

## 2.5: Accessing a Data Frame

We access the individual elements in a data frame using square brackets to identify a specific cell. For instance, the following accesses the data in the cell in row 15, column 12:

```
> int92.dat[15,12]
[1] 180
```

We can also access cells by name by putting quotes around the name:

```
> int92.dat["71","perf"]
[1] 105.1
```

This expression returns the data in the row labeled `71` and the column labeled `perf`. Note that this is not row 71, but rather the row that contains the data for the processor whose name is `71`.

We can access an entire column by leaving the first parameter in the square brackets empty. For instance, the following prints the value in every row for the column labeled `clock`:

```
> int92.dat[, "clock"]
[1] 100 125 166 175 190 ...
```

Similarly, this expression prints the values in all of the columns for row 36:

```
> int92.dat[36,]
nperf perf clock threads cores ...
36 13.07378 79.86399 80 1 1 ...
```

The functions `nrow()` and `ncol()` return the number of rows and columns, respectively, in the data frame:

```
> nrow(int92.dat)
[1] 78
> ncol(int92.dat)
[1] 16
```

Because R functions can typically operate on a vector of any length, we can use built-in functions to quickly compute some useful results. For example, the following expressions compute the minimum, maximum, mean, and standard deviation of the `perf` column in the `int92.dat` data frame:

```
> min(int92.dat[, "perf"])
[1] 36.7
> max(int92.dat[, "perf"])
[1] 366.857
> mean(int92.dat[, "perf"])
[1] 124.2859
> sd(int92.dat[, "perf"])
[1] 78.0974
```

This square-bracket notation can become cumbersome when you do a substantial amount of interactive computation within the R environment. R provides an alternative notation using the `$` symbol to more easily access a column. Repeating the previous example using this notation:

```
> min(int92.dat$perf)
[1] 36.7
> max(int92.dat$perf)
[1] 366.857
> mean(int92.dat$perf)
[1] 124.2859
> sd(int92.dat$perf)
[1] 78.0974
```

This notation says to use the data in the column named `perf` from the data frame named `int92.dat`. We can make yet a further simplification using the `attach` function. This function makes the corresponding data frame local to the current workspace, thereby eliminating the need to use the potentially awkward `$` or square-bracket indexing notation. The following example shows how this works:

```
> attach(int92.dat)
> min(perf)
[1] 36.7
> max(perf)
[1] 366.857
> mean(perf)
[1] 124.2859
> sd(perf)
[1] 78.0974
```

To change to a different data frame within your local workspace, you must first detach the current data frame:

```
> detach(int92.dat)
> attach(fp00.dat)
> min(perf)
[1] 87.54153
> max(perf)
[1] 3369
> mean(perf)
[1] 1217.282
> sd(perf)
[1] 787.4139
```

Now that we have the necessary data available in the R environment, and some understanding of how to access and manipulate this data, we are ready to generate our first regression model.

This page titled [2.5: Accessing a Data Frame](#) is shared under a [CC BY-NC 4.0](#) license and was authored, remixed, and/or curated by [David Lilja \(University of Minnesota Libraries Publishing\)](#) via [source content](#) that was edited to the style and standards of the LibreTexts platform.