Executing Code Efficiently

Alex Brink, Jaco van Blerk

Topics

- Introduction
- *Traditional* use of Call Execute
- What does SAS actually do?
- A new way of thinking
- An actual example
- Questions

Introduction

- SAS is extremely powerful… we only touch the surface
- Creating efficient code is generally achieved through
  - Concise, robust algorithms
  - Limiting unrequired variables/observations
- What if we only created necessary code?
- Call Execute gives us the ability to create dynamic programming statements
- And allows us to execute code efficiently

Traditional use of Call Execute (1)

```
%macro datasets(name=);
      data work.&name;
            set sashelp.class;
            where name="&name";
      run;
%mend datasets;
```

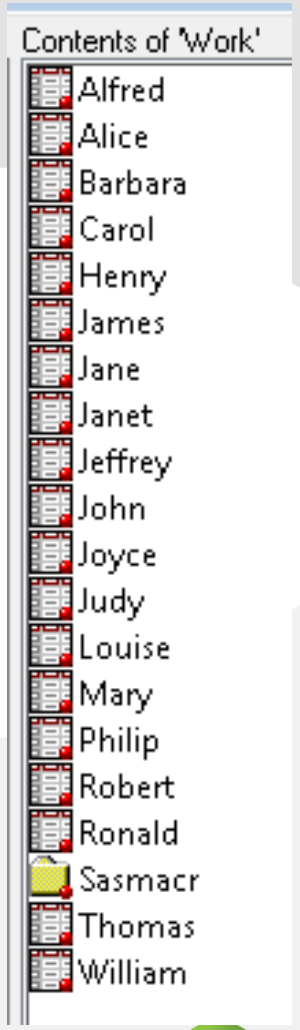
Option 1

```
%datasets(name=Alfred);
%datasets(name=Alice);
...
%datasets(name=William);
```

Option 2

```
data _null_;
      set sashelp.class;
      call execute('%datasets(name='||strip(name)||');');
run;
```

	Name	Sex	Age	Height	Weight
1	Alfred	M	14	69	112.5
2	Alice	F	13	56.5	84
3	Barbara	F	13	65.3	98
4	Carol	F	14	62.8	102.5
5	Henry	M	14	63.5	102.5
6	James	M	12	57.3	83
7	Jane	F	12	59.8	84.5
8	Janet	F	15	62.5	112.5
9	Jeffrey	M	13	62.5	84
10	John	M	12	59	99.5
11	Joyce	F	11	51.3	50.5
12	Judy	F	14	64.3	90
13	Louise	F	12	56.3	77
14	Mary	F	15	66.5	112
15	Philip	M	16	72	150
16	Robert	M	12	64.8	128
17	Ronald	M	15	67	133
18	Thomas	M	11	57.5	85
19	William	M	15	66.5	112



Traditional use of Call Execute (2)

```
data _null_;
      set sashelp.class;
      where age>=14;
      call execute('%datasets(name='||strip(name)||');');
run;
```

```
19   + data work.William;         set sashelp.class;         where name="William";      run;

NOTE: There were 1 observations read from the data set SASHELP.CLASS.
      WHERE name='William';
NOTE: The data set WORK.WILLIAM has 1 observations and 5 variables.
NOTE: DATA statement used (Total process time):
      real time           0.00 seconds
      cpu time            0.00 seconds
```

What does SAS actually do

- *Execute* places the value of its argument in the program stack, on a first in first out basis
- Execution of SAS statements generated by the execution of the macro will be delayed until after a step boundary
- If an EXECUTE routine argument is a macro invocation or resolves to one, the macro executes immediately [but not the actual macro call]
- SAS macro statements, including macro variable references, will execute immediately [again though, not the actual programming statements]

A new way of thinking

```
data _null_;
        set sashelp.class;
        call execute('data work.'||strip(name)||'; set sashelp.class; where
name="'||strip(name)||'"; run;');
run;
```

```
19   + data work.William; set sashelp.class; where name="William"; run;

NOTE: There were 1 observations read from the data set SASHELP.CLASS.
      WHERE name='William';
NOTE: The data set WORK.WILLIAM has 1 observations and 5 variables.
NOTE: DATA statement used (Total process time):
      real time           0.00 seconds
      cpu time            0.00 seconds
```

```
data _null_;
        set sashelp.class;
        call execute('data work.'||strip(name)||';');
        call execute('set sashelp.class;');
        call execute('where name="'||strip(name)||'";');
        call execute('run;');
run;
```

```
73   + data work.William;
74   + set sashelp.class;
75   + where name="William";
76   + run;

NOTE: There were 1 observations read from the data set SASHELP.CLASS.
      WHERE name='William';
NOTE: The data set WORK.WILLIAM has 1 observations and 5 variables.
NOTE: DATA statement used (Total process time):
      real time           0.00 seconds
      cpu time            0.00 seconds
```

A practical example (1)

```
data _null_;
    set content;
    by memname;

    if first.memname then do;
        call execute('data work.newclass;');
        call execute('set sashelp.class;');
    end;

        if dropfl="Y" then call execute('drop ' || strip(name) || ';');
        if ^missing(rename) then call execute('rename ' || strip(name) || '=' || strip(rename) || ';');
        if ^missing(convert) then do;
            if type=1 then call execute(strip(convert) || '=strip(put(' || strip(name) || ',best32.));');
            else if type=2 then call execute(strip(convert) || '=input(' || strip(name) || ',??best32.));');;
        end;

    if last.memname then call execute('run;');

run;
```

	Library Member Name	Variable Name	Variable Type	dropfl	rename	convert
1	CLASS	Age	1			agechar
2	CLASS	Height	1	Y		
3	CLASS	Name	2		subject	
4	CLASS	Sex	2			
5	CLASS	Weight	1			

A practical example (2)

	subject	Sex	Age	Weight	agechar
1	Alfred	M	14	112.5	14
2	Alice	F	13	84	13
3	Barbara	F	13	98	13
4	Carol	F	14	102.5	14
5	Henry	M	14	102.5	14
6	James	M	12	83	12
7	Jane	F	12	84.5	12
8	Janet	F	15	112.5	15
9	Jeffrey	M	13	84	13
10	John	M	12	99.5	12
11	Joyce	F	11	50.5	11
12	Judy	F	14	90	14
13	Louise	F	12	77	12
14	Mary	F	15	112	15
15	Philip	M	16	150	16
16	Robert	M	12	128	12
17	Ronald	M	15	133	15
18	Thomas	M	11	85	11
19	William	M	15	112	15

	Library Member Name	Variable Name	Variable Type	dropfl	rename	convert
1	CLASS	Age	1			agechar
2	CLASS	Height	1	Y		
3	CLASS	Name	2		subject	
4	CLASS	Sex	2			
5	CLASS	Weight	1			

Questions

References

- PhUSE 2014, Paper CC06, Call Execute: Let Your Program Run Your Macro (Artur Usov, OCS Consulting BV, ’s-Hertogenbosch, Netherlands)