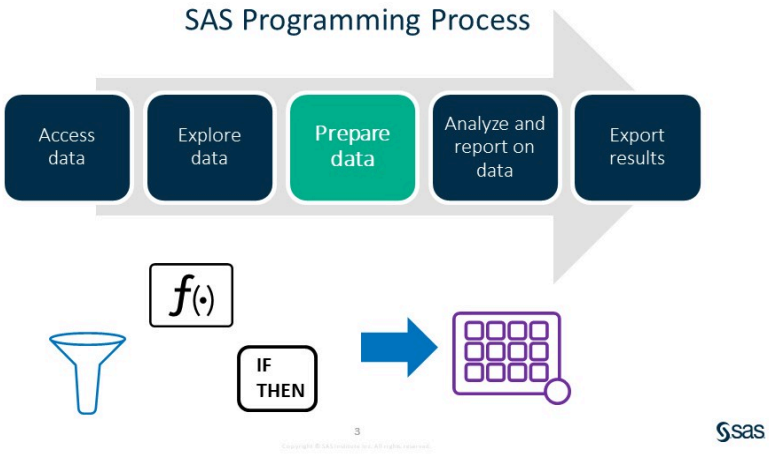
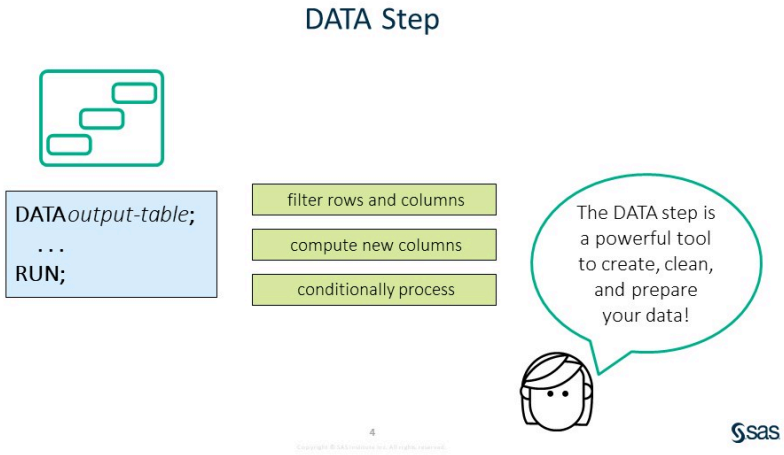
B4.1 - Reading and Filtering Data

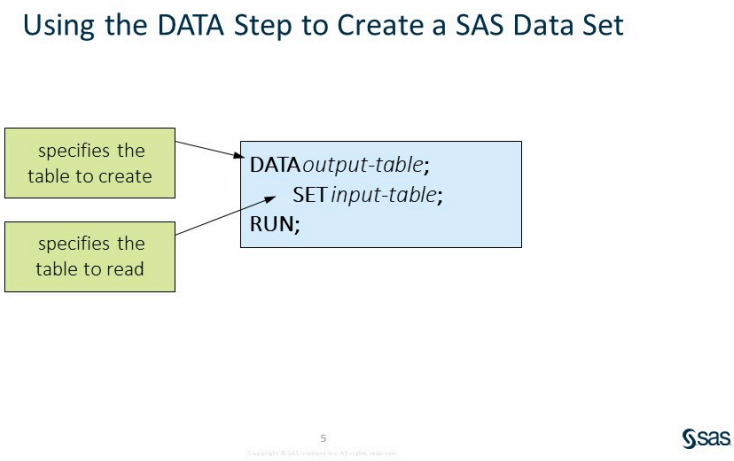


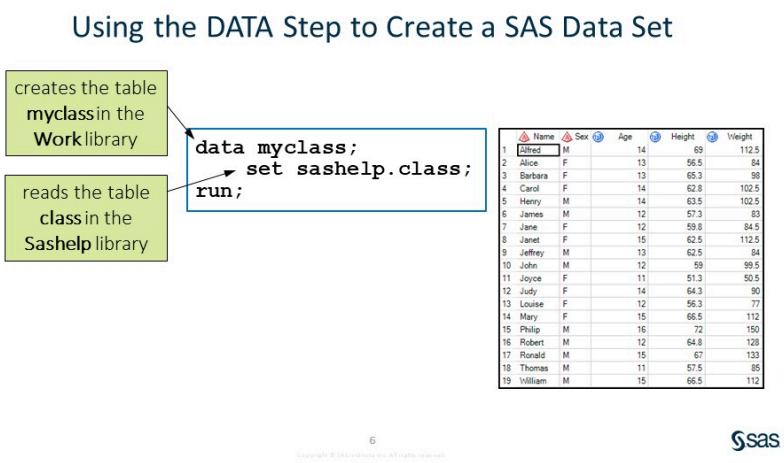
After you learn about your data, you'll probably want to make some adjustments based on what you find and what you need. This is where the DATA step really shines. In this lesson, you learn various ways to subset data and how to compute new columns using expressions and functions. You also learn how to use conditional processing to get exactly what you want in your output data.

Data Step

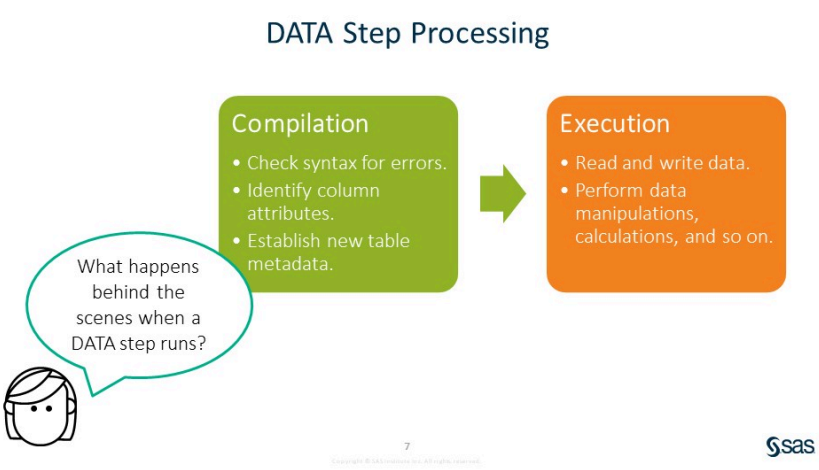


The DATA step is a powerful tool within the SAS programming language that you can use to manipulate data. It's rare that a data source that you start with is perfect as is and needs no preparation. Most analysts agree that manipulating and validating data takes much more time than reporting or analysis. For this reason, you'll appreciate that the DATA step is a robust yet simple programming tool that can do everything from simple querying to providing structure to messy web logs. Although we won't see everything the DATA step can do in this class, you learn how to do the most common data manipulation actions, such as filtering rows and columns, computing new columns, and performing conditional processing.

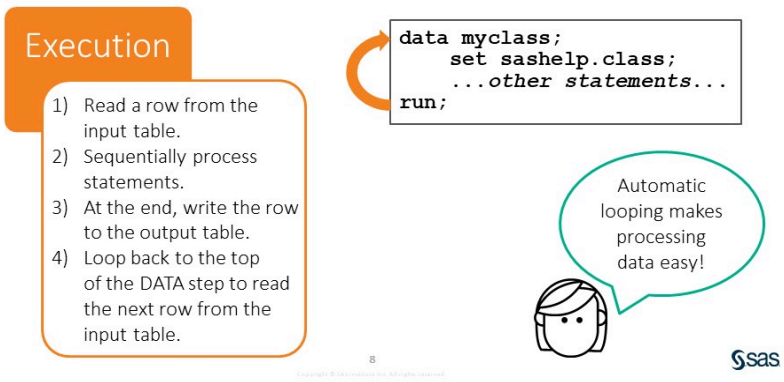
	Beyond these features, the DATA step also enables you to merge or join tables, read complex raw data, and perform repetitive processing with DO loops or arrays. These topics and many others are covered in SAS® Programming 2 and other advanced programming courses.
	When you work with data, you want to preserve your existing data and create a copy you can work on, so let's start with a simple DATA step that does just that. The DATA statement names the table you want to create, or the output table. This can be a temporary table if you use the **Work** library or a permanent table if you use any other library. Please be aware that if the table you list in the DATA statement exists and you have Write access to it, the DATA step overwrites that table. And and there is no undo! The SET statement names the existing table you are reading from, or the input table. When I reference a data source as *libref.table*, then based on a previous LIBNAME statement, SAS knows where to find the data source and how to read it. And of course the DATA step ends with a RUN statement.



In this code example, the DATA step creates the new table **myclass** in the **Work** library by reading the **class** table from the **Sashelp** library. The **myclass** table is an exact copy of **class**.



So how does the DATA step work behind the scenes? In this class, you need to have only a high-level understanding of the process. The DATA step has two phases: compilation and execution. In the compilation phase, SAS checks for syntax errors in the program and establishes the table metadata, such as column name, type, and length. In the execution phase, the data is read, processed, and written one row at time.



DATA step execution is like an automatic loop. The first time through the DATA step, the SET statement reads row number one from the input table and then processes any other statements in sequence, manipulating the values within that row. When SAS reaches the RUN statement, there is an implied OUPUT action, and the new row is written to the output table. The DATA step then automatically loops back to the top and executes the statements in order again, this time reading, manipulating, and outputting the second row. That implicit

	loop continues until all of the rows are read from the input table. As you learn more about the DATA step, it's helpful to have a deep understanding of this behind-the-scenes processing. The SAS® Programming 2 course addresses more complex DATA step code and covers the details of the compile and execute phases.

Activity 4.01

Open **p104a01.sas** from the **activities** folder and perform the following tasks:

1. Complete the DATA step to create a temporary table named **storm_new** and read **pg1.storm_summary**. Run the program and read the log.
2. Define a library named **out** pointing to the **output** folder in the main course files folder.
3. Change the program to save a permanent version of **storm_new** in the **out** library. Run the modified program.

Click here for Solution.

Note: The following activity is for you to answer questions and for you to get your response. It also serves as a checking point for if you are understanding the material. The following activity is not graded.

4.02 Question

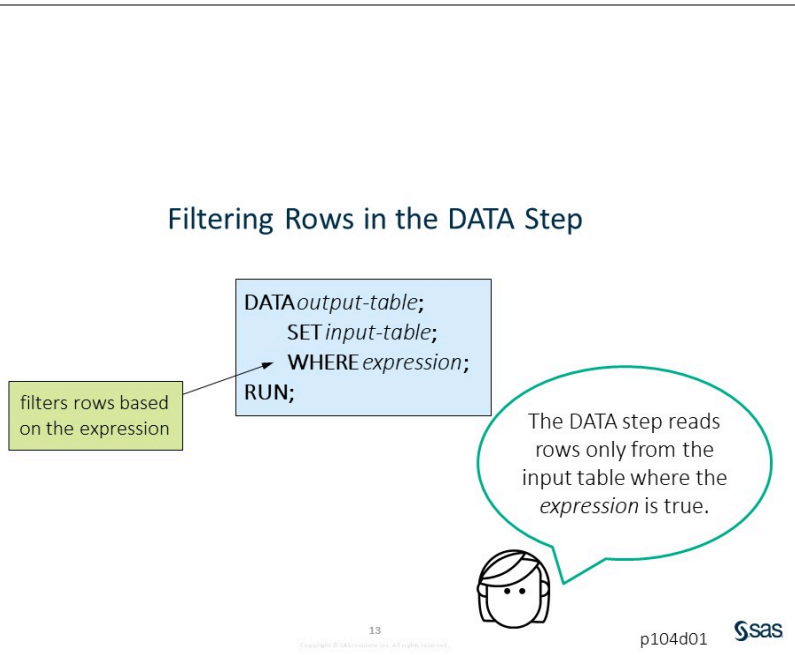
The table listed in the SET statement must be read via a library. Which data sources can be used in the SET statement? (check all that apply) 1 point

- [ ] SAS tables
- [ ] Excel spreadsheets
- [ ] DBMS tables
- [ ] comma-delimited files

Page 1 of 1

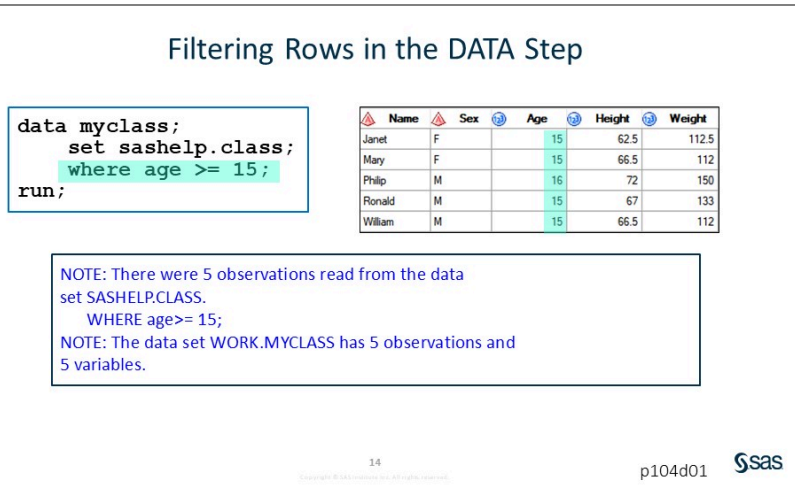
Submit Clear form

Filtering Rows in the DATA Step

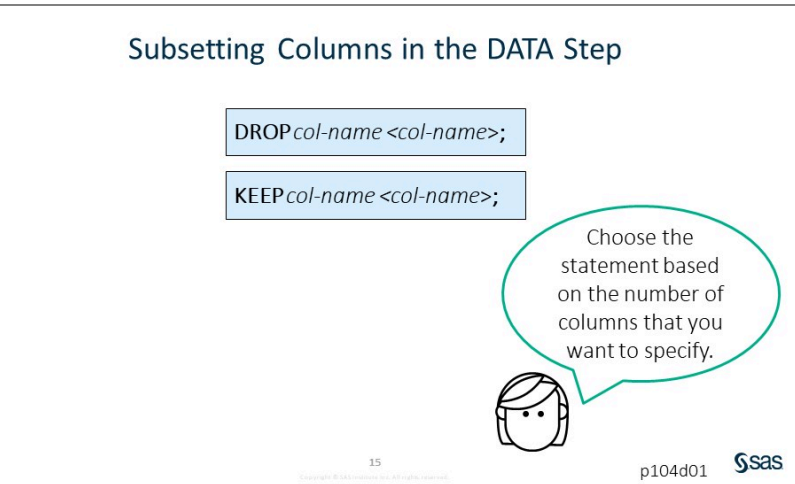


So now you have a copy of your messy data, but you undoubtedly want to filter or manipulate it in some way. The DATA step offers many different statements to help accomplish this task.

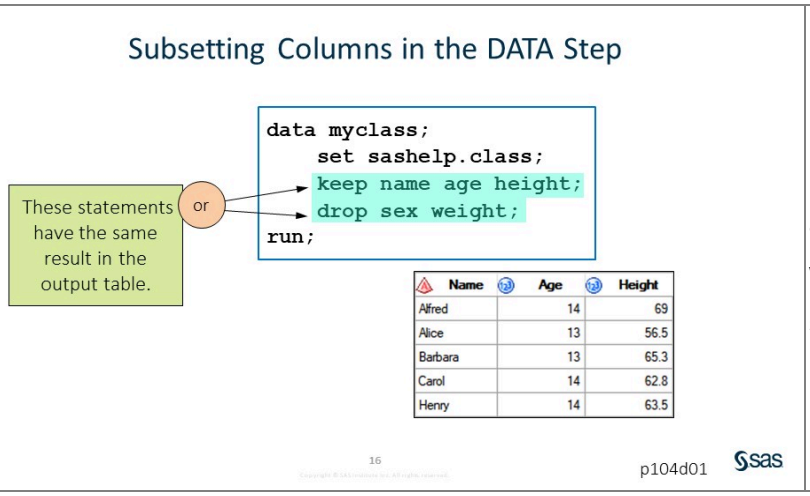
Let's start by subsetting the data with a WHERE statement. The same WHERE syntax that works in a procedure to subset data for a report or analysis works in the DATA step to filter rows. Only those rows from the input table that meet the criteria in the WHERE statement are processed by the DATA step and written to the output table.



In this example, the new table **myclass** includes only those rows that meet the WHERE criteria—that is, students with **age** greater than or equal to 15.



You can also specify the columns you want to include in your output data. To do this, you use either the DROP statement or the KEEP statement followed by the column names from the input table you want to drop or keep. It doesn't matter which statement you use, so choose the one that lets you specify fewer columns



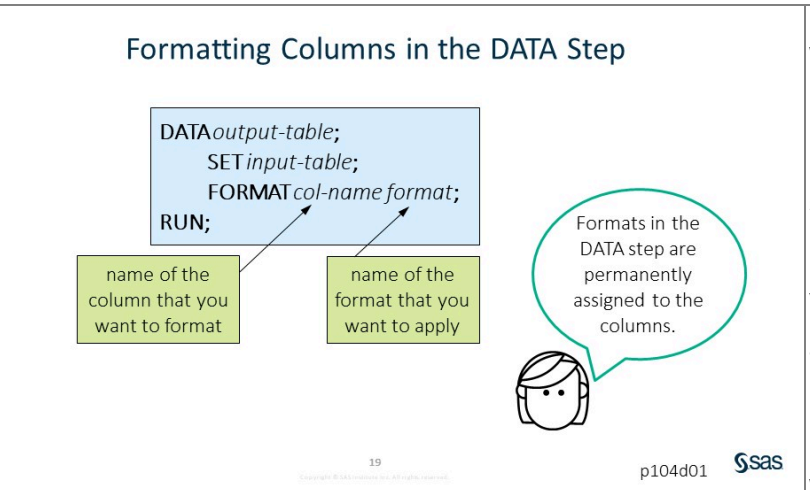
In this example, I can either keep **name**, **age**, and **height**, or drop **sex** and **weight**. Either statement returns the same output table.

Activity 4.03

Modify the program that you opened in the previous activity or open **p104a03.sas** from the **activities** folder and perform the following tasks:

1. Change the name of the output table to **storm_cat5**.
2. Include only Category 5 storms (**MaxWindMPH** greater than or equal to 156) with **StartDate** on or after 01JAN2000.
3. Add a statement to include the following columns in the output data: **Season**, **Basin**, **Name**, **Type**, and **MaxWindMPH**. How many Category 5 storms occurred since January 1, 2000?

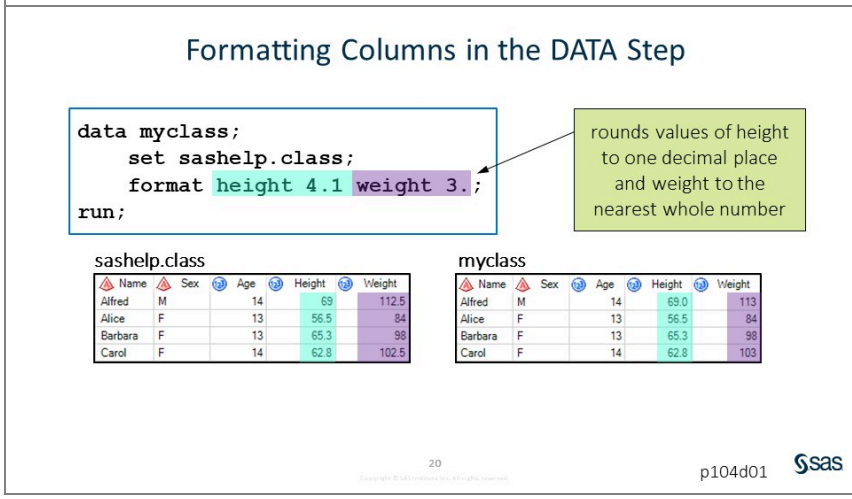
Click here for Solution.



While we are discussing columns, let's mention another statement that we can use in the DATA step: the FORMAT statement. The FORMAT statement is used in procedures to change how data values are displayed in a report or analysis.

We can use the same FORMAT statement in the DATA step, but the

impact is a little different. A FORMAT statement in the DATA step ***permanently*** assigns a format to a column in the properties of the new table. The raw data values are still stored in the table, but anytime you view the data or use it in procedures, the formats are automatically applied.



In this example code, the FORMAT statement applies the 4.1 format to **height** and the 3. format to **weight**. When you view the new table **myclass**, you see that **weight** values are rounded to the nearest whole number.