

SAS: An Introduction

- 1 Introduction
- 2 Starting SAS
- 3 Creating Data Manually in SAS
- 4 Reading an External Data File and Saving in SAS Studio
- 5 Manipulating Datasets
- 6 do Loop

- 1 Introduction
- 2 Starting SAS
- 3 Creating Data Manually in SAS
- 4 Reading an External Data File and Saving in SAS Studio
- 5 Manipulating Datasets
- 6 do Loop

What is SAS

- It is a Statistical Analysis System, a programming language composed of statements that specify how data to be processed and analyzed.
- A SAS program consists of a sequence of SAS statements grouped together into blocks, referred to as steps.
- SAS has **TWO** major steps:
 - Data steps: any portion of a SAS program that begins with a DATA statement and ends with a RUN statement is called a DATA Step.
 - PROC Steps (Procedures): Any portion of a SAS program that begins with a PROC statement and ends with a RUN statement is called a PROC Step or Procedures.
- PROC steps are in-built programs that allow us to analyze the data contained in a SAS data set. PROC steps are used to calculate descriptive statistics, to generate summary reports, and to create summary graphs and charts.

- 1 Introduction
- 2 Starting SAS
- 3 Creating Data Manually in SAS
- 4 Reading an External Data File and Saving in SAS Studio
- 5 Manipulating Datasets
- 6 do Loop

Registering SAS OnDemand Account

- Purchasing SAS is expensive, however, there is a free version.
- There are few steps as instructed here: https://www.sas.com/en_in/software/on-demand-for-academics/references/getting-started-with-sas-on-demand-for-academics-studio.html.

Scroll down and click on the link given:

Instructions for Getting Started

Step 1

Create and verify a SAS profile if you don't have one. If you do have one, just log in.



Step 2

Register for SAS OnDemand for Academics using your SAS profile credentials.

Step 3

Once you receive a confirmation email, click the link to go to SAS OnDemand for Academics.

Step 4

Log in to access SAS Studio. Educators can register a course and share information with students.

You will be directed to:

<https://www.sas.com/profile/ui/#/create>

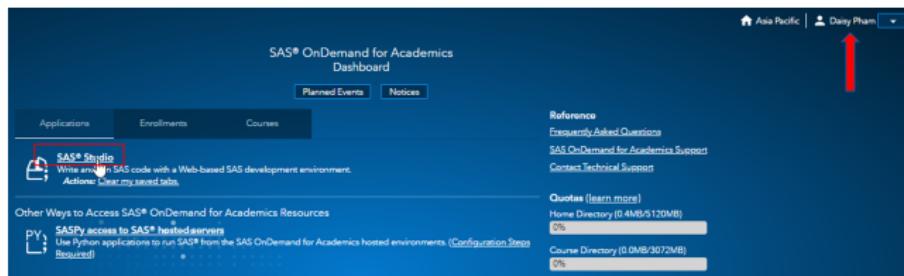
to create your own profile.

SAS Starting

After creating your own profile, you can follow the steps given. An email will be sent to your registered email which gives you the link

<https://welcome.oda.sas.com>

Open the link, you will have

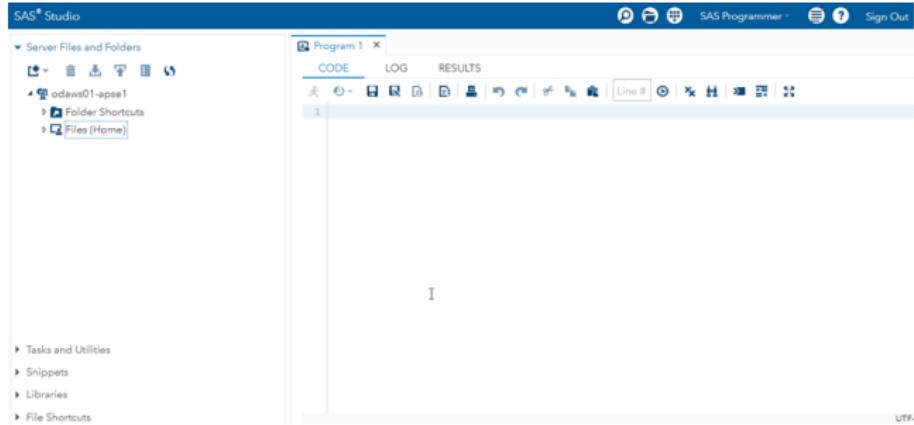


The screenshot shows the SAS OnDemand for Academics Dashboard. At the top right, there is a user profile for 'Daisy Pham' with a red arrow pointing to it. Below the header, there are tabs for 'Planned Events' and 'Notices'. The main content area has three sections: 'SAS® Studio' (highlighted with a red box), 'Other Ways to Access SAS® OnDemand for Academics Resources' (including 'SASPy access to SAS® hosted servers'), and 'Quotas' (showing Home Directory at 0 MB/5120 MB and Course Directory at 0 MB/3072 MB). A red arrow also points to the 'SAS® Studio' link.

Click "SAS Studio".

SAS Studio

You should see the window where we can start to write the code, or upload files.



1 Introduction

2 Starting SAS

3 Creating Data Manually in SAS

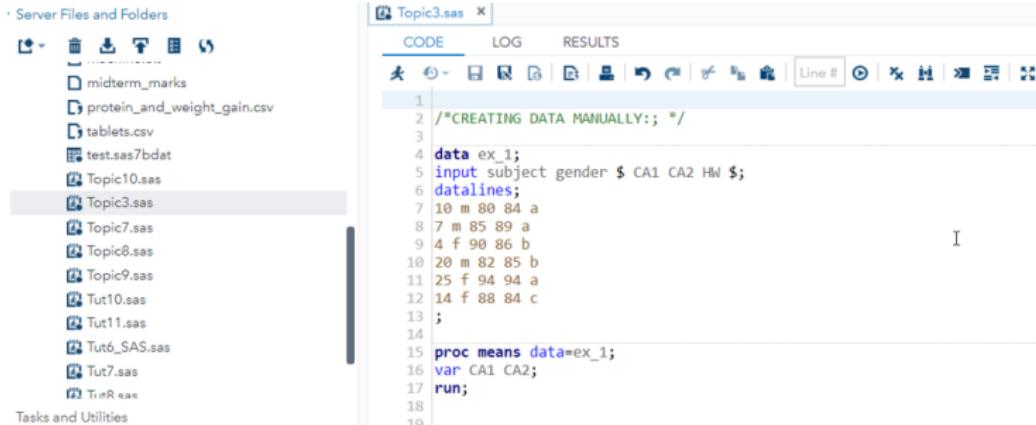
4 Reading an External Data File and Saving in SAS Studio

5 Manipulating Datasets

6 do Loop

An Easy Example (1)

Few lines of code to create a dataset (manually) and derive simple descriptive statistics.



The screenshot shows the SAS Studio interface. On the left, there is a file tree titled 'Server Files and Folders' containing various SAS and CSV files. A file named 'Topic3.sas' is selected and highlighted with a grey background. To the right of the file tree is a code editor window titled 'Topic3.sas'. The window has three tabs: 'CODE', 'LOG', and 'RESULTS'. The 'CODE' tab is active, displaying the following SAS code:

```
1 /*CREATING DATA MANUALLY: */
2
3
4 data ex_1;
5 input subject gender $ CA1 CA2 HW $;
6 datalines;
7 10 m 80 84 a
8 7 m 85 89 a
9 4 f 98 86 b
10 20 m 82 85 b
11 25 f 94 94 a
12 14 f 88 84 c
13 ;
14
15 proc means data=ex_1;
16 var CA1 CA2;
17 run;
```

Click the “running man” to run the code.

An Easy Example (2)

The output for data:

CODE LOG RESULTS OUTPUT DATA

Table: WORK.EX_1 | View: Column names | Filter: (none)

Columns Total rows: 6 Total columns: 5 Rows 1-6

	subject	gender	CA1	CA2	HW
1	10	m	80	84	a
2	7	m	85	89	a
3	4	f	90	86	b
4	20	m	82	85	b
5	25	f	94	94	a
6	14	f	88	84	c

An Easy Example (3)

The results of running the code:

The screenshot shows the SAS Studio interface with the following details:

- Title Bar:** Topic3.sas
- Tab Bar:** CODE, LOG, RESULTS (selected)
- Toolbar:** Includes icons for Save, Print, Run, Download, Help, Find, and Refresh.
- Table of Contents:** A link to the Table of Contents.
- Output Area:** Displays the results of the "The MEANS Procedure".

Variable	N	Mean	Std Dev	Minimum	Maximum
CA1	6	86.5000000	5.2057660	80.0000000	94.0000000
CA2	6	87.0000000	3.8987177	84.0000000	94.0000000

Some Basic SAS Program Rules (1)

Rules for SAS statements:

- All SAS statements (except those containing data) must end with a semicolon (;).
- SAS statements typically begin with a SAS keyword. (DATA, PROC).
- SAS statements are not case sensitive, that is, they can be entered in lowercase, uppercase, or a mixture of the two.
Example : SAS keywords (DATA, PROC) are not case sensitive
- A delimited comment begins with a forward slash-asterisk /*) and ends with an asterisk-forward slash (*/). All text within the delimiters is ignored by SAS.

Some Basic SAS Program Rules (2)

Rules for SAS names:

- All names must contain between 1 and 32 characters.
- The first character appearing in a name must be a letter (A, B, ...Z, a, b, ...z) or an underscore (_). Subsequent characters must be letters, numbers, or underscores. That is, no other characters, such as \$, %, or & are permitted. Blanks also cannot appear in SAS names.
- SAS names are not case sensitive, that is, they can be entered in lowercase, uppercase, or a mixture of the two. (SAS is only case sensitive within quotation marks.)

Rules for SAS variables:

- If the variable in the INPUT statement is followed by a dollar sign (\$), SAS assumes this is a character variable. Otherwise, the variable is considered as a numeric variable.

General Form of a DATA Step

```
data data_set_name;  
  input variables;  
  datalines;  
  the lines of data  
;  
run;
```

Input Format (Free Format)

For the **free format**: the code as in the photo in slide 8 is an example.

- The “\$” sign implies that the variable right before it is a character variable.
- Can use “.” for a missing value.

Input Format (Fixed Format)

- For manually creating a **fixed format** data, the code below is an example.

```
data ex_fix;  
input subject 1-2 gender $ 3 CA1 5-6 CA2 8-9 HW $ 10;  
datalines;  
10m 80 84a  
7 m 85 89a  
14f 88 84c  
:
```

- In the chunk of code above, subject's value take the first two positions/columns; gender's value is at the third position/column and it is a categorical variable. The 4th position/column is a space,....

- 1 Introduction
- 2 Starting SAS
- 3 Creating Data Manually in SAS
- 4 Reading an External Data File and Saving in SAS Studio
- 5 Manipulating Datasets
- 6 do Loop

Reading Data Files (1)

Most of us are using SAS studio, watching the video in the link below to know how to import an external data file to SAS studio.

[https://video.sas.com/detail/video/4664358166001/
using-the-import-data-utility-in-sas-studio](https://video.sas.com/detail/video/4664358166001/using-the-import-data-utility-in-sas-studio)

Reading Data Files (2)

The screenshot shows the SAS Studio interface. On the left, the 'Server Files and Folders' tree view displays several files and folders, including 'heats.csv' which is highlighted with a blue selection bar. A blue arrow points from this selection bar to the 'CODE' tab in the main window. The main window contains the SAS code for 'Topic7.sas'. The code reads a CSV file named 'heats.csv' and performs a PROC MEANS analysis on it.

```
/* CHANGING VARIABLE NAME */
data ex_1;
  set ex_1(rename= (var1=id var2=gender var3 = CA1 var4 = CA2 var5 = Hw));
run;

proc means data=ex_1;
var CA1 CA2;
run;
```

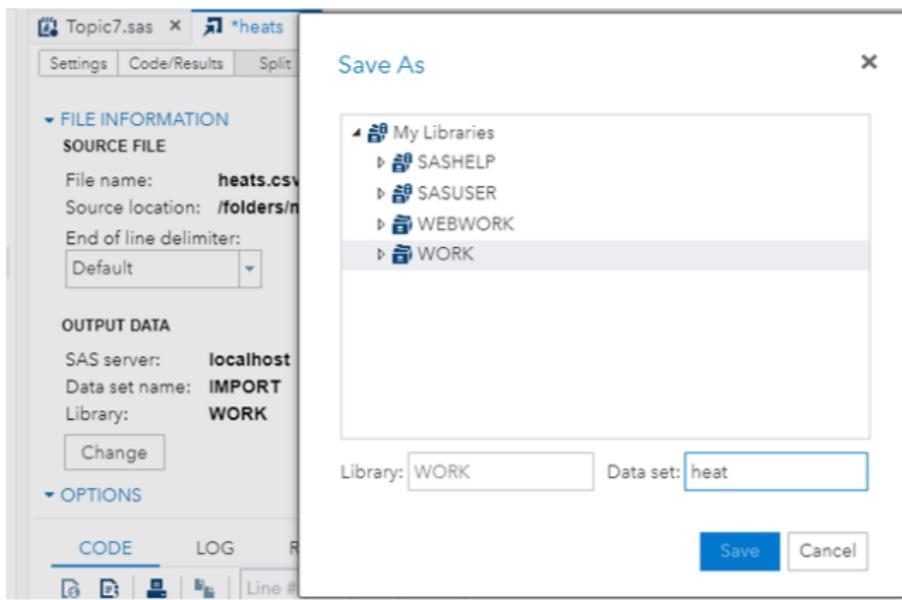
Drag the file

/folders/myfolders/Topic7.sas

Drag the file that you want to open to the code window.

Reading Data Files (3)

- After the data file that you want to open is dragged and dropped into the code window, then a new window (which has the name as the name of the file) will open, it has the code for how to import this data file. You can import the file from this window or copy the code to open the file.



Reading Data Files (4)

- The SAS data set will be created under the name “heat” in the Work directory.
- The SAS data sets in the Work directory are temporary and will be lost once we logoff the SAS studio session.

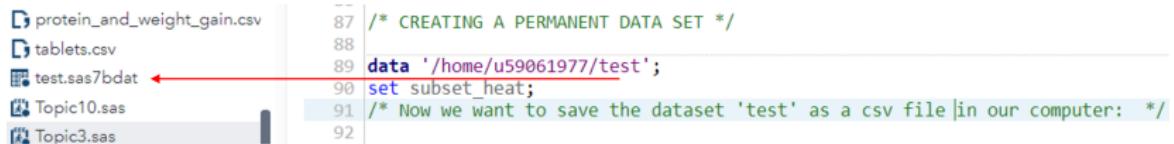
```
FILENAME REFFILE '/home/u59061977/heats.csv';
PROC IMPORT DATAFILE=REFFILE
  DBMS=CSV
  OUT=WORK.heat;
  GETNAMES=YES; /* this will set the name for the variable/columns as the same as in
RUN;

PROC CONTENTS DATA=WORK.heat; RUN;

proc means data = heat;
var heat;
run;
```

Saving Data

To create a permanent dataset:

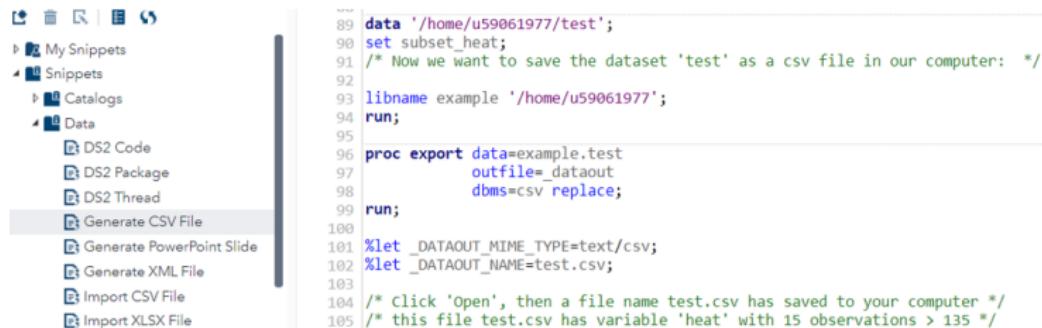


```
protein_and_weight_gain.csv  
tablets.csv  
test.sas7bdat ←  
Topic10.sas  
Topic3.sas
```

```
/* CREATING A PERMANENT DATA SET */  
87 data '/home/u59061977/test';  
88 set subset_heat;  
89 /* Now we want to save the dataset 'test' as a csv file |in our computer: */  
90  
91  
92
```

Saving Data as a CSV File

After creating the permanent dataset named 'test' in SAS, now we want to save it as a csv file in our computer: (1) click "Generate CSV File" and (2) make necessary changes accordingly, then (3) run.



```
89 data '/home/u59061977/test';
90 set subset_heat;
91 /* Now we want to save the dataset 'test' as a csv file in our computer: */
92
93 libname example '/home/u59061977';
94 run;
95
96 proc export data=example.test
97         outfile=_dataout
98         dbms=csv replace;
99 run;
100
101 %let _DATAOUT_MIME_TYPE=text/csv;
102 %let _DATAOUT_NAME=test.csv;
103
104 /* click 'Open', then a file name test.csv has saved to your computer */
105 /* this file test.csv has variable 'heat' with 15 observations > 135 */
```

You can read from the link below:

<https://communities.sas.com/t5/SAS-Communities-Library/How-to-Export-SAS-Datasets-as-a-CSV-File-in-SAS-University/ta-p/441569>

- 1 Introduction
- 2 Starting SAS
- 3 Creating Data Manually in SAS
- 4 Reading an External Data File and Saving in SAS Studio
- 5 Manipulating Datasets
- 6 do Loop

Keeping a Subset of Variables (1)

- Consider data set “ex_1_comma.txt” which has 6 observations, 5 variables of subjects, gender, CA1, CA2 and HW but there is no name given in this data set.
- Reading data set “ex_1_comma.txt” into SAS and set the name for this dataset as “example1” and name these variables as: id, gender, CA1, CA2 and HW.

```
* IMPORTING DATA FROM A TEXT FILE:;

FILENAME REFFILE '/home/u59061977/ex_1_comma.txt';

PROC IMPORT DATAFILE=REFFILE
  DBMS=DLM
  OUT=WORK.example1;
  DELIMITER=",";
  GETNAMES=NO; /* the given dataset doesnot have any variable name */
  DATAROW=1;
RUN;

/* CHANGING VARIABLE NAMES */

data example1;
  set example1(rename= (var1=id var2=gender var3 = CA1 var4 = CA2 var5 = Hw));
run;
```

Keeping a Subset of Variables (2)

- ▶ Server Files and Folders
- ▶ Tasks and Utilities
- ▶ Snippets
- ▼ Libraries
 - MAPS
 - MAPSGFK
 - MAPSSAS
 - SASDATA
 - SASHelp
 - SASUSER
 - STPSAMP
 - WC000001
 - WEBWORK
 - WORK
 - EX_1
 - EXAMPLE1

Topic3.sas x

CODE LOG RESULTS OUTPUT DATA

Table: WORK.EXAMPLE1 View: Column names

Columns Total rows: 6 Total columns: 5

		id	gender	CA1	CA2	Hw
<input checked="" type="checkbox"/>	Select all	1	10	M	80	84 A
<input checked="" type="checkbox"/>	(2) id	2	7	M	85	89 A
<input checked="" type="checkbox"/>	(3) gender	3	4	F	90	86 B
<input checked="" type="checkbox"/>	(4) CA1	4	20	M	82	85 B
<input checked="" type="checkbox"/>	(5) CA2	5	25	F	94	94 A
<input checked="" type="checkbox"/>	(6) Hw	6	14	F	88	84 C

Property Value

Label

Now, from this data set “example1”, we want to keep 4 variables, and discard gender.

Keeping a Subset of Variables (3)

We use the code:

```
data example1_nogender;  
set example1;  
keep id CA1 CA2 HW;  
run;
```

Then a new dataset “example1_nogender” will be created in Work directory which has 4 variables except ‘gender’.

The screenshot shows the SAS Studio interface. On the left, the Libraries panel is open, displaying 'My Libraries' with entries for 'SASHELP', 'SASUSER', 'WEBWORK', and 'WORK'. Under 'WORK', there are entries for 'EXAMPLE1' and 'EXAMPLE1_NOGENDER'. On the right, the main workspace shows the 'OUTPUT DATA' tab selected. The 'Table:' dropdown shows 'WORK.EXAMPLE1_NOGENDER'. The 'Columns' section lists the variables 'id', 'CA1', 'CA2', and 'Hw'. The 'RESULTS' tab shows a table with 6 rows of data:

	id	CA1	CA2	Hw
1	10	80	84	A
2	7	85	89	A
3	4	90	86	B
4	20	82	85	B
5	25	94	94	A
6	14	88	84	C

Dropping a Subset of Variables

Suppose we want to drop the variable 'id' from the data set.

```
data example1_drop_id;  
set example1;  
drop id;  
run;
```

Then a new dataset “example1_drop_id” will be created in Work directory which has 4 variables except ‘id’.

Concatenating Datasets

Suppose we have a dataset for male (e.g. example1_m and a dataset for female (e.g. example1_f) from the original dataset “example1”. We need to combine the two datasets (by rows/observations) to do analysis, then we use the code:

```
data example1_concat;  
set example1_m example1_f;  
run;
```

Match Merging Datasets

- A dataset “example1_nogender” that consists of 4 variables: id, CA1, CA2 and HW.
- Another dataset “example1_drop_id” that consists of 4 variables: gender, CA1, CA2 and HW. The values of variables in this dataset come from the same subjects in “example1_nogender”, respectively.
- We want to merge these two datasets into one.

Sort “example1_nogender” by a common variable, ‘CA1’:

```
proc sort data = example1_nogender;  
by CA1;
```

Sort “example1_drop_id” by the common variable ‘CA1’:

```
proc sort data = example1_drop_id;  
by CA1;
```

Merging these two datasets, the new dataset is named as “example1_merge”.

```
data example1_merge;  
merge example1_drop_id example1_nogender;  
by CA1;  
run;
```

Transforming Data and Creating a New Variable

- A new variable 'ave_mark' is created, which is the average mark of CA1 and CA2.
- Another variable 'final_grade' is defined as: if 'ave_mark' is at least 90 and HW is grade A then final grade is A; if 'ave_mark' is at least 85 and HW is grade A or B the final grade is B; otherwise the final grade is C.

```
*      TRANSFORMING DATA:      ;  
  
data example1_transf;      ← New dataset  
set example1;              ← Original dataset  
ave_mark= (CA1 + CA2)/2;  ← New variable  
if HW = "." then final_grade= "";  
else if ave_mark >= 90 and HW = "A" then final_grade = "A";  
else if ave_mark >= 85 and (HW = "A" or HW= "B") then final_grade= "B";  
else final_grade= "C";  
run;
```

Give Rank to a Sorted Data

- Dataset “example1_transf” has variable ‘ave_mark’.
- We want to give rank to students based on this variable ‘ave_mark’.

```
proc sort data = example1_transf;
by descending ave_mark;
data example1_rank;
set example1_transf;
rank = _n_;
run;
```

Total rows: 6 Total columns: 8

Rows 1-6

	id	gender	CA1	CA2	Hw	ave_mark	final_grade	rank
	25	F	94	94	A	94	A	1
	4	F	90	86	B	88	B	2
	7	M	85	89	A	87	B	3
	14	F	88	84	C	86	C	4
	20	M	82	85	B	83.5	C	5
	10	M	80	84	A	82	C	6

- 1 Introduction
- 2 Starting SAS
- 3 Creating Data Manually in SAS
- 4 Reading an External Data File and Saving in SAS Studio
- 5 Manipulating Datasets
- 6 do Loop

The General Form of a do Loop

```
do <index variable> = <beginning number> to <ending number>;  
statements;  
end;
```

An example:

- There are 15 observations, where measurements 'y' is a scale; first factor (categorical variable) has 3 outcomes, denoted as 1,2 and 3;
- Second factor has 5 outcomes denoted as 1,2,3,4 and 5.

An Example of do Loop (1)

```
data exdo;
do factora = 1 to 3;           ←
  do factorb = 1 to 5;          ←
    input y @ @;              ←
    output;
  end;
end;
datalines;
22 24 16 18 19
15 21 26 16 25
14 28 21 19 24
run;
```

First variable with 3 outcomes

Second variable with 5 outcomes

3rd variable, needs to key in values

An Example of do Loop (2)

The output is

Total rows: 15 Total columns: 3

	factora	factorb	y
1	1	1	22
2	1	2	24
3	1	3	16
4	1	4	18
5	1	5	19
6	2	1	15
7	2	2	21
8	2	3	26
9	2	4	16
10	2	5	25
11	3	1	14
12	3	2	28
13	3	3	21
14	3	4	19
15	3	5	24

Functions Defined by Users

Creating and calling functions in SAS is not as straight forward as in R or Python. You can read about functions defined by user in the link given below:

<https://documentation.sas.com/?docsetId=proc&docsetTarget=n0pio2crlptr35n1ny010zrfbvc9.htm&docsetVersion=9.4&locale=en>

After opening the link and reading the overview, you can read the continued parts (in the left panel), like concepts, usage, examples, etc.