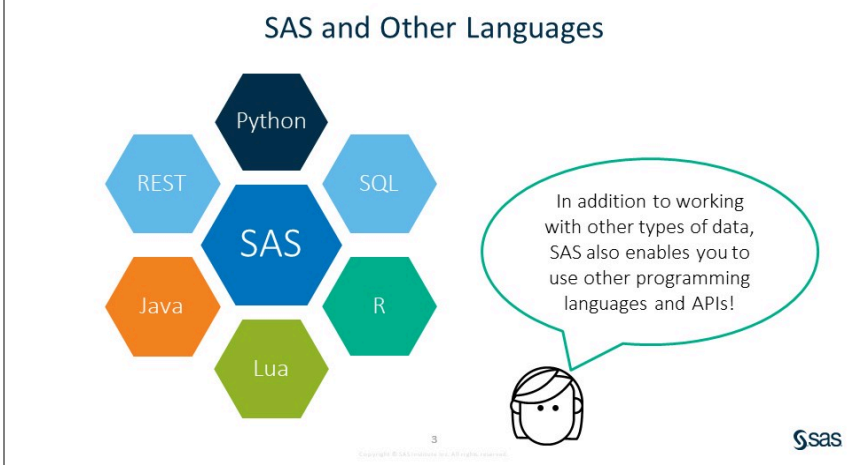
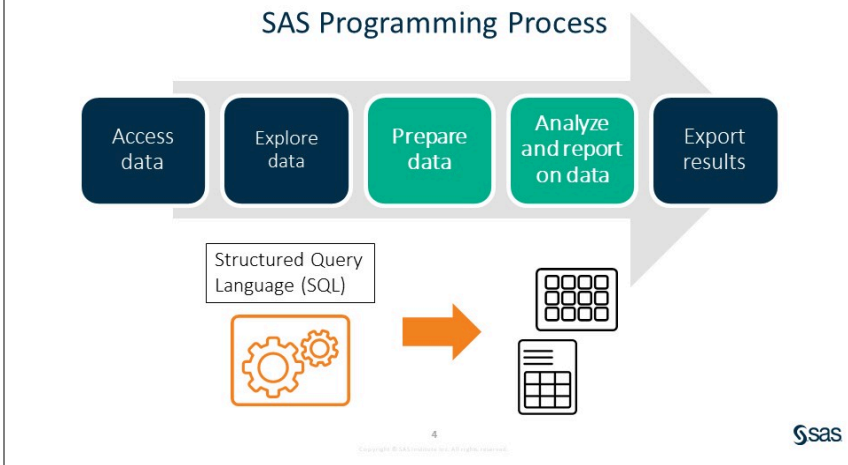
B7.1 - Using Structured Query Language (SQL) in SAS

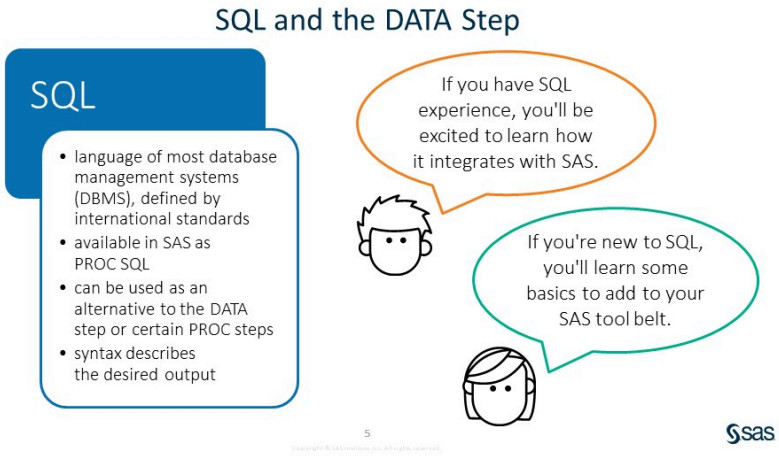


One of the great strengths of SAS has always been the ability to integrate with other types of data. We have seen in this course how SAS integrates with Excel and other Microsoft Office products. You can also read and write data from many other databases that are not part of SAS, including Oracle and Hadoop.

In addition to enabling you to use data from other sources, SAS also supports other common programming languages and APIs. You can take advantage of your knowledge and the strengths of these other languages in the code you submit in the SAS Platform.



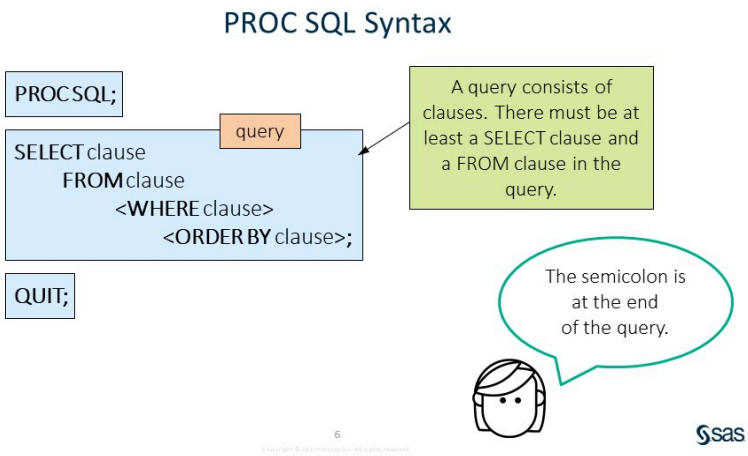
Structured Query Language (SQL) is a common language that is used by many programmers in a wide variety of software. SAS allows you to write SQL code as part of a SAS program. It's likely that you will come across SQL as you progress as a SAS programmer, so it's important to understand how SQL can be a beneficial tool, and how it compares to the SAS code you've learned to write so far. SQL is typically used in the “Prepare data” and “Analyze and report on data” phases of the SAS programming process.



SQL provides an alternative paradigm for processing and reporting on tabular data. Most database management systems use SQL as a common language to query, manipulate, and report on tables. Because of its widespread use and unique processing strengths, SQL can be used in SAS programs through a procedure, PROC SQL. Although DATA or PROC steps and SQL definitely have some overlap in terms of functionality, they process data differently behind the scenes, and they each have their advantages. That is why it is valuable to know and use both native SAS syntax and PROC SQL in your SAS programs.

For those of you who have experience writing SQL, we hope this lesson shows you how you can take advantage of your knowledge and see how it can integrate with the SAS code you have learned. And for those of you who are new to SQL, we will get you started with the fundamentals and motivate you to learn more.

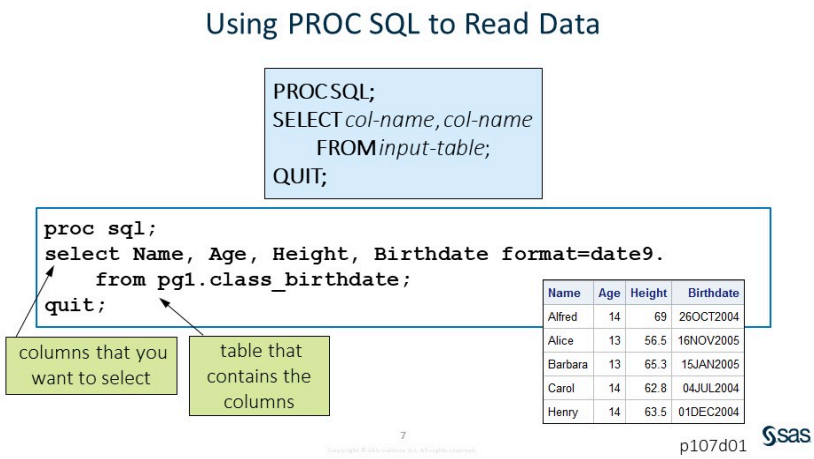
PROC SQL



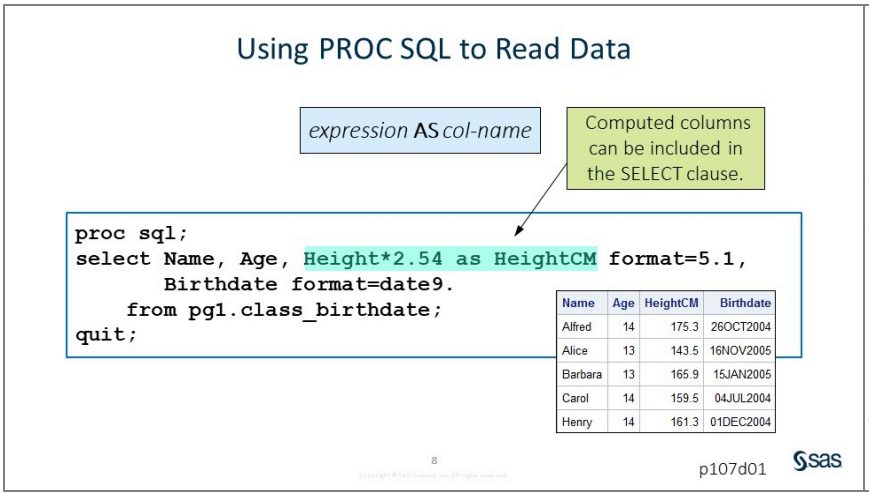
The PROC SQL statement invokes the SQL language processor, and subsequent statements are interpreted and executed as SQL until a QUIT statement is encountered.

SELECT is the most commonly used SQL statement and is usually referred to as a *query*. A query is made up of clauses that describe the desired result. At a minimum, a query must specify a list of column names to retrieve in the SELECT clause and the name of the table that contains the columns in the FROM clause. By default, an SQL query creates a report.

It is important to note that each SQL statement executes immediately and independently.



This simple query selects columns from the **class_birthdate** table and generates a report. The SELECT clause specifies the columns that you want to appear in the result, and the FROM clause specifies the table containing the source data. Notice that lists, such as column names, are always separated with commas. Also note the syntax applying a format to the **Birthdate** column. Although this is not standard SQL syntax, this SAS extension to the SQL language makes it easier to create more useful and polished reports.



In the SELECT statement, you can include computed columns in the list of columns. You simply define the expression that creates the column, and include the keyword AS and the column name. Again, the FORMAT= option is used to enhance how values are displayed in the report.

Activity 7.01:

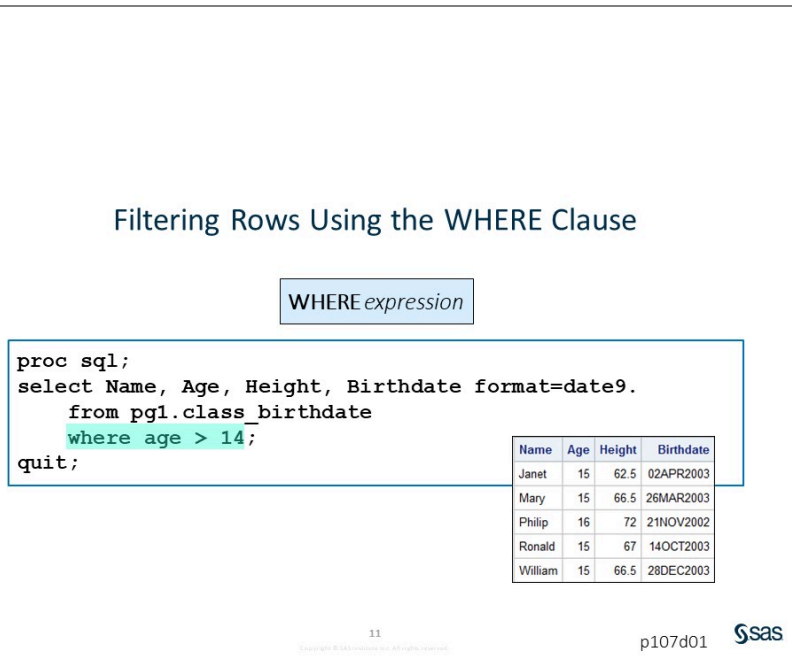
Open **p107a01.sas** from the **activities** folder.

1. What are the similarities and differences in the syntax of the two steps?
2. Run the program. What are the similarities and differences in the results?

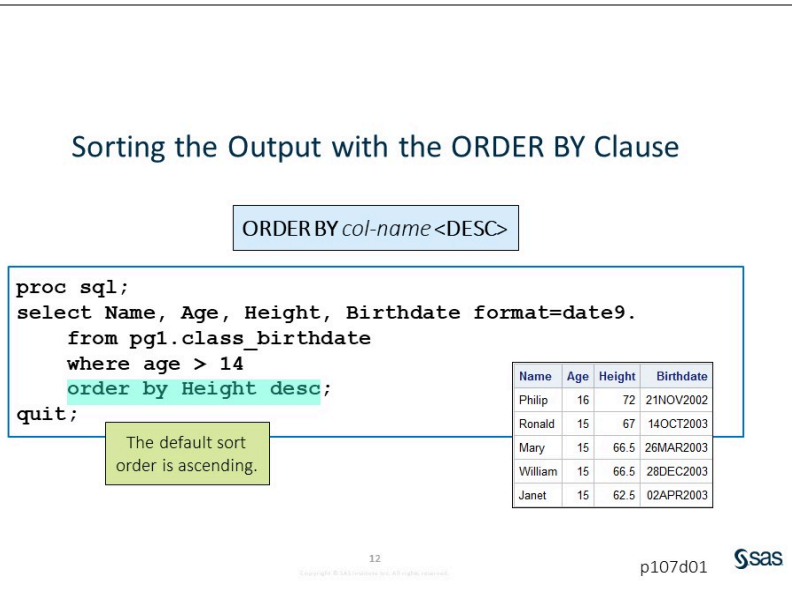
```
proc print data=pg1.class_birthdate;
    var Name Age Height Birthdate;
    format Birthdate date9.;
run;
```

```
proc sql;
select Name, Age, Height*2.54 as HeightCM format=5.1,
       Birthdate format=date9.
    from pg1.class_birthdate;
quit;
```

Click here for Solution.



So, what if I don't really want ***all*** of the rows in the table? Suppose I want to include only students who are more than 14 years old. I'll need to tell SQL how to choose only the rows I want. Let's add a WHERE clause to our SQL query. The same WHERE syntax that worked in other SAS procedures and the DATA step will work right here in SQL, too. However, remember that the WHERE expression is not a separate statement in SQL, but instead it's a clause added to the SELECT statement. Only those rows from the input table that meet the criterion provided are included in the report.



In traditional SAS syntax, if you want a report produced in a particular order, you must perform two separate steps. First sort the data, and then execute a reporting procedure. In SQL, we can do it all in one query. We can add an ORDER BY clause to describe the order in which we want the results arranged. If you want the rows ordered with the tallest person listed first (descending order), you would add the DESC keyword after the column name in the ORDER BY clause.

Demo: Reading and Filtering Data with SQL

7_1 - Demo - Reading and Filtering DAta with SQL.pdf
(https://clemson.instructure.com/courses/237270/files/23074703/download?wrap=1)
(https://clemson.instructure.com/courses/237270/files/23074703/download?download_frd=1)

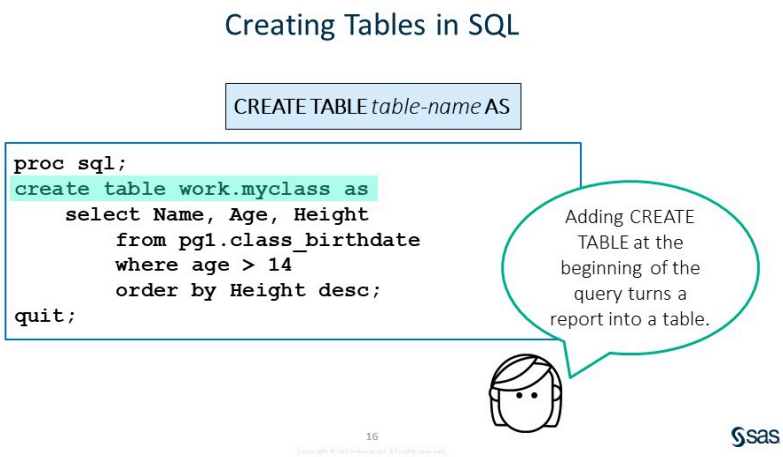
Activity 7.02:

Open **p107a02.sas** from the **activities** folder and perform the following tasks:

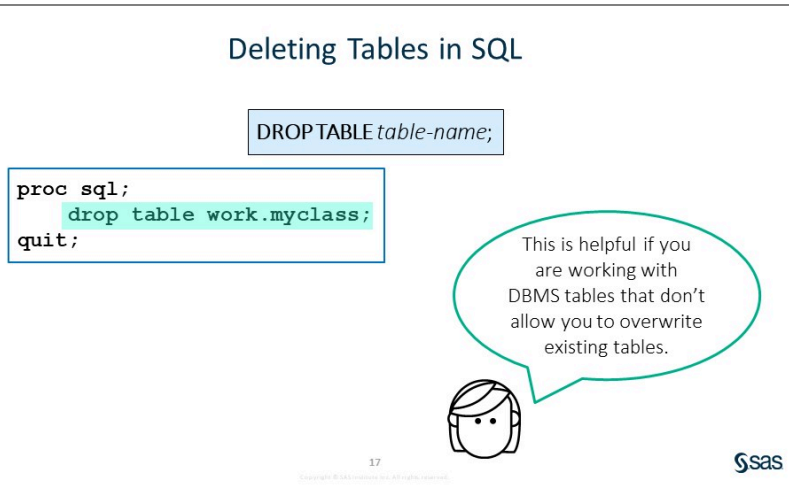
1. Complete the SQL query to display **Event** and **Cost** from **pg1.storm_damage**. Format the values of **Cost**.
2. Add a new column named **Season** that extracts the year from **Date**.
3. Add a WHERE clause to return rows where **Cost** is greater than 25 billion.
4. Add an ORDER BY clause to arrange rows by descending **Cost**.
   Which storm had the highest cost?

```
PROC SQL;
SELECT col-name, col-name <FORMAT=fmt.>, expression AS col-name
      FROM input-table
      WHERE expression
      ORDER BY col-name <DESC>;
QUIT;
```

Click here for Solution.



The report might look great, but you often need to create tables instead of reports. That's not a problem. SQL makes this task easy, too. To produce a table from the query result set instead of a report, you can change the SELECT statement to a CREATE TABLE statement.



For those writing SQL code for SAS to process in other database environments, you might need to drop or delete a table before updating it. If you have appropriate permission to make such changes within the database, you can use the DROP TABLE statement.