Modernizing Analytics: A Head-to-Head Battle Between BASE SAS and CASL in SAS Viya

Mayur Jadhav

Posten Bring AS

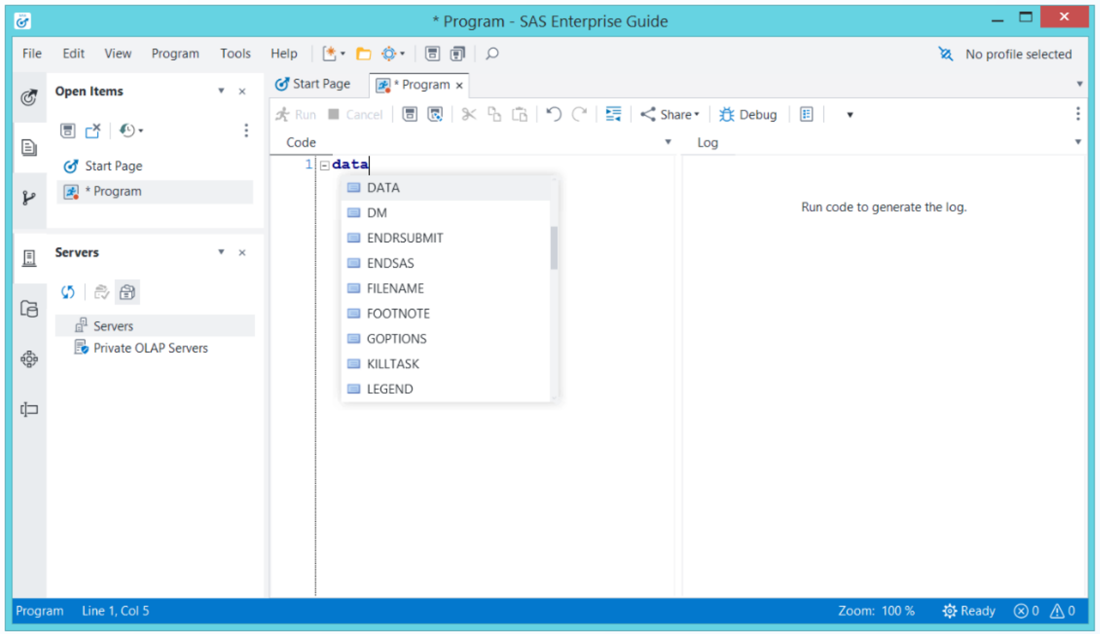
Senior Advisor, BI Developer

Executive MBA 2025

Date: 18.09.2024

BASE SAS:

This is the foundational language used for data manipulation, statistical analysis, and reporting within the traditional SAS environment. It’s a mature, robust tool that many of us are familiar with.

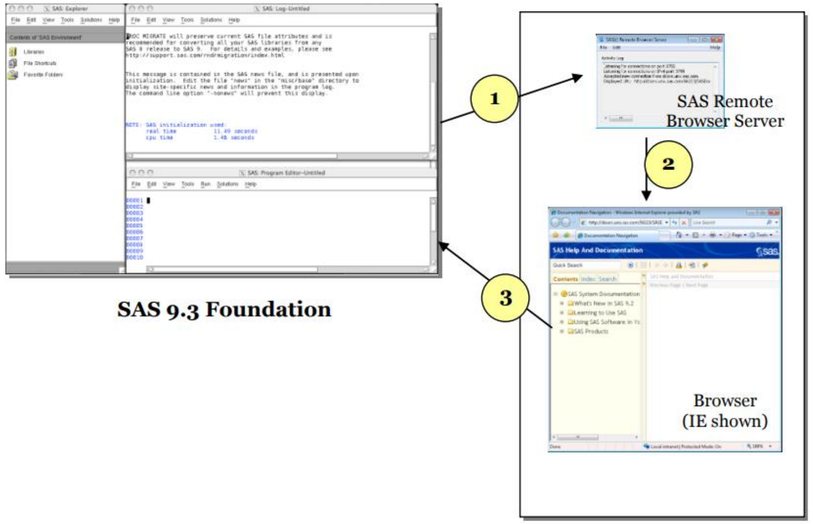


```
data
```

Run code to generate the log.

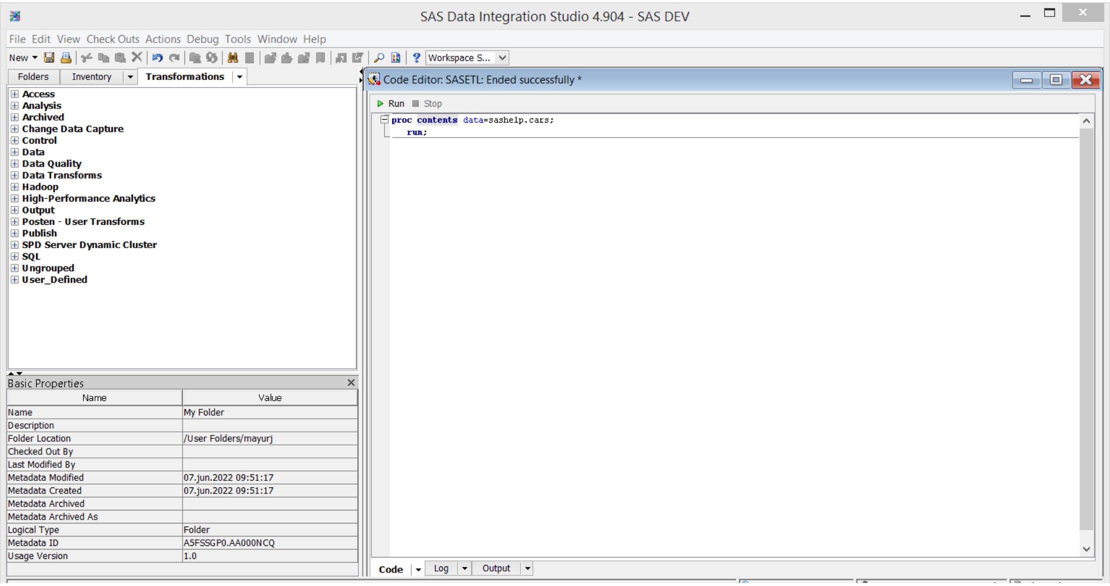
Program   Line 1, Col 5   Zoom: 100 %   Ready   0   0

System Requirements for SAS 9.3 Foundation for Linux for x64



SAS 9.3 Foundation

Desktop Computer (Linux/Windows 32bit)



Code Editor: SASETL: Ended successfully *

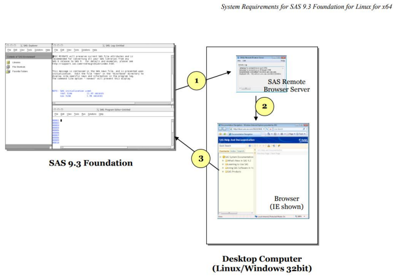
```
proc contents data=sashelp.cars;
   run;
```

Basic Properties

Name	Value
Name	My Folder
Description	
Folder Location	/User Folders/mayurj
Checked Out By	
Last Modified By	
Metadata Modified	07.jun.2022 09:51:17
Metadata Created	07.jun.2022 09:51:17
Metadata Archived	
Metadata Archived As	
Logical Type	Folder
Metadata ID	A5FSSGP0.AA000NCQ
Usage Version	1.0

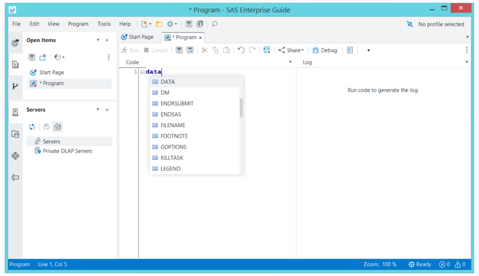
BASE SAS:

This is the foundational language used for data manipulation, statistical analysis, and reporting within the traditional SAS environment. It's a mature, robust tool that many of us are familiar with.



System Requirements for SAS 9.3 Foundation for Linux for x64

1
SAS Remote Browser Server
2
3
Browser (IE shown)
SAS 9.3 Foundation
Desktop Computer (Linux/Windows 32bit)

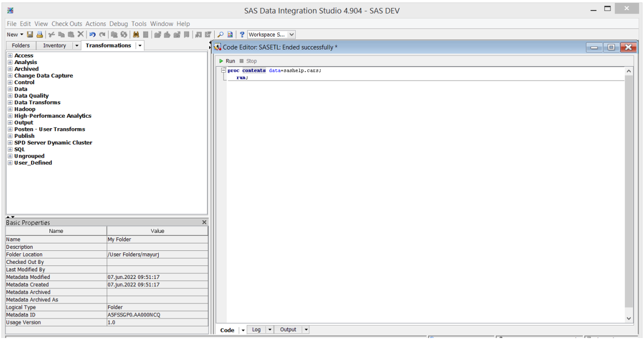


* Program - SAS Enterprise Guide

```
data
```

Run code to generate the log.

Program Line 1, Col 5 Zoom: 100 % Ready 0 0



SAS Data Integration Studio 4.904 - SAS DEV

Code Editor: SASETL: Ended successfully *

```
proc contents data=sashelp.cars;
   run;
```

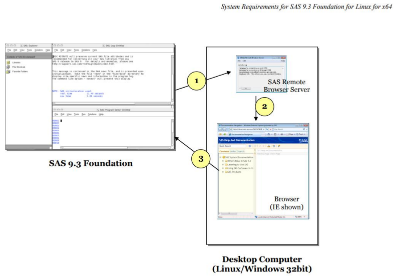
Basic Properties

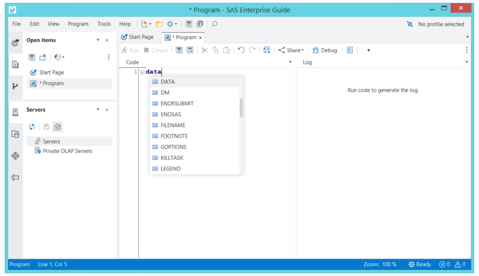
Name	Value
Name	My Folder
Description	
Folder Location	/User Folders/mayurj
Checked Out By	
Last Modified By	
Metadata Modified	07.jun.2022 09:51:17
Metadata Created	07.jun.2022 09:51:17
Metadata Archived	
Metadata Archived As	
Logical Type	Folder
Metadata ID	A5FSSGP0.AA000NCQ
Usage Version	1.0

CASL:

BASE SAS:

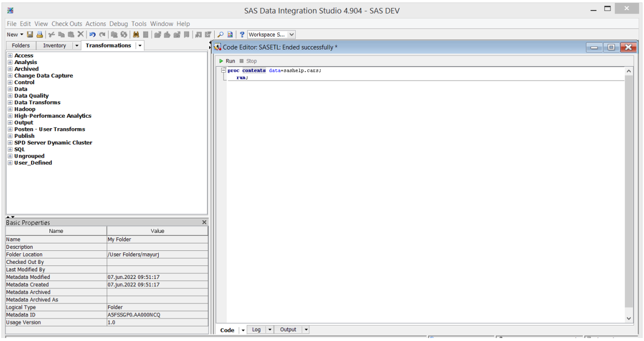
This is the foundational language used for data manipulation, statistical analysis, and reporting within the traditional SAS environment. It’s a mature, robust tool that many of us are familiar with.


System Requirements for SAS 9.3 Foundation for Linux for x64
SAS 9.3 Foundation
1
2
3
SAS Remote Browser Server
Browser (IE shown)
Desktop Computer (Linux/Windows 32bit)



```
data
```

Run code to generate the log.



Name	Value
Name	My Folder
Description	
Folder Location	/User Folders/mayurj
Checked Out By	
Last Modified By	
Metadata Modified	07.jun.2022 09:51:17
Metadata Created	07.jun.2022 09:51:17
Metadata Archived	
Metadata Archived As	
Logical Type	Folder
Metadata ID	A5FSSGP0.AA000NCQ
Usage Version	1.0

```
proc contents data=sashelp.cars;
   run;
```

CASL:

It's Like Base SAS With

The Macro Built-in.

Source

CASL (Cloud Analytic Services Language)

CASL is a powerful language for running actions in CAS.

CASL is designed to leverage the distributed computing power of SAS Viya, offering enhanced performance and scalability for modern analytics tasks.

1. statement-based language
2. case insensitive language
3. scripting language with the following strengths
   - running actions
   - working with results
   - developing analytic pipelines
   - running code in CAS with user-defined actions

CASL (Cloud Analytic Services Language)

CASL is designed to leverage the distributed computing power of SAS Viya, offering enhanced performance and scalability for modern analytics tasks.

Characteristics of CASL

- **Statement-Based**: CASL uses clear, action-oriented statements.
- **Case Insensitive**: Keywords and identifiers are case insensitive.
- **Scripting Strengths**: It excels at running actions, managing results, and building pipelines.
- **Semicolon Termination**: Statements end with a semicolon (;).
- **Flexible PROC CAS**: Multiple CASL programs can be included in one PROC CAS step.

Use Cases of CASL

- **Pipeline Development**: Build and refine complex analytic pipelines.
- **Result Manipulation**: Analyze and adjust results from actions.
- **Action Arguments**: Create precise arguments for actions.
- **Custom Actions**: Develop unique actions and functions.

CASL Programmer’s Guide

BASE SAS vs CASL

Comparative Analysis

Language Structure:	BASE SAS: Combines DATA step with SAS Procedures. CASL: Statement-based scripting language that is case insensitive and executes CAS actions.
Processing Engine:	BASE SAS: Runs on traditional SAS server. CASL: Interacts with SAS Cloud Analytic Services (CAS), enabling distributed computing.
Procedure Execution:	BASE SAS: Uses PROC statements directly. CASL: Uses CAS actions instead of procedures, though these actions often correspond to CAS-enabled PROCs.
Language Integration:	BASE SAS: Primarily SAS-centric. CASL: Can be accessed via multiple interfaces including SAS, Python, R, Java, and REST APIs
Performance:	BASE SAS: Efficient for traditional data processing tasks. CASL: Optimized for high-performance analytics, especially with large datasets.

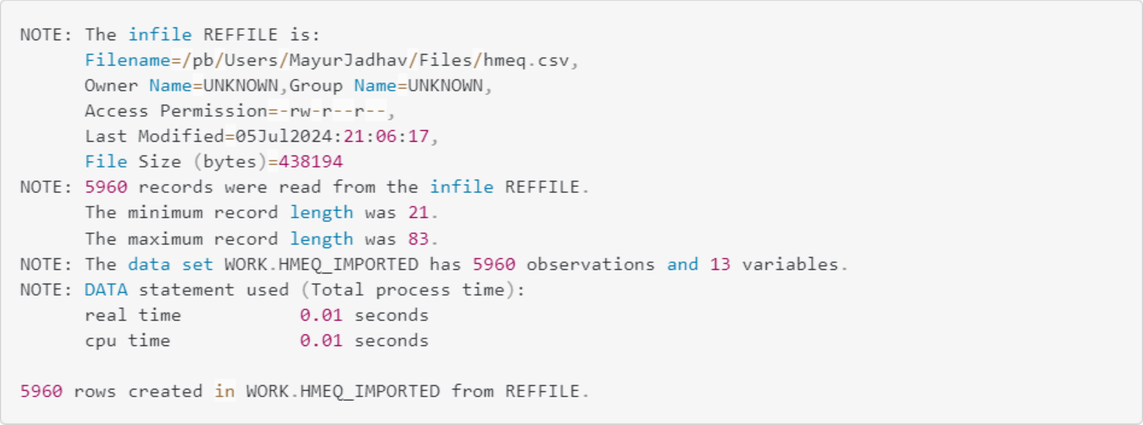
Base SAS vs CASL

Code Comparison for common tasks

Common ground for this battle: **BASE SAS** – Data step and Procedures. **CASL** – All about executing Actions

Base SAS vs. CASL #1: Import External Files

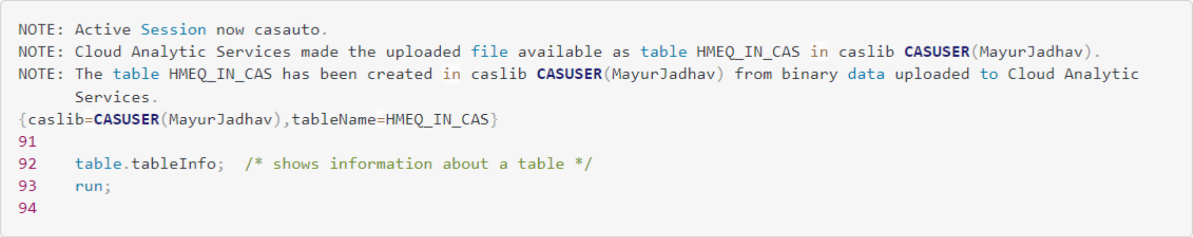
```
/* load file directly into sas  */
filename reffile "/pb/Users/MayurJadhav/Files/hmeq.csv";
proc import datafile=reffile
                    dbms=csv
                    out=work.hmeq_imported;
                    getnames=Yes;
run;
```



```
NOTE: The infile REFFILE is:
      Filename=/pb/Users/MayurJadhav/Files/hmeq.csv,
      Owner Name=UNKNOWN,Group Name=UNKNOWN,
      Access Permission=-rw-r--r--,
      Last Modified=05Jul2024:21:06:17,
      File Size (bytes)=438194
NOTE: 5960 records were read from the infile REFFILE.
      The minimum record length was 21.
      The maximum record length was 83.
NOTE: The data set WORK.HMEQ_IMPORTED has 5960 observations and 13 variables.
NOTE: DATA statement used (Total process time):
      real time           0.01 seconds
      cpu time            0.01 seconds

5960 rows created in WORK.HMEQ_IMPORTED from REFFILE.
```

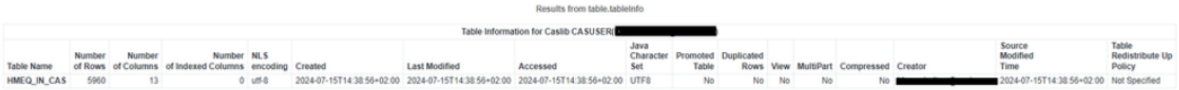
```
/* load file directly into CAS  */
proc cas;
            session casauto;
            upload path="/pb/Users/MayurJadhav/Files/hmeq.csv"
            importoptions={filetype="CSV" getNames=True}
            casout={
                        name="hmeq_in_cas"
                        replace=True
                        }
;
run;

table.tableInfo;  /* shows information about a table */
run;
```

Output Log:



```
NOTE: Active Session now casauto.
NOTE: Cloud Analytic Services made the uploaded file available as table HMEQ_IN_CAS in caslib CASUSER(MayurJadhav).
NOTE: The table HMEQ_IN_CAS has been created in caslib CASUSER(MayurJadhav) from binary data uploaded to Cloud Analytic
      Services.
{caslib=CASUSER(MayurJadhav),tableName=HMEQ_IN_CAS}

      table.tableInfo;  /* shows information about a table */
      run;

```

Results:



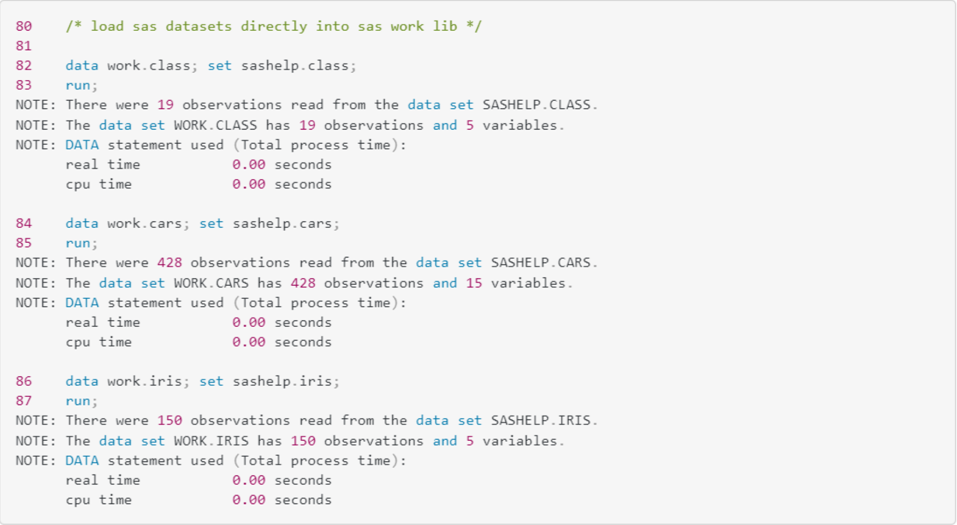
Results from table.tableInfo

Table Information for Caslib CASUSER([redacted])

Table Name	Number of Rows	Number of Columns	Number of Indexed Columns	NLS encoding	Created	Last Modified	Accessed	Java Character Set	Promoted Table	Duplicated Rows	View	MultiPart	Compressed	Creator	Source Modified Time	Table Redistribute Up Policy
HMEQ_IN_CAS	5960	13	0	utf-8	2024-07-15T14:38:56+02:00	2024-07-15T14:38:56+02:00	2024-07-15T14:38:56+02:00	UTF8	No	No	No	No	No	[redacted]	2024-07-15T14:38:56+02:00	Not Specified

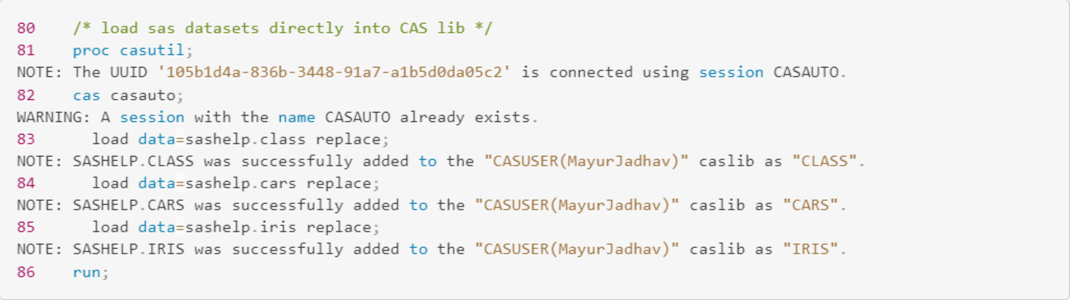
Base SAS vs. CASL #2: Load Datasets

```
/* load sas datasets directly into sas work lib */

data work.class; set sashelp.class; run;
data work.cars; set sashelp.cars; run;
data work.iris; set sashelp.iris; run;
```

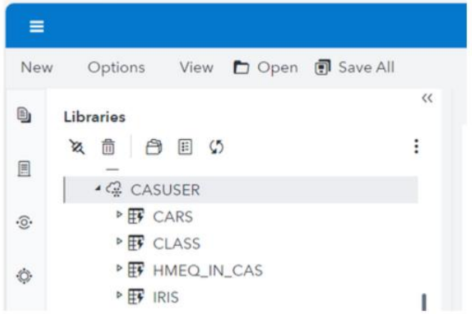


```
80    /* load sas datasets directly into sas work lib */
81
82    data work.class; set sashelp.class;
83    run;
NOTE: There were 19 observations read from the data set SASHELP.CLASS.
NOTE: The data set WORK.CLASS has 19 observations and 5 variables.
NOTE: DATA statement used (Total process time):
      real time           0.00 seconds
      cpu time            0.00 seconds

84    data work.cars; set sashelp.cars;
85    run;
NOTE: There were 428 observations read from the data set SASHELP.CARS.
NOTE: The data set WORK.CARS has 428 observations and 15 variables.
NOTE: DATA statement used (Total process time):
      real time           0.00 seconds
      cpu time            0.00 seconds

86    data work.iris; set sashelp.iris;
87    run;
NOTE: There were 150 observations read from the data set SASHELP.IRIS.
NOTE: The data set WORK.IRIS has 150 observations and 5 variables.
NOTE: DATA statement used (Total process time):
      real time           0.00 seconds
      cpu time            0.00 seconds
```

```
/* load sas datasets directly into CAS lib */
proc casutil;
  cas casauto;
  load data=sashelp.class replace;
  load data=sashelp.cars replace;
  load data=sashelp.iris replace;
run;
```

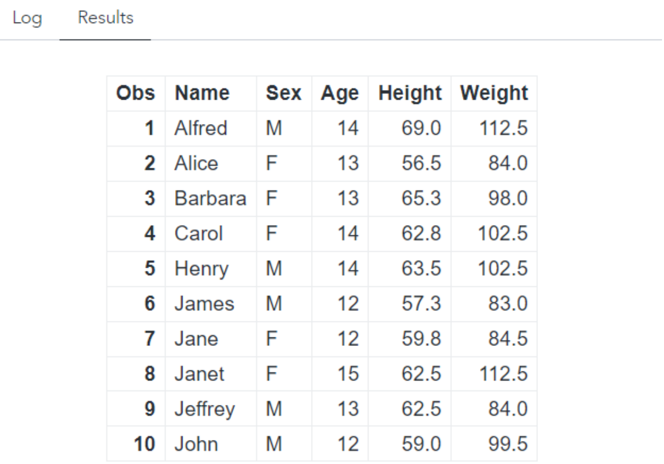


```
80    /* load sas datasets directly into CAS lib */
81    proc casutil;
NOTE: The UUID '105b1d4a-836b-3448-91a7-a1b5d0da05c2' is connected using session CASAUTO.
82    cas casauto;
WARNING: A session with the name CASAUTO already exists.
83      load data=sashelp.class replace;
NOTE: SASHELP.CLASS was successfully added to the "CASUSER(MayurJadhav)" caslib as "CLASS".
84      load data=sashelp.cars replace;
NOTE: SASHELP.CARS was successfully added to the "CASUSER(MayurJadhav)" caslib as "CARS".
85      load data=sashelp.iris replace;
NOTE: SASHELP.IRIS was successfully added to the "CASUSER(MayurJadhav)" caslib as "IRIS".
86    run;
```



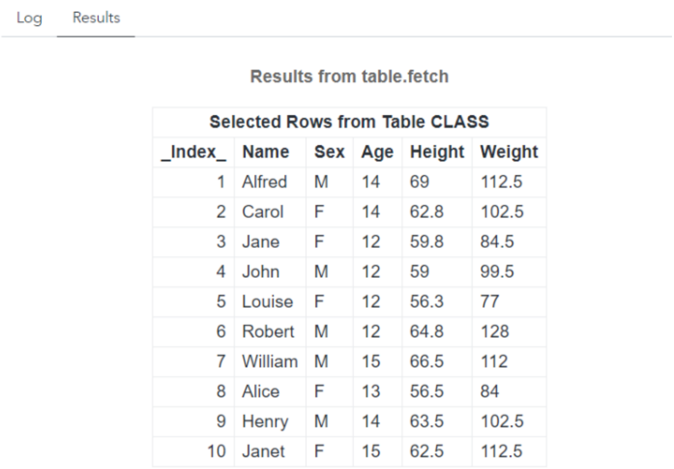
Base SAS vs. CASL #3: Print Sample Data Values

```
/* print sample data values */

proc print data=work.class (obs=10);
        var name sex age height weight;
run;
```



Obs	Name	Sex	Age	Height	Weight
1	Alfred	M	14	69.0	112.5
2	Alice	F	13	56.5	84.0
3	Barbara	F	13	65.3	98.0
4	Carol	F	14	62.8	102.5
5	Henry	M	14	63.5	102.5
6	James	M	12	57.3	83.0
7	Jane	F	12	59.8	84.5
8	Janet	F	15	62.5	112.5
9	Jeffrey	M	13	62.5	84.0
10	John	M	12	59.0	99.5

```
/* print sample data values in CASL */
proc cas;
  session casauto;

  table.fetch /
    format=true,
                      fetchvars = {"name", "sex", "age", "height", "weight"},
    table="class",
    to=10;
run;
quit;
```



Results from table.fetch

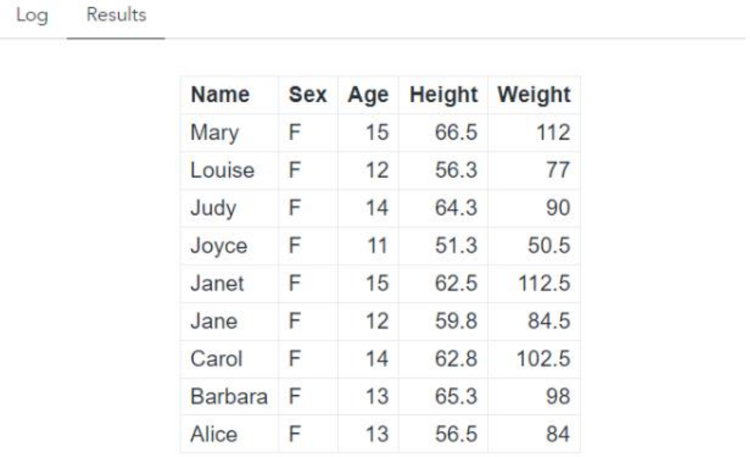
Selected Rows from Table CLASS

Index	Name	Sex	Age	Height	Weight
1	Alfred	M	14	69	112.5
2	Carol	F	14	62.8	102.5
3	Jane	F	12	59.8	84.5
4	John	M	12	59	99.5
5	Louise	F	12	56.3	77
6	Robert	M	12	64.8	128
7	William	M	15	66.5	112
8	Alice	F	13	56.5	84
9	Henry	M	14	63.5	102.5
10	Janet	F	15	62.5	112.5

table.fetch Action: Fetches rows from a table or view.

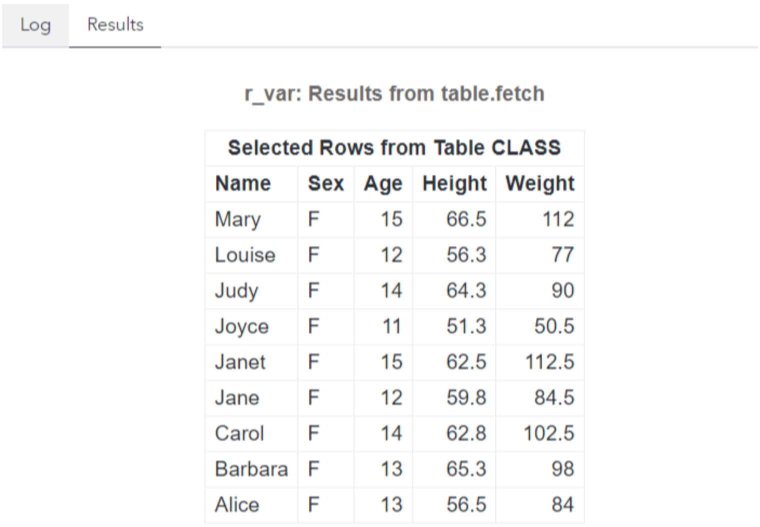
SAS vs. CASL #4: Data Handling (Filtering, Grouping, and Sorting)

```
/* data handling: filtering, grouping, and sorting by variables */

proc sql outobs=10;
                    select name, sex, age, height, weight from work.class
                    where sex="F"
                    group by name, age
                    order by name desc, age desc;
quit;
```



Name	Sex	Age	Height	Weight
Mary	F	15	66.5	112
Louise	F	12	56.3	77
Judy	F	14	64.3	90
Joyce	F	11	51.3	50.5
Janet	F	15	62.5	112.5
Jane	F	12	59.8	84.5
Carol	F	14	62.8	102.5
Barbara	F	13	65.3	98
Alice	F	13	56.5	84

```
/* data handling in CASL: filtering, grouping, and sorting by variables */
proc cas;
  session casauto;
  classtbl.name  ="class";
  classtbl.where = "sex = 'F'";
  fvars = {"name", "sex", "age", "height", "weight"};

/* results of the fetch action are saved in the "r_var" variable */
  table.fetch  result=r_var/
    format=false,
    fetchvars = fvars,  index=false,
    sortby={
      {name="name", order="descending"},
      {name="age", order="descending"}
    },
    table=classtbl,
    to=10;
describe r_var;
print r_var;
run;  quit;
```



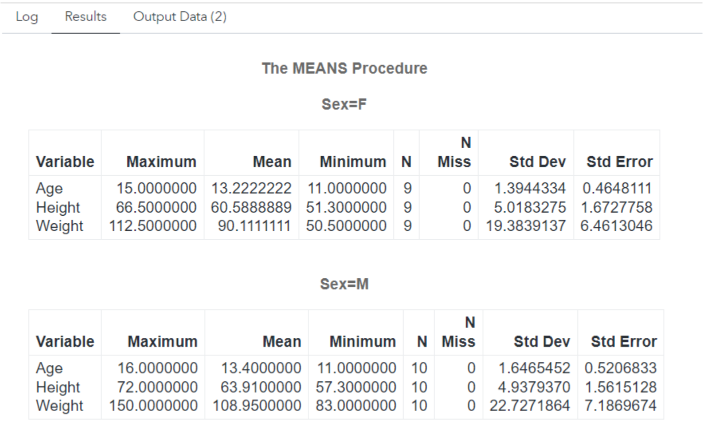
r_var: Results from table.fetch

Selected Rows from Table CLASS

Name	Sex	Age	Height	Weight
Mary	F	15	66.5	112
Louise	F	12	56.3	77
Judy	F	14	64.3	90
Joyce	F	11	51.3	50.5
Janet	F	15	62.5	112.5
Jane	F	12	59.8	84.5
Carol	F	14	62.8	102.5
Barbara	F	13	65.3	98
Alice	F	13	56.5	84

SAS vs. CASL #5: Generate Descriptive Statistics

```
/* Generate descriptive statistics */
proc sort data=class out=classbysex;
  by sex;
run;
proc means data=classbysex max mean min n nmiss std stderr;
                    by sex;
    output out=summary_stats
    ;
run;
```



The MEANS Procedure

Sex=F

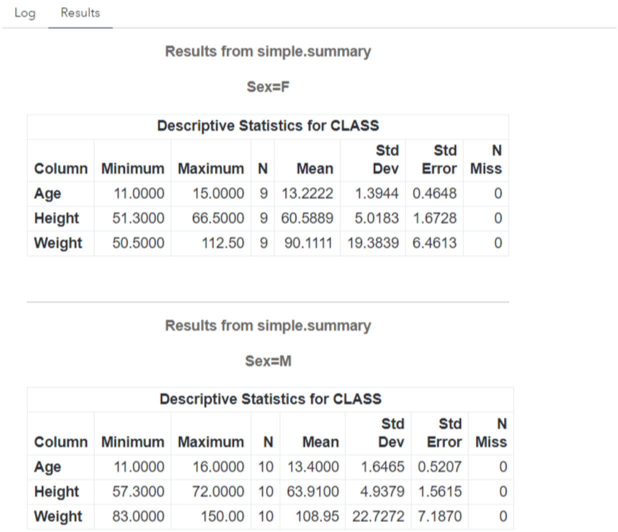
Variable	Maximum	Mean	Minimum	N	N Miss	Std Dev	Std Error
Age	15.0000000	13.2222222	11.0000000	9	0	1.3944334	0.4648111
Height	66.5000000	60.5888889	51.3000000	9	0	5.0183275	1.6727758
Weight	112.5000000	90.1111111	50.5000000	9	0	19.3839137	6.4613046

Sex=M

Variable	Maximum	Mean	Minimum	N	N Miss	Std Dev	Std Error
Age	16.0000000	13.4000000	11.0000000	10	0	1.6465452	0.5206833
Height	72.0000000	63.9100000	57.3000000	10	0	4.9379370	1.5615128
Weight	150.0000000	108.9500000	83.0000000	10	0	22.7271864	7.1869674

```
/* Generate descriptive statistics in CASL*/
proc cas;
 tbl1.name = "class";
 tbl1.groupBy = "sex";

 simple.summary /
    table = tbl1
    subSet = {"MAX", "MEAN", "MIN", "N", "NMISS", "STD", "STDERR"};
run;
quit;
```

simple.summary Action: Generates descriptive statistics of numeric variables such as the sample mean, sample variance, sample size, sum of squares, and so on.



Results from simple.summary

Sex=F

Descriptive Statistics for CLASS

Column	Minimum	Maximum	N	Mean	Std Dev	Std Error	N Miss
Age	11.0000	15.0000	9	13.2222	1.3944	0.4648	0
Height	51.3000	66.5000	9	60.5889	5.0183	1.6728	0
Weight	50.5000	112.50	9	90.1111	19.3839	6.4613	0

Results from simple.summary

Sex=M

Descriptive Statistics for CLASS

Column	Minimum	Maximum	N	Mean	Std Dev	Std Error	N Miss
Age	11.0000	16.0000	10	13.4000	1.6465	0.5207	0
Height	57.3000	72.0000	10	63.9100	4.9379	1.5615	0
Weight	83.0000	150.00	10	108.95	22.7272	7.1870	0

When to CASL

Long Data Steps: CASL your code if data steps take 25 minutes or more to run.

Slow Procs: Convert PROC steps to CASL if they run longer than 25 minutes and have CAS equivalents.

No Data Science Procs: Use CASL if you lack data science and machine learning procedures in SAS 9.

Available Procs: Even with similar SAS 9 procedures, CASL can offer performance benefits.

Large Data Size: Start using CASL if your data size is 50GB or more; benefits can also be seen with 25GB datasets.

Extensive By Group Processing: Test and benchmark CASL for tasks involving heavy by-group processing, sorting, or merging.

SMP vs. MPP: For SMP (single server) environments, consider not using CASL unless jobs are non-performant; MPP (distributed server) environments benefit more from CASL.

Minimize Data Movement: Reduce back-and-forth data transfers between CAS and SAS 9 by strategically using CASL for data prep tasks.

https://www.lexjansen.com/phuse-us/2022/as/PRE_AS09.pdf

Thank You!

References

- https://documentation.sas.com/doc/en/pgmsascdc/9.4_3.4/caslpg/titlepage.htm
- https://communities.sas.com/t5/SAS-Communities-Library/BASE-SAS-vs-CASL-A-Comparative-Analysis-That-Will-Help-You-in/ta-p/935794
- https://communities.sas.com/t5/SAS-Communities-Library/CASL-It-s-like-Base-SAS-with-the-MACRO-Built-In/ta-p/651202
- https://www.lexjansen.com/phuse-us/2022/as/PRE_AS09.pdf
- https://support.sas.com/resources/papers/proceedings20/4454-2020.pdf