The Magic of PROC SQL

John Hotaling, Quintiles, Inc., Boston, MA

The SQL procedure provided by SAS places a great deal of power into a programmer’s hands. It can be used to examine existing data or to join data together in a manner which is sometimes preferable to merging.

The dataset being used in the examples is set up as follows:

```plaintext
/G47/G44/G57/G44
/G52/G51/G48/G1E
/G4C/G51/G53/G58/G57
/G57/G55/G57
/G53/G44/G57/G4C/G47
/G59/G4C/G56/G4C/G57
/G47/G52/G56/G48/G1E
/G46/G44/G55/G47/G56/G1E
/G13/G14
/G13/G14/G14/G14
/G13
/G13/G14/G13/G14/G15/G15
/G13
/G13/G14/G13/G14/G16/G16
/G13
/G13/G14/G13/G14/G17/G17
/G13
/G14/G14/G13/G15/G14/G14
/G18
/G14/G14/G13/G15/G15/G15
/G13
/G14/G14/G13/G15/G16/G16
/G13
/G14/G14/G13/G15/G17/G17
/G13
/G13/G14/G13/G16/G14/G14
/G13
/G13/G14/G13/G16/G15/G15
/G18
/G13/G14/G13/G16/G17/G17
/G13
/G13/G14/G13/G17/G14/G14
/G13
/G13/G14/G13/G17/G17/G16
/G18
/G13/G14/G13/G17/G17/G16
/G13
/G14/G14/G13/G29/G14/G19/G27/G28/G26/G19/G1C
/G13/G14/G2D/G24/G31/G1C/G19
/G14/G13/G15/G30/G14/G1C/G2D/G38/G31/G1A/G15
/G13/G15/G2D/G24/G31/G1C/G19
/G14/G13/G16/G30/G13/G17/G24/G33/G35/G1A/G14
/G13/G16/G2D/G24/G31/G1C/G19
/G14/G13/G18/G29/G13/G18/G32/G26/G37/G17/G1A
/G11
/G1E/G55/G58/G51/G1E
/G47/G44/G57/G44
/G57/G5A/G52/G1E
/G56/G48/G57
/G57/G5A/G52/G1E
/G49/G52/G55/G50/G44/G57/G47/G52/G45/G59/G47/G44/G57/G48/G0C/G0B/G47/G44/G57/G48/G1A/G11/G1E
/G55/G58/G51/G1E
/G47/G44/G57/G44
/G57/G5A/G52/G1E
/G54/G58/G4C/G57/G1E
/G53/G55/G52/G46/G56/G54/G4F/G1E
/G49/G55/G52/G50/G52/G51/G48
/G4A/G55/G52/G58/G53/G45/G5C/G57/G55/G57/G1E
/G54/G58/G4C/G57/G1E
/G33/G24/G37/G2C/G27
/G14/G13/G14
/G14/G13/G15
/G14/G13/G16
/G14/G13/G17
```

Examining the Data

One feature of the SQL procedure is to provide distinct values of key variables (similar to using the NODUPKEY option in a sort). The first examples show a listing of distinct PATID and then a sum of the distinct PATIDs.

```plaintext
data one;
  input trt patid visit dose;
  cards;
  0 101 1 10
  0 101 2 20
  0 101 3 30
  0 101 4 40
  1 102 1 15
  1 102 2 20
  1 102 3 30
  1 102 4 40
  0 103 1 10
  0 103 2 25
  0 103 3 35
  0 104 1 10
  0 104 4 45
;run;

data two;
  input patid sex $ (dob vdate) (date7. date7.);
  cards;
  101 F 16DEC69 01JAN86
  102 M 19JAN72 02JAN86
  103 M 04APR71 03JAN86
  105 F 03OCT47 .
;run;

data two;
  set two;
  format dob vdate date7.;
run;
```

```plaintext
title5 'COUNT of PATID';
proc sql;
  select count(distinct patid) as COUNT
  from one;
quit;
```

COUNT of PATID

```
4
```

The next two examples show how SQL will provide a summary of data grouped by another variable (in this case, the treatment).

```plaintext
title5 'COUNT of PATID per treat';
proc sql;
  select trt,count(distinct patid) as COUNT
  from one
  group by trt;
quit;
```

COUNT of PATID per treatment

```
TRT       COUNT
---------
0         3
1         1
```

```plaintext
title5 'SUM of DOSE per patient';
proc sql;
  select patid, sum(dose) as totdose
  from one
  group by patid;
quit;
```

SUM of DOSE per patient

```
PATID      TOTDOSE
---------
101        100
102        105
103         70
104         45
```

SQL Joins

In addition, SQL can be used to join datasets together rather than performing a merge. Key variables by which the datasets will be joined do not need to have the same variable name. The types of joins include inner, right outer, left outer, and full.

When working with multiple datasets, SQL needs to know which dataset to pull required variables from. This is specified in the "from" statement through the use of aliases. The appropriate alias is then placed in front of the variable name in the "select" statement. For these examples, aliases are a and b.

```plaintext
data one;
```

```plaintext
data two;
```

```plaintext
data two;
  set two;
  format dob vdate date7.;
run;
```
Inner joins keep only records where data comes from both datasets.

\[
\text{title5 'SQL Inner Join';}
\]
\[
\text{proc sql;}
\]
\[
\quad \text{select a.*, b.sex}
\]
\[
\quad \text{from one a, two b}
\]
\[
\quad \text{where (a.patid=b.patid);}
\]
\[
\text{quit;}
\]

SQL Inner Join

<table>
<thead>
<tr>
<th>TRT</th>
<th>PATID</th>
<th>VISIT</th>
<th>DOSE</th>
<th>SEX</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>101</td>
<td>1</td>
<td>10</td>
<td>F</td>
</tr>
<tr>
<td>0</td>
<td>101</td>
<td>2</td>
<td>20</td>
<td>F</td>
</tr>
<tr>
<td>0</td>
<td>101</td>
<td>3</td>
<td>30</td>
<td>F</td>
</tr>
<tr>
<td>0</td>
<td>101</td>
<td>4</td>
<td>40</td>
<td>F</td>
</tr>
<tr>
<td>1</td>
<td>102</td>
<td>1</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>102</td>
<td>2</td>
<td>20</td>
<td>II</td>
</tr>
<tr>
<td>1</td>
<td>102</td>
<td>3</td>
<td>30</td>
<td>II</td>
</tr>
<tr>
<td>1</td>
<td>102</td>
<td>4</td>
<td>40</td>
<td>II</td>
</tr>
<tr>
<td>0</td>
<td>103</td>
<td>1</td>
<td>10</td>
<td>II</td>
</tr>
<tr>
<td>0</td>
<td>103</td>
<td>2</td>
<td>25</td>
<td>II</td>
</tr>
<tr>
<td>0</td>
<td>103</td>
<td>4</td>
<td>35</td>
<td>II</td>
</tr>
</tbody>
</table>

Outer joins take data from which ever dataset is specified to be the "master" by using LEFT or RIGHT. A left join will have the first dataset mentioned being the master and a right join will have the second dataset being the master.

\[
\text{title5 'SQL Left Outer Join';}
\]
\[
\text{proc sql;}
\]
\[
\quad \text{select a.*, b.sex}
\]
\[
\quad \text{from one a left join two b}
\]
\[
\quad \text{on (a.patid=b.patid)};
\]
\[
\text{quit;}
\]

SQL Left Outer Join

<table>
<thead>
<tr>
<th>TRT</th>
<th>PATID</th>
<th>VISIT</th>
<th>DOSE</th>
<th>SEX</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>101</td>
<td>4</td>
<td>40</td>
<td>F</td>
</tr>
<tr>
<td>0</td>
<td>101</td>
<td>2</td>
<td>20</td>
<td>F</td>
</tr>
<tr>
<td>0</td>
<td>101</td>
<td>3</td>
<td>30</td>
<td>F</td>
</tr>
<tr>
<td>0</td>
<td>101</td>
<td>4</td>
<td>40</td>
<td>II</td>
</tr>
<tr>
<td>1</td>
<td>102</td>
<td>1</td>
<td>15</td>
<td>II</td>
</tr>
<tr>
<td>1</td>
<td>102</td>
<td>2</td>
<td>20</td>
<td>II</td>
</tr>
<tr>
<td>1</td>
<td>102</td>
<td>3</td>
<td>30</td>
<td>II</td>
</tr>
<tr>
<td>1</td>
<td>102</td>
<td>4</td>
<td>40</td>
<td>II</td>
</tr>
<tr>
<td>0</td>
<td>103</td>
<td>1</td>
<td>10</td>
<td>II</td>
</tr>
<tr>
<td>0</td>
<td>103</td>
<td>2</td>
<td>25</td>
<td>II</td>
</tr>
<tr>
<td>0</td>
<td>103</td>
<td>4</td>
<td>35</td>
<td>II</td>
</tr>
<tr>
<td>0</td>
<td>104</td>
<td>1</td>
<td>10</td>
<td>II</td>
</tr>
</tbody>
</table>

Full joins take records from either dataset which meet the criteria of the ON clause.

\[
\text{title5 'SQL Full Join';}
\]
\[
\text{proc sql;}
\]
\[
\quad \text{select a.*, b.sex}
\]
\[
\quad \text{from one a full join two b on a.patid=b.patid};
\]
\[
\text{quit;}
\]

SQL Full Join

<table>
<thead>
<tr>
<th>TRT</th>
<th>PATID</th>
<th>VISIT</th>
<th>DOSE</th>
<th>SEX</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>101</td>
<td>4</td>
<td>40</td>
<td>F</td>
</tr>
<tr>
<td>0</td>
<td>101</td>
<td>2</td>
<td>20</td>
<td>F</td>
</tr>
<tr>
<td>0</td>
<td>101</td>
<td>3</td>
<td>30</td>
<td>F</td>
</tr>
<tr>
<td>0</td>
<td>101</td>
<td>1</td>
<td>10</td>
<td>II</td>
</tr>
<tr>
<td>1</td>
<td>102</td>
<td>2</td>
<td>20</td>
<td>II</td>
</tr>
<tr>
<td>1</td>
<td>102</td>
<td>3</td>
<td>30</td>
<td>II</td>
</tr>
<tr>
<td>1</td>
<td>102</td>
<td>1</td>
<td>15</td>
<td>II</td>
</tr>
<tr>
<td>0</td>
<td>103</td>
<td>4</td>
<td>40</td>
<td>II</td>
</tr>
<tr>
<td>0</td>
<td>103</td>
<td>1</td>
<td>10</td>
<td>II</td>
</tr>
<tr>
<td>0</td>
<td>103</td>
<td>2</td>
<td>25</td>
<td>II</td>
</tr>
<tr>
<td>0</td>
<td>104</td>
<td>4</td>
<td>35</td>
<td>II</td>
</tr>
<tr>
<td>0</td>
<td>104</td>
<td>1</td>
<td>10</td>
<td>II</td>
</tr>
</tbody>
</table>

Creating New Tables

As a final example, a new SQL table, which functions similarly to a SAS dataset, is created by selecting variables from multiple datasets, renaming variables, calculation of new variables, sorting, and the use of a complex WHERE statement.

Every variable is selected from the dataset a, sex is renamed as gender, and age is calculated from the vdate and dob variables. Finally, only records where vdate is not missing and the patid variables match are included in the final table.

\[
\text{proc print;}
\]
\[
\quad \text{title5 'Creation of new table with aliases, ordering, and complex WHERE clause';}
\]
\[
\text{run;}
\]
proc sql;
create table three as
    select a.*, b.sex as gender, vdate, dob, (vdate-
dob)/365.25
    as age
from one a, two b
where (a.patid-b.patid) and (vdate is not
    missing)
order by trt, patid;
quit;

Creation of new table with aliases, ordering, and
complex WHERE clause

<table>
<thead>
<tr>
<th>TRT</th>
<th>PATID</th>
<th>VISIT</th>
<th>DOSE</th>
<th>SEX</th>
<th>VDATE</th>
<th>DOB</th>
<th>AGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>101</td>
<td>3</td>
<td>30</td>
<td>F</td>
<td>01JAN96</td>
<td>16DEC96</td>
<td>26.0424</td>
</tr>
<tr>
<td>0</td>
<td>101</td>
<td>1</td>
<td>10</td>
<td>F</td>
<td>01JAN96</td>
<td>16DEC96</td>
<td>26.0424</td>
</tr>
<tr>
<td>0</td>
<td>101</td>
<td>4</td>
<td>40</td>
<td>F</td>
<td>01JAN96</td>
<td>16DEC96</td>
<td>26.0424</td>
</tr>
<tr>
<td>0</td>
<td>101</td>
<td>2</td>
<td>20</td>
<td>F</td>
<td>01JAN96</td>
<td>16DEC96</td>
<td>26.0424</td>
</tr>
<tr>
<td>0</td>
<td>103</td>
<td>2</td>
<td>25</td>
<td>M</td>
<td>03JAN96</td>
<td>04APR71</td>
<td>24.7502</td>
</tr>
<tr>
<td>0</td>
<td>103</td>
<td>1</td>
<td>10</td>
<td>M</td>
<td>03JAN96</td>
<td>04APR71</td>
<td>24.7502</td>
</tr>
<tr>
<td>0</td>
<td>103</td>
<td>4</td>
<td>35</td>
<td>M</td>
<td>03JAN96</td>
<td>04APR71</td>
<td>24.7502</td>
</tr>
<tr>
<td>1</td>
<td>102</td>
<td>3</td>
<td>30</td>
<td>M</td>
<td>02JAN96</td>
<td>19JUN72</td>
<td>23.5373</td>
</tr>
<tr>
<td>1</td>
<td>102</td>
<td>2</td>
<td>20</td>
<td>M</td>
<td>02JAN96</td>
<td>19JUN72</td>
<td>23.5373</td>
</tr>
<tr>
<td>1</td>
<td>102</td>
<td>4</td>
<td>40</td>
<td>M</td>
<td>02JAN96</td>
<td>19JUN72</td>
<td>23.5373</td>
</tr>
<tr>
<td>1</td>
<td>102</td>
<td>1</td>
<td>15</td>
<td>M</td>
<td>02JAN96</td>
<td>19JUN72</td>
<td>23.5373</td>
</tr>
</tbody>
</table>

These examples should give some insight into the capabilities of
the SQL procedure and how powerful it can be. As with any
power, however, it should be used wisely!

**Trademarks**

SAS is a registered trademark of SAS Institute, Inc., Cary, NC.

For further information, the author can be reached at
jhotalin@qcamb.quintiles.com or at

Quintiles, Inc,
465 Medford Street, Suite 300
Boston, MA 02129
(617) 886-9000