

# tabularray and kableExtra

## Flexible L<sup>A</sup>T<sub>E</sub>X tables

### Table of contents

<b>Why use tabularray?</b>	<b>2</b>
<b>Text styles</b>	<b>4</b>
<b>Colors</b>	<b>4</b>
<b>Precedence: Rows, Columns, Cells</b>	<b>6</b>
<b>Lines: Horizontal and Vertical</b>	<b>6</b>
<b>Width</b>	<b>7</b>
<b>Cell-specific settings</b>	<b>8</b>
<b>Extra tabularray arguments</b>	<b>9</b>
<b>Solved bugs</b>	<b>9</b>
Issue #616 . . . . .	10
Issue #634 . . . . .	11
Issue #636 . . . . .	11
Issue #645 . . . . .	12
Issue #660 . . . . .	12
Issue #701 . . . . .	13
Issue #738 . . . . .	14
<b>Remaining bugs</b>	<b>15</b>
Issue #366 . . . . .	15
Issue #571 . . . . .	17

This vignette requires this `kableExtra` development branch:

```
remotes::install_github("vincentarelbundock/kableExtra@tabularray")
```

This vignette demonstrates how to harness the capabilities of `tabularray` and `kableExtra` for crafting advanced  $\text{\LaTeX}$  tables in R. Readers will learn to use the `kbl()` function for transforming data frames into sophisticated  $\text{\LaTeX}$  tables. The integration of `kableExtra` with `tabularray` brings a host of benefits, including versatile cell alignment and the ability to create colorful, multirow, and multicolumn tables. The guide covers controlling rows, columns, cells, line manipulation, and table width management, aiming to equip users with the knowledge to create responsive, visually appealing, and well-structured  $\text{\LaTeX}$  tables.

[Click here to visit the `tabularray` website and read its documentation.](#)

To create tables with `tabularray` in `kableExtra`, users only need to set the `tabular` argument of the `kbl` function:

```
library(kableExtra)

df <- data.frame(Car = row.names(mtcars), mtcars[, 1:3])[1:4,]
row.names(df) <- NULL

kbl(df, tabular = "tblr")
```

Car	mpg	cyl	disp
Mazda RX4	21.0	6	160
Mazda RX4 Wag	21.0	6	160
Datsun 710	22.8	4	108
Hornet 4 Drive	21.4	6	258

## Why use `tabularray`?

`kableExtra` supports the `tabularray` package for  $\text{\LaTeX}$  to create tables with advanced formatting options. `tabularray` offers several benefits:

- Versatile cell alignment
- Multirow and multicolumn support
- Flexible column types
- Advanced line customization
- Colorful table options
- Improved vertical spacing

- Compatibility with L<sup>A</sup>T<sub>E</sub>X3
- Support for long tables
- Integration with popular L<sup>A</sup>T<sub>E</sub>X libraries

One of the most important benefits of using `tabulararray` is that code generated by `kableExtra` is very readable, and much easier to edit manually than when using other L<sup>A</sup>T<sub>E</sub>X packages. For example, here is the code for a table with colored and bolded rows:

```
kbl(df, tabular = "tblr") |>
  row_spec(2:3, bold = TRUE, background = "pink") |>
  cat()
```

```
\begin{tblr}{
  colspec={Q[halign=l]Q[halign=r]Q[halign=r]Q[halign=r]},
  rowspec={Q[]Q[]Q[bg=pink, font=\bfseries]Q[bg=pink, font=\bfseries]Q[]Q[]},
}
\hline
Car & mpg & cyl & disp\\
\hline
Mazda RX4 & 21.0 & 6 & 160\\
Mazda RX4 Wag & 21.0 & 6 & 160\\
Datsun 710 & 22.8 & 4 & 108\\
Hornet 4 Drive & 21.4 & 6 & 258\\
\hline
\end{tblr}
```

Here are some of the important things to notice:

- The tabular environment starts with `\begin{tblr}` and ends with `\end{tblr}`.
- The data and style are completely independent: The tabular content is untouched, and we add `colspec` and `rowspec` headers to specify the style.
- Each row and each column gets a `Q[]` entry, which acts as a styling operator. We can insert arguments in the square brackets to change the style of all the cells in a column or row. The available arguments [are described in the `tabulararray` documentation](#).

In contrast, when using other L<sup>A</sup>T<sub>E</sub>X packages, each entry of the table must be modified, which arguably makes the code messier and harder to edit:

```
kbl(df, format = "latex") |>
  row_spec(2:3, bold = TRUE, background = "pink") |>
  cat()
```

```

\begin{tabular}[t]{l|r|r|r}
\hline
Car & mpg & cyl & disp\\
\hline
Mazda RX4 & 21.0 & 6 & 160\\
\hline
\cellcolor{pink}{\textbf{Mazda RX4 Wag}} & \cellcolor{pink}{\textbf{21.0}} & \cellcolor{pink}{\textbf{6}} & \cellcolor{pink}{\textbf{160}}\\
\hline
\cellcolor{pink}{\textbf{Datsun 710}} & \cellcolor{pink}{\textbf{22.8}} & \cellcolor{pink}{\textbf{4}} & \cellcolor{pink}{\textbf{108}}\\
\hline
Hornet 4 Drive & 21.4 & 6 & 258\\
\hline
\end{tabular}

```

## Text styles

The core functions to modify text styles in `kableExtra` are `cell_spec()`, `row_spec()`, and `column_spec()`. All the arguments of these functions are supported, except for 3 arguments of the `row_spec()` function: `angle`, `font_size`, and `align`. Users can achieve a similar effect using the `cell_spec()` function (see the section below on cell-specific settings).

Here is an example of a table with bold text and strikeouts:

```

kbl(df, tabular = "tblr") |>
  row_spec(2:3, bold = TRUE) |>
  column_spec(1, strikeout = TRUE)

```

Car	mpg	cyl	disp
<del>Mazda RX4</del>	21.0	6	160
<del><b>Mazda RX4 Wag</b></del>	<b>21.0</b>	<b>6</b>	<b>160</b>
<del><b>Datsun 710</b></del>	<b>22.8</b>	<b>4</b>	<b>108</b>
<del>Hornet 4 Drive</del>	21.4	6	258

## Colors

In `tabularray`, color names are supported through the integration of the `xcolor` and `ninecolors` packages. The `xcolor` package is a comprehensive solution in LaTeX for color customization, offering a wide range of predefined color names and the ability to define custom colors using various color models like RGB, CMYK, and HTML. This flexibility allows

for precise color specification and is ideal for setting text, table elements, and other document components in LaTeX.

The basic L<sup>A</sup>T<sub>E</sub>X colors are: black, blue, brown, cyan, darkgray, gray, green, lightgray, lime, magenta, olive, orange, pink, purple, red, teal, violet, white, yellow.

The `ninecolors` package adds suffix to 9 of those colors to set 1 of 13 different hues: gray, red, brown, yellow, olive, green, teal, cyan, azure, blue, violet, magenta, purple. For all colors, 0 means black, and 10 is white. Two colors with the same numbered suffix have the same luminance (ex: `gray3` and `olive3`). These nine colors are carefully selected to ensure proper color contrast according to the Web Content Accessibility Guidelines (WCAG). This feature is particularly useful for creating documents with high readability and accessibility standards. To get proper WCAG Color Contrast, the `ninecolors` author recommends choosing two colors with different names, with at least a 5 unit difference in level.

The `col_spec()`, `row_spec()`, and `cell_spec()` functions support these colors out of the box:

```
df |>
  kbl(tabular = "tblr") |>
  column_spec(1, color = "purple3") |>
  row_spec(2:3, background = "azure9")
```

Car	mpg	cyl	disp
Mazda RX4	21.0	6	160
Mazda RX4 Wag	21.0	6	160
Datsun 710	22.8	4	108
Hornet 4 Drive	21.4	6	258

Of course, `kableExtra` themes are also supported:

```
df |>
  kbl(tabular = "tblr") |>
  kable_styling(latex_options = "striped")
```

Car	mpg	cyl	disp
Mazda RX4	21.0	6	160
Mazda RX4 Wag	21.0	6	160
Datsun 710	22.8	4	108
Hornet 4 Drive	21.4	6	258

## Precedence: Rows, Columns, Cells

In some contexts, user-specified settings like cell colors or text styles enter in conflict. In such cases, these two rules are applied:

1. `cell_spec()` settings have the highest precedence.
2. The last function call determines if row or column settings have precedence: `column_spec()` vs `row_spec()`

In the following example, we define conflicting background colors at the cell, column, and row level. In that case, the cell wins. When there is only a conflict between rows and columns, rows win because `row_spec()` is called after `column_spec()`.

```
df2 <- df
df2[1, 1] <- cell_spec(df2[1, 1], background = "yellow", format = "tblr")
kbl(df2, tabular = "tblr", escape = FALSE) |>
  column_spec(1:2, background = "pink") |>
  row_spec(1:2, background = "azure8")
```

Car	mpg	cyl	disp
Mazda RX4	21.0	6	160
Mazda RX4 Wag	21.0	6	160
Datsun 710	22.8	4	108
Hornet 4 Drive	21.4	6	258

## Lines: Horizontal and Vertical

The `tabulararray` package offers advanced table formatting options in  $\text{\LaTeX}$ , especially for vertical and horizontal lines. The syntax to control these lines is slightly different than for other output format in `kableExtra`, so it deserves some explanation. Consider this example:

```
kbl(df,
  tabular = "tblr",
  toprule = "", midrule = "", bottomrule = "",
  linesep = "hlines={dash=dotted, fg=brown6}",
  vline = "vlines={dash=dashed, fg=green4, wd=2pt}"
)
```

Car	mpg	cyl	disp
Mazda RX4	21.0	6	160
Mazda RX4 Wag	21.0	6	160
Datsun 710	22.8	4	108
Hornet 4 Drive	21.4	6	258

To achieve this result, we began by removing the default horizontal rules (`toprule`, `midrule`, `bottomrule`), otherwise, there would be double horizontal lines in the table. Then, we use the `linesep` argument to tell `tabulararray` how to format horizontal lines in the table, and the `vline` argument to control vertical lines.

When `vlines` and `hlines` are in plural form, they control all lines in the table. `tabulararray` also supports another syntax for specifying individual lines. For example, we can control the color, width, and type of lines as follows::

```
kbl(df,
    tabular = "tblr",
    toprule = "", midrule = "", bottomrule = "",
    linesep = "hline{1-6}={dash=solid, fg=brown6}",
    vline = "vline{2,3}={dash=dotted, fg=green4}"
)
```

Car	mpg	cyl	disp
Mazda RX4	21.0	6	160
Mazda RX4 Wag	21.0	6	160
Datsun 710	22.8	4	108
Hornet 4 Drive	21.4	6	258

## Width

To fix the width of certain columns or expand the table to full page width, `kableExtra` can use `X` or `Q` columns in `tabulararray`. `X` columns are designed for scenarios where you want the table columns to automatically adjust their widths to fill the entire available space. This feature is particularly useful for creating tables that span the full width of a page or container. When you use `X` columns, the width of each column is proportionally divided based on the available space, allowing for a responsive and evenly distributed layout. On the other hand, `Q` columns function like the standard `p` columns in  $\text{\LaTeX}$ , where you manually specify the width of each column. The content in these columns is wrapped to fit within the set width. This

type of column is ideal when precise control over column width is needed, such as in tables with varying content lengths or specific design requirements.

When using the `kable_styling()` function in R for  $\text{\LaTeX}$  output and setting `full_width = TRUE`, the table automatically employs `X` columns. This configuration makes the table expand to fill the width of its container, with each column adjusting its width to fit proportionally. This is a key feature for creating full-width tables that need to be responsive and aesthetically balanced in their layout.

This example illustrates how to customize the width of a single columns, while expanding the rest of the table to fill the entire page width:

```
kbl(df,
    tabular = "tblr",
    align = "lccc") |>
kable_styling(full_width = TRUE) |>
column_spec(1, width = "8cm")
```

Car	mpg	cyl	disp
Mazda RX4	21.0	6	160
Mazda RX4 Wag	21.0	6	160
Datsun 710	22.8	4	108
Hornet 4 Drive	21.4	6	258

## Cell-specific settings

To apply cell-specific settings, we can use the `cell_spec()` function. In `kbl()`, it is necessary to use `escape=TRUE` and `tabular="tblr"`. In `cell_spec()`, it is necessary to use `tabular="tblr"`:

```
df2 <- df
df2[2, 2] <- cell_spec(
  df2[2, 2],
  background = "pink",
  align = "c",
  color = "azure2",
  font_size = 20,
  angle = 45,
  format = "tblr")
```



```
kbl(df2, tabular = "tblr", escape = FALSE)
```

Car	mpg	cyl	disp
Mazda RX4	21	6	160
Mazda RX4 Wag	21	6	160
Datsun 710	22.8	4	108
Hornet 4 Drive	21.4	6	258

## Extra tabularray arguments

The [tabularray document](#) describes many other arguments which are available to control cells, rows, and columns of a table. For example, the `ht` argument can control the height of a row.

`kableExtra` does not have a `height` argument, but it is possible (and fun) to push through extra (unspecified) `tabularray` arguments through `background` argument. To understand how this works, note that when we call `background="red"`, `kableExtra` simply adds a string to the `rowspec` header: `fg=red`. If we extend the `background` entry, then we can specify height and vertical alignment too. Here, we use `f` for “foot” (also: `h`, `m`) and specify the height in `em` units (also: `cm`, `pt`, `in`):

```
kbl(df, tabular = "tblr") |>
  row_spec(2, background = "pink, ht=3.5em, valign=f")
```

Car	mpg	cyl	disp
Mazda RX4	21.0	6	160
Mazda RX4 Wag	21.0	6	160
Datsun 710	22.8	4	108
Hornet 4 Drive	21.4	6	258

## Solved bugs

This section prints several tables which previous included bugs, as reported by `kableExtra` users on the issue tracker. This illustrates that `tabularray` solves many longstanding prob-

lems.

## Issue #616

```
mtcars %>%  
  head(n = 10) %>%  
  kbl(tabular = "tblr") %>%  
  kable_styling() |>  
  row_spec(seq(1, 10, by = 2), background = "gray8, ht=1cm") |>  
  row_spec(seq(2, 10, by = 2), background = "white, ht=1cm")
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1
Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4
Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2
Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2
Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4

## Issue #634

```
cs_dt <- mtcars[1:10, 1:2]
cs_dt$mpg = cell_spec(cs_dt$mpg, align = "c")
kbl(cs_dt, tabular = "tblr", escape = FALSE) %>%
  kable_styling(latex_options = "striped")
```

	mpg	cyl
Mazda RX4	21	6
Mazda RX4 Wag	21	6
Datsun 710	22.8	4
Hornet 4 Drive	21.4	6
Hornet Sportabout	18.7	8
Valiant	18.1	6
Duster 360	14.3	8
Merc 240D	24.4	4
Merc 230	22.8	4
Merc 280	19.2	6

## Issue #636

```
set.seed(1024)
paint <- function(x) {
  col <- ifelse(x < 0.5, "yellow", "red")
  sapply(seq_along(x), function(i) cell_spec(
    formatC(x[i], format = "f", digits = 2), background = col[i], format = "tblr")
  )
}
DF <- data.frame(V1 = sample(letters,10,T), V2 = abs(rnorm(10)), V3 = abs(rnorm(10)))
DF[,-1] = lapply(DF[,-1], paint)

kbl(DF, tabular = "tblr", digits = 2, escape = FALSE) |>
  kable_styling(latex_options = "striped")
```

V1	V2	V3
u	1.74	0.38
v	0.11	0.84
u	0.91	0.58
m	1.06	1.45
o	0.02	0.28
x	0.29	1.12
m	0.03	1.32
p	0.73	1.41
g	0.53	0.94
h	0.20	2.28

### Issue #645

```
mtcars[1:3, 1:3] |>
  kbl(tabular = "tblr", booktabs = TRUE) |>
  row_spec(0, background = "teal3", bold = TRUE, color = "white")
```

	mpg	cyl	disp
Mazda RX4	21.0	6	160
Mazda RX4 Wag	21.0	6	160
Datsun 710	22.8	4	108

### Issue #660

```
kbl(mtcars[1:7, ], align = "c", tabular = "tblr", booktabs = TRUE, linesep = "") %>%
  kable_styling(latex_options = c("striped", "HOLD_position"), full_width = TRUE) %>%
  column_spec(1, width = "4cm")
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225	105	2.76	3.460	20.22	1	0	3	1
Duster 360	14.3	8	360	245	3.21	3.570	15.84	0	0	3	4

## Issue #701

```
dt <- data.frame(
  title = c(rep("a", 3), rep("b", 3), rep("c", 3), rep("d", 3)),
  value = 1:12
)
kbl(dt, format = "latex", tabular = "tblr") |>
  collapse_rows(1, latex_hline = "none") |>
  pack_rows(index = c('a' = 3, 'b' = 3, 'c' = 3, 'd' = 3)) |>
  row_spec(c(1, 5, 9, 13), background = "gray8")
```

title	value
<b>aa</b>	
	1
a	2
a	3
<b>bb</b>	
	4
b	5
b	6
<b>cc</b>	
	7
c	8
c	9
<b>dd</b>	
	10
d	11
d	12

## Issue #738

```
t <- head(cars)
t$dist <- cell_spec(t$dist, background = "red", format = "tblr")
kbl(t, tabular = "tblr", escape = FALSE) |> kable_styling(latex_options = c("striped"))
```

speed	dist
4	2
4	10
7	4
7	22
8	16
9	10

## Remaining bugs

The bugs in this section should be easy to fix now.

### Issue #366

```
groups <- c(3, 3, 5, 4, 3, 3, 3, 2, 3, 2, 3)
groups <- setNames(groups, c("a", "b", "c", "d", "e", "f", "g", "h", "i", "j", "k"))
x <- data.frame(Term = c('Other', 'White', 'Black', '18 to 24', '25 to 39', '40 to 59', 'M
  uni = c('Ref', '0.87(0.36, 2.1)', '1.3(0.62, 2.7)~', 'Ref', '0.65(0.32, 1.3)',
  adj = c('Ref', '0.63(0.21, 1.9)', '0.86(0.40, 1.8)', 'Ref', '1.5(0.43, 5.1)~',
  uni1 = c('Ref', '1.1(0.37, 3.2)~', '0.67(0.19, 2.3)', 'Ref', '0.38(0.16, 0.89)*
  adj1 = c('Ref', '0.73(0.22, 2.5)', '0.47(0.14, 1.6)', 'Ref', '0.88(0.15, 5.2)',

kbl(x, tabular = "tblr", row.names = FALSE) |>
  kable_styling(latex_options = c("striped", "scale_down")) |>
  group_rows(index = groups)
```

Term	uni	adj	uni1	adj1
<b>a</b>				
Other	Ref	Ref	Ref	Ref
White	0.87(0.36, 2.1)	0.63(0.21, 1.9)	1.1(0.37, 3.2) <sup>^</sup>	0.73(0.22, 2.5)
Black	1.3(0.62, 2.7) <sup>^</sup>	0.86(0.40, 1.8)	0.67(0.19, 2.3)	0.47(0.14, 1.6)
<b>b</b>				
18 to 24	Ref	Ref	Ref	Ref
25 to 39	0.65(0.32, 1.3)	1.5(0.43, 5.1) <sup>^</sup>	0.38(0.16, 0.89)*	0.88(0.15, 5.2)
40 to 59	0.69(0.38, 1.2)	2.1(0.75, 5.9) <sup>^</sup>	0.61(0.24, 1.5)	1.9(0.48, 7.6) <sup>^</sup>
<b>c</b>				
Married	Ref	Ref	Ref	Ref
No longer married	4.5(1.5, 13.5)** <sup>^</sup>	2.5(0.73, 8.5) <sup>^</sup>	3.5(1.0, 12.5) <sup>^</sup>	1.7(0.48, 6.0) <sup>^</sup>
Never married	3.8(1.6, 9.1)** <sup>^</sup>	2.8(0.79, 9.6) <sup>^</sup>	2.4(0.84, 7.0) <sup>^</sup>	1.8(0.39, 8.0) <sup>^</sup>
Living with partner	2.7(0.86, 8.3) <sup>^</sup>	2.0(0.62, 6.4) <sup>^</sup>	1.7(0.34, 8.3) <sup>^</sup>	1.2(0.28, 5.5) <sup>^</sup>
Missing	7.1(2.4, 20.7)***	5.9(1.0, 34.4) <sup>^</sup>	8.0(2.5, 25.4)***	6.5(0.95, 44.8) <sup>^</sup>
<b>d</b>				
0-1	Ref	Ref	Ref	Ref
2-5	1.9(0.47, 7.3) <sup>^</sup>	0.99(0.14, 7.1)	1.2(0.23, 6.5) <sup>^</sup>	0.36(0.06, 2.2)
6-10	2.5(0.89, 6.9) <sup>^</sup>	0.70(0.16, 3.1)	2.5(0.81, 7.8) <sup>^</sup>	0.41(0.09, 1.8)
11+	4.1(1.5, 11.6)** <sup>^</sup>	0.57(0.11, 3.0)	4.1(1.3, 13.3)* <sup>^</sup>	0.36(0.08, 1.7)
<b>e</b>				
0-1	Ref	Ref	Ref	Ref
2-5	4.2(2.2, 7.8)***	3.3(0.87, 12.8) <sup>^</sup>	4.4(2.1, 9.3)***	3.2(0.45, 23.2) <sup>^</sup>
6+	14.1(2.6, 75.9)** <sup>^</sup>	4.2(0.85, 20.3) <sup>^</sup>	18.2(2.8, 120.2)** <sup>^</sup>	4.6(0.55, 38.9) <sup>^</sup>
<b>f</b>				
0-1	Ref	Ref	Ref	Ref
2-5	1.6(0.81, 3.1) <sup>^</sup>	1.3(0.44, 3.8) <sup>^</sup>	2.9(0.96, 8.7) <sup>^</sup>	3.0(0.75, 12.0) <sup>^</sup>
6+	3.7(1.5, 9.2)** <sup>^</sup>	2.1(0.47, 9.6) <sup>^</sup>	6.0(1.9, 18.6)** <sup>^</sup>	3.1(0.54, 18.1) <sup>^</sup>
<b>g</b>				
0-1	Ref	Ref	Ref	Ref
2-5	2.8(1.2, 6.3)* <sup>^</sup>	0.65(0.13, 3.2)	3.8(1.4, 10.4)* <sup>^</sup>	0.95(0.11, 8.0)
6+	16.8(3.1, 92.5)** <sup>^</sup>	2.6(0.54, 12.5) <sup>^</sup>	22.5(3.2, 155.8)** <sup>^</sup>	3.3(0.41, 26.8) <sup>^</sup>
<b>h</b>				
Yes	Ref	NA	Ref	NA
No	0.59(0.28, 1.2)	NA	0.99(0.33, 3.0)	NA



}

## Issue #571

Notes:

- Striping does not take groups into account, but at least the indentation works.

```
kbl(mtcars[1:10, 1:6], tabular = "tblr") |>
  pack_rows("Group 1", 4, 7) |>
  pack_rows("Group 2", 8, 10) |>
  kable_styling(latex_options = c("striped"))
```

	mpg	cyl	disp	hp	drat	wt
Mazda RX4	21.0	6	160.0	110	3.90	2.620
Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875
Datsun 710	22.8	4	108.0	93	3.85	2.320
<b>Group 1</b>						
Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215
Hornet Sportabout	18.7	8	360.0	175	3.15	3.440
Valiant	18.1	6	225.0	105	2.76	3.460
Duster 360	14.3	8	360.0	245	3.21	3.570
<b>Group 2</b>						
Merc 240D	24.4	4	146.7	62	3.69	3.190
Merc 230	22.8	4	140.8	95	3.92	3.150
Merc 280	19.2	6	167.6	123	3.92	3.440