Canis, which forms part of the
           wolf-like canids, and is the most widely abundant terrestrial
           carnivore. \\
  cat    & The cat is a domestic species of small carnivorous mammal. It is the
           only domesticated species in the family Felidae and is often referred
           to as the domestic cat to distinguish it from the wild members of the
           family. \\
\end{tabular}
```

The issue is that the `l` type column typesets its contents **in a single row** at its natural width, even if there is a page border in the way. To overcome this you can use the `p` column. This typesets its contents as paragraphs with the width you specify as an argument and vertically aligns them at the top – which you'll want most of the time. Compare the above outcome to the following:

```
\begin{tabular}{cp{9cm}}
  Animal & Description \\
  dog    & The dog is a member of the genus Canis, which forms part of the
           wolf-like canids, and is the most widely abundant terrestrial
           carnivore. \\
  cat    & The cat is a domestic species of small carnivorous mammal. It is the
           only domesticated species in the family Felidae and is often referred
           to as the domestic cat to distinguish it from the wild members of the
           family. \\
\end{tabular}
```

If your table has many columns of the same type it is cumbersome to put that many column definitions in the preamble. You can make things easier by using `*{num}{string}`, which repeats the string num times. So `*{6}{c}` is equivalent to `cccccc`. To show you that it works here is the first table of this lesson with the newly learned syntax:

```
\begin{tabular}{*{3}{l}}
  Animal & Food  & Size   \\
  dog    & meat  & medium \\
  horse  & hay   & large  \\
  frog   & flies & small  \\
\end{tabular}
```

Animal	Food	Size
dog	meat	medium
horse	hay	large
frog	flies	small

1 Adding rules (lines)

A word of advice prior to introducing rules: lines should be used really sparsely in tables, and normally vertical ones look **unprofessional**. In fact, for professional tables you shouldn't use any of the standard lines; instead you should get familiar with the facilities of the `booktabs` package, which is why it is covered here first. For the sake of completeness the standard lines are shown in the more-info page.

`booktabs` provides four different types of lines. Each of those commands has to be used as the first thing in a row or following another rule. Three of the rule commands are: `\toprule`, `\midrule`, and `\bottomrule`. From their names the intended place of use should be clear:

```
\begin{tabular}{lll}
  \toprule
  Animal & Food  & Size   \\
  \midrule
  dog    & meat  & medium \\
  horse  & hay   & large  \\
  frog   & flies & small  \\
  \bottomrule
\end{tabular}
```

Animal	Food	Size
dog	meat	medium
horse	hay	large
frog	flies	small

The fourth rule command provided by booktabs is `\cmidrule`. It can be used to draw a rule that doesn't span the entire width of the table but only a specified column range. A column range is entered as a number span: `{number-number}`. Even if you only want to draw the rule for a single column you need to specify that as a range (with both numbers matching).

```
\begin{tabular}{lll}
  \toprule
  Animal & Food  & Size   \\
  \midrule
  dog    & meat  & medium \\
  \cmidrule{1-2}
  horse  & hay   & large  \\
  \cmidrule{1-1}
  \cmidrule{3-3}
  frog   & flies & small  \\
  \bottomrule
\end{tabular}
```

Animal	Food	Size
dog	meat	medium
horse	hay	large
frog	flies	small

Sometimes a rule would be too much of a separation for two rows but to get across the meaning more clearly you want to separate them by some means. In this case you can use `\addlinespace` to insert a small skip.

```
\begin{tabular}{cp{9cm}}
  \toprule
  Animal & Description \\
  \midrule
  dog    & The dog is a member of the genus Canis, which forms part of the
           wolf-like canids, and is the most widely abundant terrestrial
           carnivore. \\
  \addlinespace
  cat    & The cat is a domestic species of small carnivorous mammal. It is the
           only domesticated species in the family Felidae and is often referred
```

```
          to as the domestic cat to distinguish it from the wild members of the
          family. \\
  \bottomrule
\end{tabular}
```

Animal	Description
dog	The dog is a member of the genus Canis, which forms part of the wolf-like canids, and is the most widely abundant terrestrial carnivore.
cat	The cat is a domestic species of small carnivorous mammal. It is the only domesticated species in the family Felidae and is often referred to as the domestic cat to distinguish it from the wild members of the family.

2 Merging cells

In LaTeX you can merge cells horizontally by using the `\multicolumn` command. It has to be used as the first thing in a cell. `\multicolumn` takes three arguments:

- The number of cells which should be merged
- The alignment of the merged cell
- The contents of the merged cell

```
\begin{tabular}{lll}
  \toprule
  Animal & Food  & Size   \\
  \midrule
  dog    & meat  & medium \\
  horse  & hay   & large  \\
  frog   & flies & small  \\
  fuath  & \multicolumn{2}{c}{unknown} \\
  \bottomrule
\end{tabular}
```

Merging cells vertically isn't supported by LaTeX. Usually it suffices to leave cells empty to give the reader the correct idea of what was meant without explicitly making cells span rows.

```
\begin{tabular}{lll}
  \toprule
  Group     & Animal & Size   \\
  \midrule
  herbivore & horse  & large  \\
            & deer   & medium \\
            & rabbit & small  \\
  \addlinespace
  carnivore & dog    & medium \\
            & cat    & small  \\
            & lion   & large  \\
  \addlinespace
  omnivore  & crow   & small  \\
            & bear   & large  \\
            & pig    & medium \\
  \bottomrule
\end{tabular}
```

Group	Animal	Size
herbivore	horse	large
	deer	medium
	rabbit	small
carnivore	dog	medium
	cat	small
	lion	large
omnivore	crow	small
	bear	large
	pig	medium

3 Exercises

Use the simple table example to start experimenting with tables. Try out different alignments using the l, c and r column types. What happens if you have too few items in a table row? How about too many? Experiment with the `\multicolumn` command to span across columns.

Mon	Tue	Wed	Thu	Fri	Sat	Sun
Exercise	Work	Work	None	Work	Exercise	None
Exercise	Work		None	Work	Exercise	None
				Learn to play Music		

4 More on this topic

4.1 The other preamble contents

As the lesson didn't cover all the available preamble-tokens, a few others are explained with examples here. You might want to revisit the tables at the start of the lesson to get an overview of the things available. The short descriptions provided there should suffice to understand what the different column types m, b, w, and W do after you understood l, c, r, and p. If not you might want to experiment a bit with them. What's still missing are the handy other preamble-tokens $>$**,** $<$**, @, !, and** $|$.

4.2 Styling a column

Since $>$ and $<$ can be used to put things before and after the cell contents of a column, you can use these to add commands which affect the look of a column. For instance, if you want to italicize the first column and put a colon after it, you can do the following:

```
\begin{tabular}{>{\itshape}l<{:} *{2}{l}}
  \toprule
  Animal & Food  & Size   \\
  \midrule
  dog    & meat  & medium \\
  horse  & hay   & large  \\
  frog   & flies & small  \\
  \bottomrule
\end{tabular}
```

Animal:	Food	Size
dog:	meat	medium
horse:	hay	large
frog:	flies	small

You may want the first cell not to be affected because it is the table head. Here `\multicolumn` may be used. Remember that it can be used to change a single cell's alignment as shown below.

```
\begin{tabular}{>{\itshape}l<{:} *{2}{l}}
  \toprule
  \multicolumn{1}{l}{Animal} & Food  & Size   \\
  \midrule
  dog    & meat  & medium \\
  horse  & hay   & large  \\
  frog   & flies & small  \\
  \bottomrule
\end{tabular}
```

4.3 Manipulating the space between columns

Usually LaTeX pads each column by some space on both sides to give a balanced look and separate them. This space is defined with the length `\tabcolsep`. Due to the fact that each column is padded on both sides you get one `\tabcolsep` on either end of the table, and `2\tabcolsep` between two columns – one from each column. You can adjust this space to any length using `\setlength`:

```
\setlength\tabcolsep{1cm}

\begin{document}
\begin{tabular}{lll}
  Animal & Food  & Size   \\
  dog    & meat  & medium \\
  horse  & hay   & large  \\
  frog   & flies & small  \\
\end{tabular}
\end{document}
```

4.4 Vertical rules

Sometimes you have to use vertical rules.

```
\begin{tabular}{l|ll}
  Animal & Food  & Size   \\[2pt]
  dog    & meat  & medium \\
  horse  & hay   & large  \\
  frog   & flies & small  \\
\end{tabular}

\begin{tabular}{l!{|}ll}
  Animal & Food  & Size   \\[2pt]
  dog    & meat  & medium \\
  horse  & hay   & large  \\
  frog   & flies & small  \\
\end{tabular}
```

You might notice that the behavior of | is pretty similar to `!{decl}`; it adds the vertical rule between two columns leaving the padding as it is. There is a huge downside to this though; vertical rules don't work with the horizontal rules provided by `booktabs`. You can use the horizontal rules provided by LaTeX; those are `\hline` (corresponding to `\toprule`, `\midrule`, and `\bottomrule`) and `\cline` (which behaves like `\cmidrule`). As shown above, vertical rules will span any space specified in the optional argument to `\\`.

4.5 Customizing booktabs rules

All the `booktabs` rules and also `\addlinespace` support an optional argument in brackets with which you can specify the rule's thickness. In addition the trimming provided by `\cmidrule` can be customized by specifying a length in braces after r or l.

```
\begin{tabular}{@{} lll@{}} \toprule[2pt]
  Animal & Food  & Size   \\ \midrule[1pt]
  dog    & meat  & medium \\
  \cmidrule[0.5pt](r{1pt}l{1cm}){1-2}
  horse  & hay   & large  \\
  frog   & flies & small  \\ \bottomrule[2pt]
\end{tabular}
```

Animal	Food	Size
dog	meat	medium
horse	hay	large
frog	flies	small

4.6 Numeric alignment in columns

The alignment of numbers in tables can be handled by the column type S that is provided by the siunitx package.

A simple example with two aligned numeric columns would be:

```
\usepackage{siunitx}
\begin{document}
\begin{tabular}{SS}
\toprule
{Values} &  {More Values} \\
\midrule
1        &   2.3456 \\
1.2      &   34.2345 \\
-2.3     &   90.473 \\
40       &   5642.5 \\
5.3      &   1.2e3 \\
0.2      &    1e4 \\
\bottomrule
\end{tabular}
\end{document}
```

Note that any non-numeric cell must be "protected" by enclosing it in braces.

Values	More Values
1	2.3456
1.2	34.2345
-2.3	90.473
40	5642.5
5.3	1.2×10^3
0.2	1×10^4

4.7 Specifying the total table width

The width of a `tabular` environment is automatically determined based on the contents of the table. There are two commonly used mechanisms to specify a different total width.

Note that it is almost always preferable to format the table to a specified width as below (perhaps using a font size such as `\small` if necessary) rather than scaling a table with `\resizebox` and similar commands which will produce inconsistent font sizes and rule widths.

`tabular*` The `tabular*` environment takes an additional width argument that specifies the total width of the table. Stretchy space must be added to the table using the `\extracolsep` command. This space is added between all columns from that point in the preamble. It is almost always used with `\fill`, a special space that stretches to be as large as necessary.

```
\begin{center}
\begin{tabular}{cc}
\hline
A & B\\
C & D\\
\hline
\end{tabular}
\end{center}

\begin{center}
\begin{tabular*}{.5\textwidth}{@{\extracolsep{\fill}}cc@{}}
\hline
A & B\\
C & D\\
\hline
\end{tabular*}
\end{center}

\begin{center}
\begin{tabular*}{\textwidth}{@{\extracolsep{\fill}}cc@{}}
\hline
A & B\\
C & D\\
\hline
\end{tabular*}
\end{center}
```

A	B
C	D

A	B
C	D

A	B
C	D

`tabularx` The `tabularx` environment, provided by the package of the same name, has a similar syntax to `tabular*` but instead of adjusting the inter-column space, adjusts the widths of columns specified by a new column type, `X`. This is equivalent to a specification of `p{...}` for an automatically determined width.

```
\begin{center}
\begin{tabular}{lp{2cm}}
\hline
A & B B B B B B B B B B B B B B B B B B B B B B B B\\
C & D D D D D D D\\
\hline
\end{tabular}
\end{center}

\begin{center}
\begin{tabularx}{.5\textwidth}{lX}
\hline
A & B B B B B B B B B B B B B B B B B B B B B B B B\\
C & D D D D D D D\\
\hline
\end{tabularx}
\end{center}

\begin{center}
\begin{tabularx}{\textwidth}{lX}
\hline
A & B B B B B B B B B B B B B B B B B B B B B B B B\\
C & D D D D D D D\\
\hline
\end{tabularx}
\end{center}
```

A	B B B B B B
	B B B B B B
	B B B B B B
	B B B B B B
C	D D D D D
	D D

A	B B B B B B B B B B B B B B B B B B B B
	B B B B
C	D D D D D D D

A	B B B B B B B B B B B B B B B B B B B B B B B B
C	D D D D D D D

4.8 Multi-page tables

A `tabular` forms an unbreakable box so it must be small enough to fit on one page, and is often placed in a floating `table` environment.

Several packages provide variants with similar syntax that do allow page breaking. Here we show the `longtable` package:

```
\usepackage{longtable}
\begin{document}
\begin{longtable}{cc}
\multicolumn{2}{c}{A Long Table}\\
Left Side & Right Side\\
\hline
\endhead
\hline
\endfoot
aa & bb\\
Entry & b\\
a & b\\
a & b\\
a & b\\
a & b\\
a & bbb\\
a & b\\
a & b\\
a & b\\
a & b\\
a & b\\
a & b\\
a & b b b b b b\\
a & b b b b b\\
a & b b\\
A Wider Entry & b\\
\end{longtable}

\end{document}
```

A Long Table	
Left Side	Right Side
aa	bb
Entry	b
a	b
a	b
a	b
a	b
a	bbb
a	b
a	b
a	b
a	b
a	b
a	b
a	b b b b b b
a	b b b b b

A Long Table

Left Side	Right Side
a	b b
A Wider Entry	b

longtable is notable in that it preserves the column widths over all pages of the table; however in order to achieve this it may take several runs of LaTeX so that wide entries encountered later in the table can affect the column widths in earlier pages.

4.9 Table notes

It is quite common to need footnote-like marks in a table referring to notes under the table. The threeparttable package simplifies the markup for such tables, arranging that the notes are set in a block the same width as the table. Refer to the package documentation for full details, but we show a simple example here.

```
\begin{table}[h]
\begin{threeparttable}
   \caption{An Example}
   \begin{tabular}{ll}
   \toprule
    An entry & 42\tnote{1}\\
    Another entry & 24\tnote{2}\\
   \bottomrule
   \end{tabular}
   \begin{tablenotes}
   \item [1] the first note.
   \item [2] the second note.
   \end{tablenotes}
\end{threeparttable}
\end{table}
```

Table 2: An Example

An entry	42[1]
Another entry	24[2]

[1] the first note.
[2] the second note.

4.10 Line spacing in tables

In the main lesson we demonstrated \addlinespace from the booktabs package, which is useful for adding extra space between specific lines.

There are two general parameters that control line spacing, \arraystretch and \extrarowheight (the latter from the array package).

\renewcommand\arraystretch{1.5} will increase the baseline spacing by 50%.

Often, especially when using `\hline`, it is better just to increase the height (**above**) of rows, without increasing their depth below the baseline. The following example demonstrates the `\extrarowheight` parameter.

```
\begin{center}
\begin{tabular}{cc}
\hline
Square& $x^2$\\
\hline
Cube& $x^3$\\
\hline
\end{tabular}
\end{center}


\begin{center}
\setlength\extrarowheight{2pt}
\begin{tabular}{cc}
\hline
Square& $x^2$\\
\hline
Cube& $x^3$\\
\hline
\end{tabular}
\end{center}
```

Square	x^2
Cube	x^3

Square	x^2
Cube	x^3