Lab: Manipulating Data frames

The “dataframe” is one of the most essential data structures used in R. It is conceptually equivalent to a database “relation” and to the typical rectangular dataset with variables as columns and cases as rows. For this activity, you will gain some skill with manipulating a dataframe.

Task 1:

R offers several built-in dataframes: For this activity we will use the “mtcars” dataset that contains 11 variables and 32 cases representing different models of cars.

The goal is to create a new variable for this dataframe that represents the engine displacement per cylinder in cubic inches for each vehicle. You may not know what displacement is (or maybe even cylinders), but it will suffice to know that values in the column named “disp” divided by values in the column named “cyl” will yield the appropriate quantity.

One fundamental principle of working with data is that you should never overwrite or change your original raw data. Therefore, your very first line of code should be:

```
my_mtcars <- mtcars  # Copy original dataframe into a new one
```

From that point forward you can work on my_mtcars without mucking up the original data. Also note that in order to establish that you have completed the assignment correctly, your last command should summarize your new variable using the summary() function. The output of that final command should look exactly like this:

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  17.77   26.92   34.48   35.03   43.19   59.00 
```

Task 2:

Gather some basic “demographic” information from about five friends or family members, and then enter those data into a data frame using the appropriate R commands. Finally, summarize the contents of the data frame, again using the appropriate R commands. Keep the demographics “light” to avoid getting too personal: For each person report 1) the number of pets that they have (dogs, cats, etc.); 2) their birth order in their family (i.e., 1 for first born, etc.); and 3) the number of siblings they have.

Collect the necessary data from your friends and family members, write, test, and submit the necessary code in R to accomplish the following:

1. Create three vectors of integers as described above, using the c( ) (concatenate) command to store data reported by group members, with these variable names: Pets, Order, and Siblings.
2. Also create a vector of user IDs for the friends and family members.
3. Bind those four vectors together into a data frame called myFriends.

4. Use the appropriate R command to report the structure of your data frame as well as a summary of the data (with minimums, means, maximums, etc. as shown on page 32. The result should show, “`X obs. Of 4 variables`,” where X is the number of friends and family members who reported their data.
5. Use the $ notation explained on page 33 to list all of the values for each of the variables in the myFriends data frame (example myGroup$Pets).

Hints: All of the examples that you need in order to write the necessary R commands are right there in Chapter 5. The most challenging part of this challenge will probably be getting the data from your friends and family members. Don’t wait too long! It’s okay if not everyone you ask participates. Use the user IDs of the friends and family members from item #2 above to keep track of who participated.

Learning Goals for this activity:

A. Create vectors of integers using the c( ) (concatenate) command.
B. Manipulating columns within a dataframe.
C. Calculating a new variable and attaching the new variable to a dataframe.
D. Practice using and interpreting the results of the summary() command.
E. Bind vectors together into a data frame.
F. Use the appropriate R command to report the structure of a data frame and summarize data.
G. Use of the $ notation to list values of variables.

Essential Guide for All IST687 Activities (appears at the end of all activity guides)

1. All IST687 activities work on what some people call a “constructivist learning” model. By developing a product on your own, testing it to find flaws, improving it, and comparing your solution to the solutions of other people, you can obtain a deeper understanding of a problem, the tools that might solve that problem, and a range of solutions that those tools may facilitate. The constructivist model only works to the extent that the student/learner has the drive to explore a problem, be frustrated, fail, try again, possibly fail again, and finally push through to a satisfactory level of understanding.
2. Each IST687 activity builds on skills and knowledge developed in the previous activities, so your success across the span of the course depends at each stage on your investment in earlier stages. Take the time to experiment, play, try new things, practice, improve, and learn as much as possible. These investments will pay off later.
3. Using the expertise of others, the Internet, and other sources of information is not only acceptable - it is expected. You must ***always, always, always*** give credit to your sources. For example, if you find a chunk of code from r-bloggers.com that helps you with developing a solution, by all means borrow that chunk of code, but

make sure to use a comment in your code to document the source of the borrowed code chunk. The discussion boards in the learning management system have been setup to encourage appropriate sharing of knowledge and wisdom among peers. Feel free to ask a question or pose a solution on these boards.

4. Building on the previous point, when submitting code as your solution to the activity, the comments matter at least as much, if not more than the code itself. A good rule of thumb is that every line of code should have a comment, and every meaningful block of code should be preceded by a comment block that is just about as long as the code itself. As noted above, you can use comments to give proper credit to your sources and you can use comments to identify your submission as your own.