

SAS 9 Update – New Functionalities for SAS Programmers

Featuring:

PROC FCMP, Hash Object, Using Java/C in SAS, ODS Graphics Designer, Enterprise Guide

PhUSE Day, 4th of May, Frankfurt
Gregor Herrmann
SAS Germany



**THE
POWER
TO KNOW.**

SAS 9 Update

New Functionalities for SAS Programmers

- **User defined functions**
- **External functions**
- **Java Object**
- **DATA Step Hash Object**
- **ODS Graphics Designer**
- **Enterprise Guide**

SAS 9 Update

New Functionalities for SAS Programmers

- **User defined functions**
- External functions
- Java Object
- DATA Step Hash Object
- ODS Graphics Designer
- Enterprise Guide

The FCMP Procedure

User defined functions

- Enables programmers to create and test SAS functions and subroutines using DATA step syntax
- Enables programmers to store these functions in a package within a SAS data set
- Enables programmers to more easily read, write, and maintain complex code with independent and reusable subroutines.
- Functions and subroutines can be referenced anywhere where SAS normally allows functions to be used (DATA step, WHERE statements, etc.)

Advantages of PROC FCMP

- The advantages of using the FCMP procedure include the following:
 - A library of **reusable** routines can be built using DATA step syntax. Routines can be reused in any SAS program that has access to their storage locations.
 - Complex programs can be **simplified** by abstracting common computations into named program units. FCMP routines enable programmers to read, write, and maintain complex code easily.
 - FCMP routines are **independent** from their use because any program that calls a routine is not affected by the routine's implementation.

The FCMP Procedure

A simple Example

Creating a Function

```
proc FCMP outlib = sasuser.funcs.trial;  
  function Study_Day(intervention_date, event_date);  
    If event_date < intervention_date then  
      return(event_date - intervention_date);  
    else  
      return(event_date - intervention_date + 1);  
  endsub;  
run;
```

Calling a Function

```
options cmplib = sasuser.funcs;  
data _null_;  
  start = '15Feb2006'd;  
  today = '27Mar2006'd;  
  sd = study_day(start, today);  
  put sd=;  
run;
```

Returned value for
sd was 41

```
152 Options cmplib = sasuser.funcs;  
153 Data _null_;  
154   start = '15Feb2006'd;  
155   today = '27Mar2006'd;  
156   sd = study_day(start, today);  
157  
158   put sd=;  
159   Run;  
  
sd=41  
NOTE: DATA statement used (Total process time):  
      real time           0.06 seconds  
      cpu time            0.04 seconds
```

SAS 9 Update

New Functionalities for SAS Programmers

- User defined functions
- **External functions**
- Java Object
- DATA Step Hash Object
- ODS Graphics Designer
- Enterprise Guide

The PROTO Procedure

External Functions

- PROC PROTO enables you to register, in batch mode, external functions that are written in the C or C++ programming languages.

PROC PROTO PACKAGE=catalog-entry <option(s)>;

MAPMISS type1=value1 type2=value2 ...;

LINK load-module;

MAPMISS- Specify alternative values, by type, to pass to functions if values are missing.

LINK- Specify the name, path, and load module that contains your functions.

PROC PROTO Example

This example shows how to use PROC PROTO to prototype two external C language functions called SPLIT and CASHFLOW. These functions are contained in the shared library that is specified by the LINK statement.

```
proc proto package = sasuser.myfuncs.mathfun
  label = "package of math functions";
  link "math.dll";

  int split(int x "number to split")
    label = "splitter function";

  int cashflow(double amt,
               double rate,
               int periods,
               double * flows / iotype=0)
    label = "cash flow function";

run;
```

PROC PROTO Example

```
proc fcmp libname=sasuser.myfuncs;  
  array flows[20];  
  a = split(32);  
  put a;  
  b = cashflow(1000, .07, 20, flows);  
  put b;  
  put flows
```

Output: Listing

Output from the SPLIT and CASHFLOW Functions

```
                                The SAS System                                1  
  
                                The FCMP Procedure  
  
16  
12  
70 105 128.33333333 145.83333333 159.83333333 171.5 181.5 190.25 198.02777778 205.02777778  
211.39141414 217.22474747 222.60936286 227.60936286 232.27602953 236.65102953 240.76867658  
244.65756547 248.341776 251.841776
```

SAS 9 Update

New Functionalities for SAS Programmers

- User defined functions
- External functions
- **Java Object**
- DATA Step Hash Object
- ODS Graphics Designer
- Enterprise Guide

Using the Java Object

```
data _null_;  
  length return $ 40;  
  declare javaobj myPro ("java.lang.System");  
  declare javaobj myDate ("java.util.Date");  
  
  myPro.callStaticStringMethod("getProperty", "java.version", return);  
  javaVersion = return;  
  putlog "NOTE: Java Version = " return;  
  
  myPro.callStaticStringMethod("getProperty", "user.name", return);  
  userName = return;  
  putlog "NOTE: User Name = " return;  
  
  myPro.delete();  
  myDate.delete();  
run;
```

```
2180  
2181    myPro.delete();  
2182    myDate.delete();  
2183 run;  
  
NOTE: Java Version = 1.5.0_12  
NOTE: User Name = schhrp  
NOTE: DATA statement used (Total process time):  
      real time           0.03 seconds  
      user cpu time       0.00 seconds  
      system cpu time     0.00 seconds  
      Memory              172k  
      OS Memory           14192k  
      Timestamp           9/29/2010  2:26:22 PM
```

SAS 9 Update

New Functionalities for SAS Programmers

- User defined functions
- External functions
- Java Object
- **DATA Step Hash Object**
- ODS Graphics Designer
- Enterprise Guide

The DATA step hash object

- The DATA step hash object is one of the most versatile new features of Base SAS programming.
 - reduced data processing time for complex data join operations like looking up or modifying values from one data set in another data set
 - a fast, easy way to perform in-memory table lookups without sorting or indexing
- With SAS 9.2, we introduce the ability to store duplicate keys in a hash object
 - The hash object can now function as an in-memory index, a sort heap, or even a general-purpose list object.
- The hash object allocates memory as needed. There is no need to pre-size the hash object.

The DATA step hash object in SAS 9.2

- New initialization options
 - duplicate: 'replace' | 'error'
 - multidata: 'y' | 'n'
- More graceful memory management
 - DEFINEDONE(memrc: 'y')
- New methods for handling multiple data values
 - HAS_NEXT / HAS_PREV
 - FIND_NEXT / FIND_PREV

The DATA step hash object with multiple values

```
data visits;
  attrib patient length=8 visit length=8 informat=date. format=date.;
  input patient visit @@;
  datalines;

1 26MAR08 2 26MAR08 1 28MAR08 3 27MAR08 2 27MAR08
2 28MAR08 3 31MAR08 5 31MAR08 1 31MAR08 4 01APR08
;

data _null_;
  length r 8;
  dcl hash h(dataset:'visits', multidata: 'y');
  h.definekey('patient');
  h.definedata('patient', 'visit');
  rc = h.definedone(memrc: 'y');
  if (rc ^= 0) then do;
    h.delete();
    stop;
  end;
  call missing (patient, visit);
  do patient = 1 to 5;
    rc = h.find();
    if (rc = 0) then do;
      put patient= @15 visit= date.;
      h.has_next(result: r);
      do while(r ne 0);
        rc = h.find_next();
        put 'dup ' patient= @15 visit= date.;
        h.has_next(result: r);
      end;
    end;
  end;
end;
run;
```

SAS Log:

```
patient=1      visit=26MAR08
dup patient=1  visit=31MAR08
dup patient=1  visit=28MAR08
patient=2      visit=26MAR08
dup patient=2  visit=28MAR08
dup patient=2  visit=27MAR08
patient=3      visit=27MAR08
dup patient=3  visit=31MAR08
patient=4      visit=01APR08
patient=5      visit=31MAR08
```

SAS 9 Update

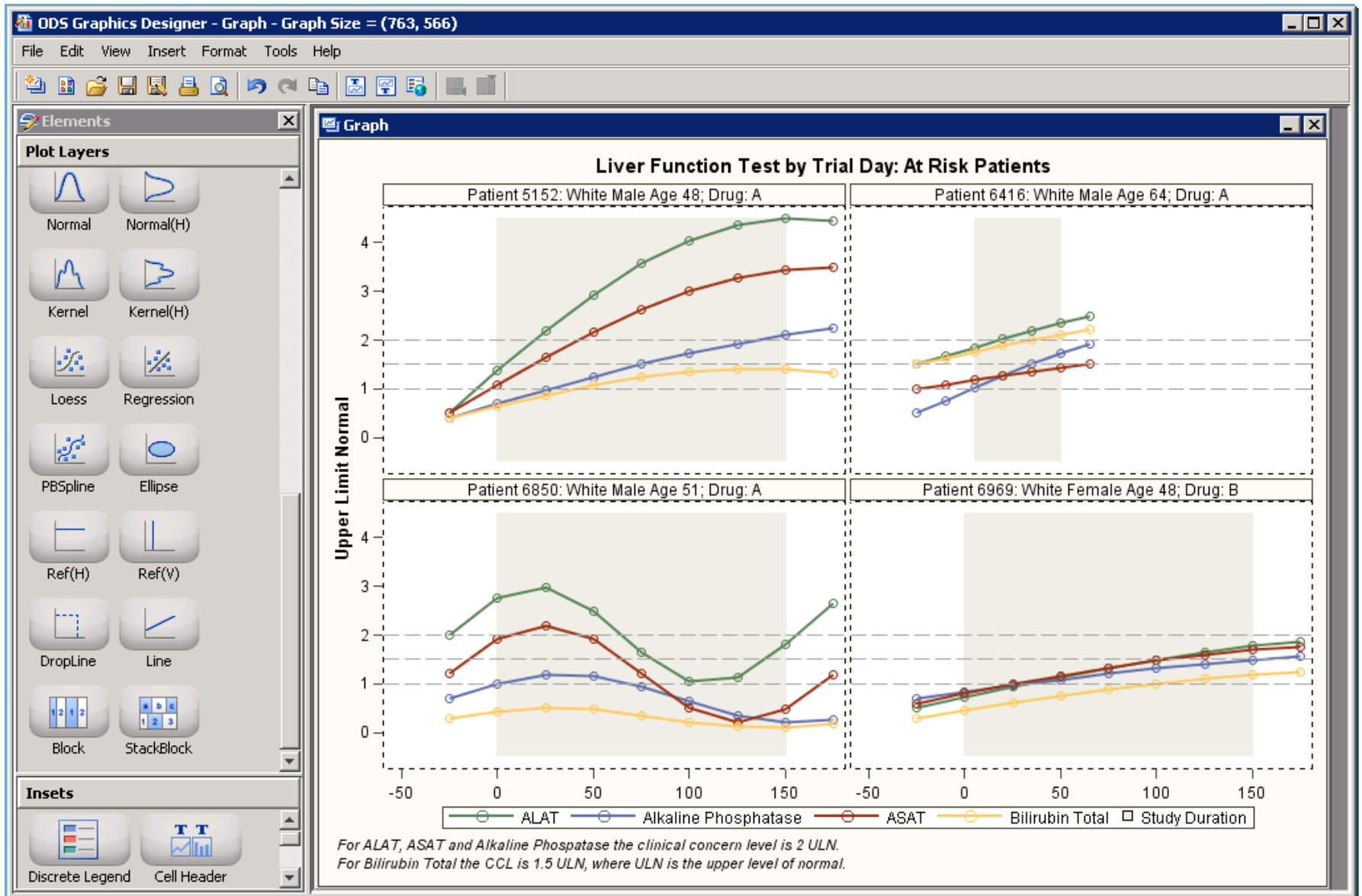
New Functionalities for SAS Programmers

- User defined functions
- External functions
- Java Object
- DATA Step Hash Object
- **ODS Graphics Designer**
- Enterprise Guide

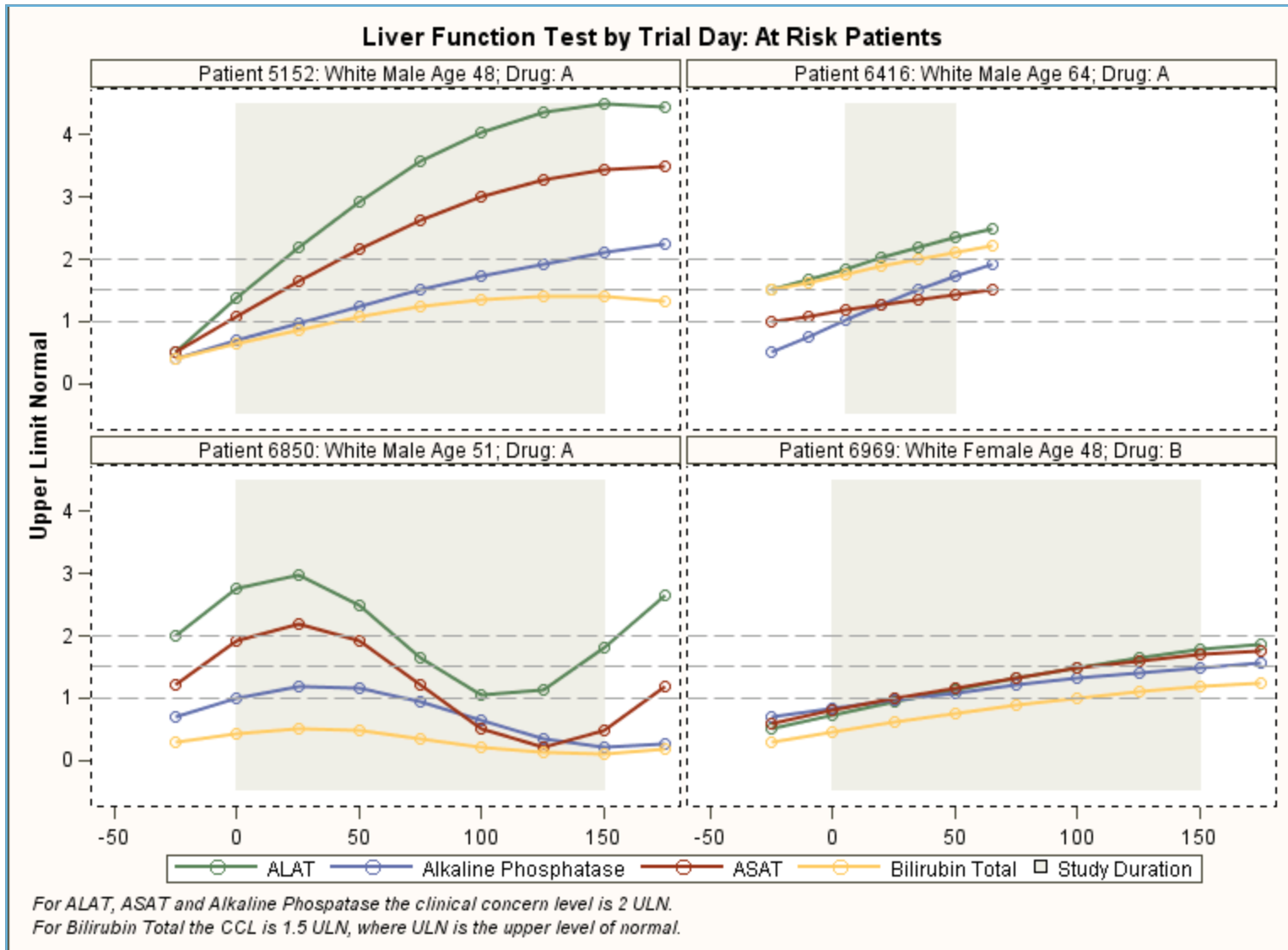
ODS Graphics Designer

- Allows the interactive design of ODS graphics via “drag-and-drop”
- Generates program code, which can be used to create permanent templates with then can be used with the SGRENDER procedure
- Graphics can also be saved as .SGD File und executed in “batch” with PROC SGDESIGN

ODS Graphics Designer



A „simple“ Example



SAS 9 Update

New Functionalities for SAS Programmers

- User defined functions
- External functions
- Java Object
- DATA Step Hash Object
- ODS Graphics Designer
- **Enterprise Guide**

SAS Enterprise Guide 4.3

Programming with Tool-Tips and „auto complete“

- Best support for
 - Programmer
 - Application Developer

The diagram illustrates the SAS programming environment, showing how it supports both programmers and application developers. It features several overlapping windows and a detailed tooltip.

Code Snippets:

```
/* The G3D procedure plots the data in */  
/* the shape of a cowboy hat */  
proc g3d data=hat;  
    plot y*x=z;  
run;
```

Procedure Lists:

- PROC List:** PROC, QUIT, RDISPLAY, RESETLINE, RGET, RSUBMIT, RUN, SASFILE, SIGNOFF.
- PROC ME List:** MEANS, METADATA, MI, MIANALYZE, MIGRATE, MIXED, MODECLUS, MODEL, MULTTEST.
- PROC MEANS DATA List:** ALPHA=, CHARTYPE, CLASSDATA=, COMPLETETYPES, CSS, CV, DATA=, DESCEND, DESCENDING.

PROC MEANS DATA Tooltip:

Keyword: MEANS
Kontext: [Prozedurdefinition] PROC MEANS

Syntax: PROC MEANS <option(s)> <statistic-keyword(s)>;
BY <DESCENDING> variable-1 <...> <DESCENDING> variable-n <NOTSORTED>;
CLASS variable(s) </ option(s)>;
FREQ variable;
ID variable(s);
OUTPUT <OUT=SAS-data-set> <output-statistic-specification(s)> <id-group-specification(s)>
 <maximum-id-specification(s)>
 <minimum-id-specification(s)> </ option(s)>;
TYPES request(s);
VAR variable(s) </ WEIGHT=weight-variable>;
WAYS list;
WEIGHT variable;

The MEANS procedure provides data summarization tools to compute descriptive statistics for variables across all observations and within groups of observations. For example, PROC MEANS

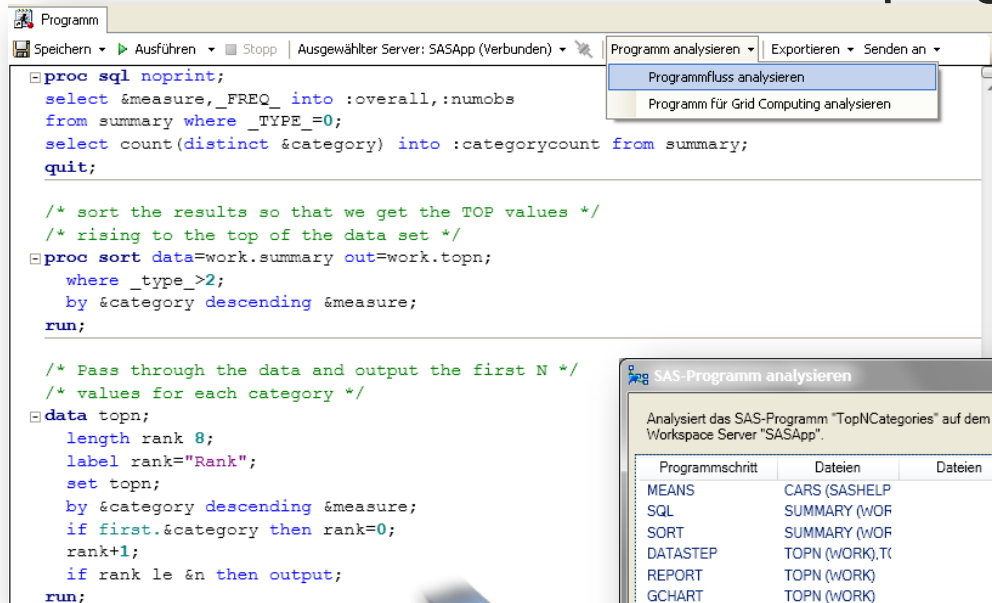
- o calculates descriptive statistics based on moments
- o estimates quantiles, which includes the median
- o calculates confidence limits for the mean
- o identifies extreme values
- o performs a t test.

By default, PROC MEANS displays output. You can also use the OUTPUT statement to store the statistics in a SAS data set. PROC MEANS and PROC SUMMARY are very similar.

SAS Enterprise Guide 4.3

Analyzing programs with *code analyzer*

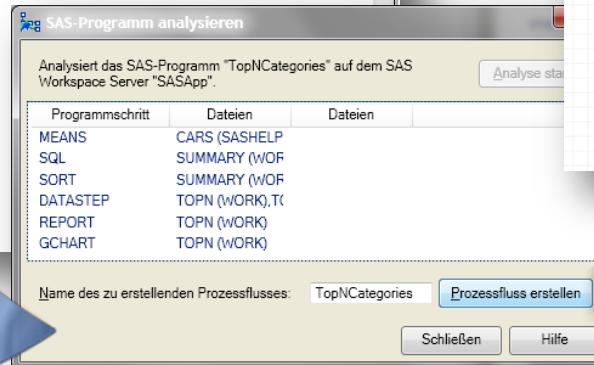
- Visualizes SAS programs and transforms them to an Enterprise Guide process flow
- Enables you to have a quick overview and additional documentation for SAS programs



```
proc sql noprint;
select &measure, _FREQ_ into :overall, :numobs
from summary where _TYPE_ =0;
select count(distinct &category) into :categorycount from summary;
quit;

/* sort the results so that we get the TOP values */
/* rising to the top of the data set */
proc sort data=work.summary out=work.topn;
  where _type_ >2;
  by &category descending &measure;
run;

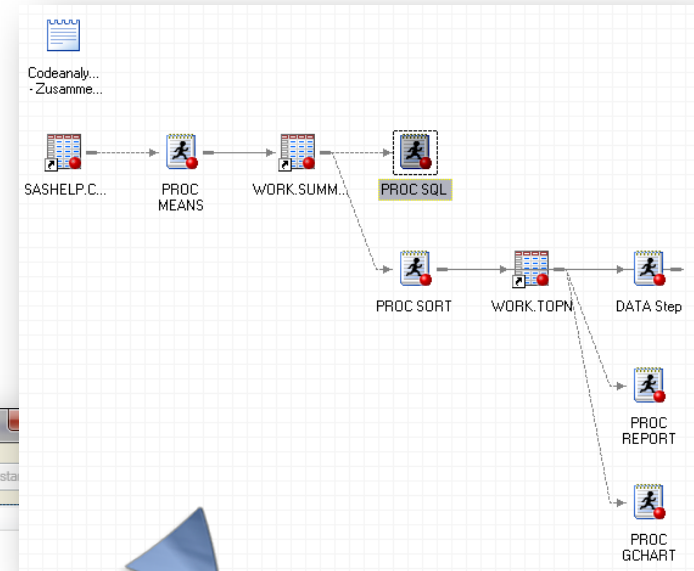
/* Pass through the data and output the first N */
/* values for each category */
data topn;
  length rank 8;
  label rank="Rank";
  set topn;
  by &category descending &measure;
  if first.&category then rank=0;
  rank+1;
  if rank le &n then output;
run;
```



Programmschritt	Dateien	Dateien
MEANS	CARS (SASHELP	
SQL	SUMMARY (WOF	
SORT	SUMMARY (WOF	
DATASEP	TOPN (WORK),T	
REPORT	TOPN (WORK)	
GCHART	TOPN (WORK)	

Name des zu erstellenden Prozessflusses: TopNCategories Prozessfluss erstellen

Schließen Hilfe



SAS 9.3 – Outlook

SAS Foundation (BASE, ODS,)

- ODS Graphics will be part of Base SAS
 - HTML is the new default Destination for Windows and UNIX
 - New default style for HTML (HTMLBlue)
 - Default for ODS Graphics will be on
 - ODS Graphics Designer & Editor will be shipped with Base SAS

Data Types

- Cascading Style Sheets
- Entirely New Style Engine
- Context Base Style Selection
- Attribute Selectors
- Multiple Style Elements
- Media Types

Name	Sex	Age	Height	Weight
Alfred	M	14	69	112.5
Alice	F	13	56.5	84
Barbara	F	13	65.3	98
Carol	F	14	62.8	102.5
Henry	M	14	63.5	102.5
James	M	12	57.3	83
Jane	F	12	59.8	84.5
Janet	F	15	62.5	112.5
Jeffrey	M	13	62.5	84

```
.table td[type=num] {  
    background-color: silver;  
}
```

```
.table td[type=char] {  
    color: purple;  
    font-weight: bold;  
}
```

```
ods pdf cssstyle='mystyle.css';
```

SGF11 - [Unveiling the Power of Cascading Style Sheets \(CSS\) in ODS](#)
- [Introduction to ODS Graphics for the Non-Statistician](#)



Q & A