# MODULE 5

# FILTERING DATA IN R

**Contents**



In the Module 4, you learned how to retrieve data from the class webpage, enter your own data into a CSV file, load that data into R, and how to view that data in R. In this module, we will learn how to create subsets (i.e., filter) a data.frame into smaller data.frames. For example, you may want to create a data.frame that contains just male bears from a data.frame with both male and female bears, or a data.frame that only contains sales during summer months from a data.frame that contains all sales. Less often you may wish to eliminate a particular individual from the data.frame, perhaps if it is considered to be erroneous.

## 5.1 Filtering a data.frame

It is common to create a new data.frame that contains only some of the individuals from an existing data.frame. The process of creating the newer, smaller data.frame is called filtering (or subsetting) and is accomplished with `filterD()`. The `filterD()` function requires the original data.frame as the first argument and a condition statement as the second argument. The condition statement is used to either include or exclude individuals from the original data.frame. Condition statements consist of the name of a variable in the original data.frame, a comparison operator, and a comparison value (Table 5.1). The results from `filterD()` should be assigned to an object, which is then the name of the new data.frame.

The following are examples of new data.frames created from *bears* (which was created in the previous module). The name of the new data.frame (i.e., object left of the assignment operator) can be any valid object name. As demonstrated below, the new data.frame (or its structure) should be examined after each filtering to ensure that the data.frame actually contains the items that you desire.

Table 5.1. Condition operators used in `filterD()` and their results. Note that *var* generically represents a variable in the original data.frame and *value* is a generic value or level. Both *variable* and *value* would be replaced with specific items (see examples in main text).

| Condition Operator | Individuals Returned from Original Data Frame |
|---|---|
| *var* == *value* | all individual that are **equal** to the given value |
| *var* != *value* | all individuals that are **NOT equal** to the given value |
| *var* > *value* | all individuals that are **greater than** the given value |
| *var* >= *value* | all individuals that are **greater than or equal** to the given value |
| *var* < *value* | all individuals that are **less than** the given value |
| *var* <= *value* | all individuals that are **less than or equal** to the given value |
| *condition* , *condition* | all individuals that **meet both conditions** |
| *condition* \| *condition* | all individuals that **meet one or both conditions**[1] |

- Only individuals from *Bayfield* county.

```
> bf <- filterD(bears,loc=="Bayfield")
> bf
  length.cm weight.kg      loc
1     139.0       110 Bayfield
2     138.0        60 Bayfield
3     139.0        90 Bayfield
4     120.5        60 Bayfield
5     149.0        85 Bayfield
```

- Individuals from both *Bayfield* and *Ashland* counties.

```
> bfash <- filterD(bears,loc %in% c("Bayfield","Ashland"))
> bfash
  length.cm weight.kg      loc
1     139.0       110 Bayfield
2     138.0        60 Bayfield
3     139.0        90 Bayfield
4     120.5        60 Bayfield
5     149.0        85 Bayfield
6     141.0       100  Ashland
7     141.0        95  Ashland
```

- Individuals with a weight greater than 100 kg.

```
> gt100 <- filterD(bears,weight.kg>100)
> gt100
  length.cm weight.kg      loc
1     139.0       110 Bayfield
2     166.0       155  Douglas
3     151.5       140  Douglas
4     129.5       105  Douglas
5     150.0       110  Douglas
```

- Individuals from *Douglas* County that weighed at least 150 kg.

```
> do150 <- filterD(bears,loc=="Douglas",weight.kg>=150)
> do150
  length.cm weight.kg     loc
1       166       155 Douglas
```

## 5.2 Selecting Individuals

In some instances, you may need to select or exclude an individual from a data.frame. Positions within an object are identified within square brackets. As data.frames are two-dimensional objects they are indexed by a row and a column, in that order. For example, the item in the third row and second column of `bears` is selected below.

```
> bears[3,2]
[1] 90
```

An entire row or column may be selected by omitting the other dimension. For example, one could select the entire second column with `bears[,2]`, but this is also the `weight.kg` variable and is better selected with `bears$weight.kg`. As a better example, the entire third row is selected below (note that the column designation was omitted).

```
> bears[3,]
  length.cm weight.kg      loc
3       139        90 Bayfield
```

Multiple rows are selected by combining row indices together with `c()`. For example, the third, fifth, and eighth rows are selected below (again, the column index is omitted).

```
> bears[c(3,5,8),]
  length.cm weight.kg      loc
3       139        90 Bayfield
5       149        85 Bayfield
8       150        85  Douglas
```

Finally, rows can be excluded by preceding the row indices with a negative sign.

```
> bears[-c(3,5,8,10,12),]
   length.cm weight.kg      loc
1      139.0       110 Bayfield
2      138.0        60 Bayfield
4      120.5        60 Bayfield
6      141.0       100  Ashland
7      141.0        95  Ashland
9      166.0       155  Douglas
11     129.5       105  Douglas
```