Paper 2100-2016
Super boost data transpose puzzle
Ahmed Al-Attar, AnA Data Warehousing Consulting LLC, McLean, VA

ABSTRACT
This paper compares different solutions to a data transpose puzzle presented to the SAS® User Group at the US Census Bureau (CenSAS). The presented solutions ranged from a SAS® 101 multi-step solution to an advanced solution utilizing not widely known techniques yielding 85% run time savings!

INTRODUCTION
While working with the International Data Base (IDB) data at the US Census Bureau, I needed to find a way to transpose a wide horizontal table with 101 numeric columns and two rows per group, into a thinner table with 101 rows per group and new column representing the aggregation of two columns per row.

<table>
<thead>
<tr>
<th>scheme</th>
<th>regno</th>
<th>CTY</th>
<th>YR</th>
<th>SEX</th>
<th>MAXAGE</th>
<th>P000</th>
<th>P001</th>
<th>P002</th>
<th>P003</th>
<th>P004</th>
<th>....</th>
<th>....</th>
<th>....</th>
<th>P100</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>AA</td>
<td>1981</td>
<td>2</td>
<td>100</td>
<td>553</td>
<td>559</td>
<td>540</td>
<td>516</td>
<td>493</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>AA</td>
<td>1981</td>
<td>3</td>
<td>100</td>
<td>529</td>
<td>536</td>
<td>517</td>
<td>494</td>
<td>472</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>AA</td>
<td>1982</td>
<td>2</td>
<td>97</td>
<td>549</td>
<td>551</td>
<td>558</td>
<td>539</td>
<td>516</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>AA</td>
<td>1982</td>
<td>3</td>
<td>100</td>
<td>561</td>
<td>528</td>
<td>536</td>
<td>517</td>
<td>493</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>AA</td>
<td>1983</td>
<td>2</td>
<td>97</td>
<td>569</td>
<td>547</td>
<td>550</td>
<td>559</td>
<td>539</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>AA</td>
<td>1983</td>
<td>3</td>
<td>100</td>
<td>582</td>
<td>559</td>
<td>528</td>
<td>536</td>
<td>517</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>AA</td>
<td>1984</td>
<td>2</td>
<td>98</td>
<td>585</td>
<td>566</td>
<td>547</td>
<td>549</td>
<td>559</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>AA</td>
<td>1984</td>
<td>3</td>
<td>100</td>
<td>600</td>
<td>580</td>
<td>559</td>
<td>527</td>
<td>536</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>AA</td>
<td>1985</td>
<td>2</td>
<td>99</td>
<td>587</td>
<td>577</td>
<td>562</td>
<td>542</td>
<td>545</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>AA</td>
<td>1985</td>
<td>3</td>
<td>100</td>
<td>603</td>
<td>598</td>
<td>578</td>
<td>557</td>
<td>526</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>AA</td>
<td>1986</td>
<td>2</td>
<td>100</td>
<td>577</td>
<td>580</td>
<td>571</td>
<td>556</td>
<td>536</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 1: Original IDB Birth Rate table

Original data characteristics:
- **Sorted By**: Scheme, regno, cty, yr, sex
- There are 101 age variables (P000 – P100)
- Each Combination of Scheme, regno, cty, yr has a row for **Male** (Sex=2) and a row for **Female** (Sex=3)
- 240,616 observations

Resulting data table should have
- **Male & Female** Records transposed
- **Both** Column is calculated by summing Male & Female numbers
- popAge is 0-100
- 12+ Millions observation ((240,616/2) * 101)
Figure 2: Desired Transposed Birth Rate table

It was very important to come up with a solution that has the least amount of data reads and the fastest run time.

Once I developed my solution, I wanted to validate it and see if there could be a better solution. That's when I reached out to the SAS® users group at the US Census Bureau (CenSAS), and posted it as a Data Transpose Puzzle. I got two helpful solutions with different levels of complexity, and approach.
**SOLUTION 1**
This solution was contributed by a Branch Chief at the Census. It consisted of 3 steps (Data Step, Proc Sort, Data Step).

```sas
%let g_srcDsName = censas.idb194;

data b (keep = scheme regno cty yr popage pop);
  set &g_srcDsName;
  array popagea(101) 8 p000-p100;
  do i = 1 to 101;
    popage = i - 1;
    if sex = '2' then pop = popagea{i};
    else if sex = '3' then pop = popagea{i};
    output;
  end;
run;

proc sort data=b;
  by scheme regno cty yr popage;
run;

data c (drop = pop);
  set b;
  by scheme regno cty yr popage;
  retain male female;
  if first.popage then male = pop;
  if last.popage then
    do;
      female = pop;
      both = male + female;
      output;
    end;
  end;
run;
```

**Pros:**
- Clear & Simple
- Easy to follow
- Adoptable by levels of skills

**Cons:**
- Ignored data characteristics (Sorted by)
- Multiple data reads
- Intermediate table
- Typical SAS® 101 approach
- Suitable for teaching but not for Production deployment
SOLUTION 2

This solution was contributed by a SAS® on-site Technical Advisor at the Census Bureau. It consisted of 2 steps (Proc Transpose, Proc SQL).

```sas
%let g_srcDsName = censas.idb194;

Proc transpose data=&g_srcDsName( drop=maxage sex)
  out=WORK.alattartest2( rename= (sex_1=Male 
  sex_2=Female))
  name=Popagetem 
  prefix=sex_; 
  by scheme regno CTY YR;
run;

Proc sql;
create table final as 
select scheme ,regno ,CTY ,YR 
,substr(Popagetem,2,4) as Popage 
,Male ,Female ,sum(Male+Female) as both 
from alattartest2;
quit;
```

Pros:
- Utilized data characteristics (Sorted by)
- Easy to follow
- Mixing Procs & SQL

Cons:
- Multiple data reads
- Proc Transpose performs multiple internal data reads
- Intermediate table
SOLUTION 3

This is the solution I had developed prior to ask for input from the SAS® users group. It consisted of a single step (Data Step).

```sas
%let g_srcDsName = censas.idb194;

DATA WORK.idb194_transposed (KEEP=scheme regno cty yr Popage male female both);
   ARRAY p P000-P100;
   ARRAY MP MP000-MP100;
   ARRAY FP FP000-FP100;
   LENGTH  pFirstPos $20 init_rb8_str $808;
   RETAIN MP: FP: init_rb8_str;
   if (_n_=1) then do;
      pFirstPos = ADDRLONG(p000);
      init_rb8_str = PEEKCLONG(ADDRLONG(p000));
   end;

   SET &g_srcDsName;
   BY   scheme regno cty yr;
   LENGTH   Popage 4 male female both 8;
   LABEL   PopAge = 'Population at Age'
      male   = 'Male Population'
      female = 'Female Population'
      both   = 'Both sexes Population';

   /* Find the destination address */
   if (sex=2) then   call POKECLONG(PEEKCLONG(ADDRLONG(p000)),ADDRLONG(mp000),808);
   else   call POKELONG(PEEKCLONG(ADDRLONG(p000)),ADDRLONG(fp000),808);

   IF (last.yr) THEN
   DO i=1 to dim(p);
      popAge = i-1;
      male   = mp[i];
      female = fp[i];
      both   = Sum(male,female);
      OUTPUT;
   END;
RUN;
```
**Pros:**
- Utilized data characteristics (Sorted by)
- Single data read
- Single Output – No Intermediate table(s)

**Cons:**
- Uses unfamiliar yet very powerful functions
- Requires advanced skills level

**RUN TIMES COMPARISON**

In order to have a fair and comprehensive comparison, I used the same two data sets with all three solutions on the same SAS® platform. The table below illustrates the differences in their run times.

<table>
<thead>
<tr>
<th>Table Sizes</th>
<th>Solution 1 3 Steps mm:ss.ss</th>
<th>Solution 2 2 Steps mm:ss.ss</th>
<th>Solution 3 1 Step mm:ss.ss</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>run-time</td>
<td>% of S1</td>
<td>run-time</td>
</tr>
<tr>
<td><strong>Sample Data (2,310 records)</strong></td>
<td>00:00.36</td>
<td>27.78%</td>
<td>00:00:08</td>
</tr>
<tr>
<td><strong>Full Data (240,616 records)</strong></td>
<td>00:28.71</td>
<td>17.10%</td>
<td>00:03.42</td>
</tr>
</tbody>
</table>

Table 1: Solutions run times and percent of improvement

**CONCLUSION**

Some of the unwieldy known features of the SAS® programming language can offer more elegant and efficient solutions to data manipulation problems.

Whether you are a SAS® programmer with many years of experience or a novice user who is responsible for maintaining legacy programs, implementing updated approaches can allow you to streamline your SAS® applications, expedite the development and debugging process, and minimize future maintenance of the code.

Investigating and researching alternative approaches and solutions is a worthwhile investment for any SAS® programmer, regardless of their level of experience.
REFERENCES


CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Ahmed Al-Attar
AnA Data Warehousing Consulting, LLC.
McLean, VA 22101
Cell Phone: 703-477-7972
E-mail: ahmed.al-attar@anadwc.com
Web: www.anadwc.com

TRADEMARKS

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.