28th -29th October | Tokyo | PharmaSUG Japan 2026

AT-028

The Power of Unified Analytic Platform: Harmonizing SAS, R, and GenAI on SAS Viya

William Kuan
Sr. Industry Consultant, Health and Life Sciences Solutions

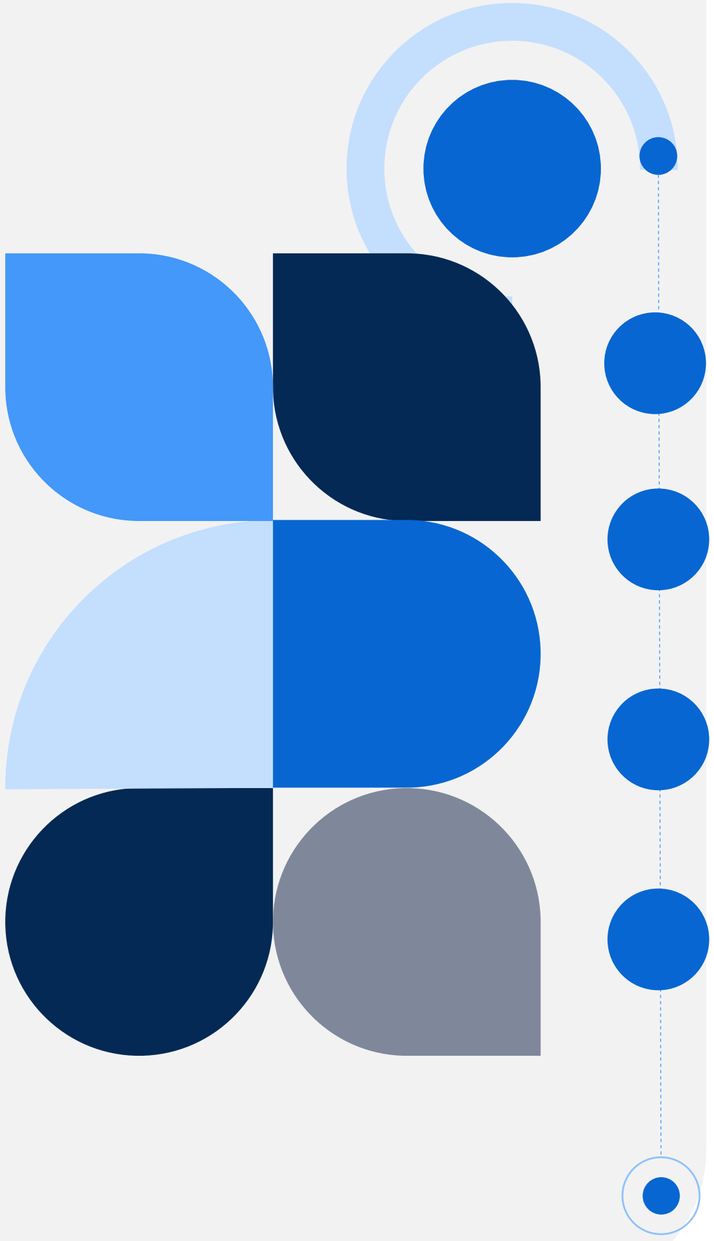


Copyright © SAS Institute Inc. All rights reserved.

DISCLAIMER

- This presentation reflects the opinions of the presenters and should not be construed or considered to represent the opinions of presenter’s employer.

Copyright © SAS Institute Inc. All rights reserved.



AGENDA

Benefits of the Unified Analytics Platform

Work Seamlessly with R and SAS together - **PROC R**

Access to Open File Format – **SAS/ACCESS to DuckDB**

Increase Efficiency - **SAS Viya Copilot, SAS Viya MCP Server**

Copyright © SAS Institute Inc. All rights reserved.

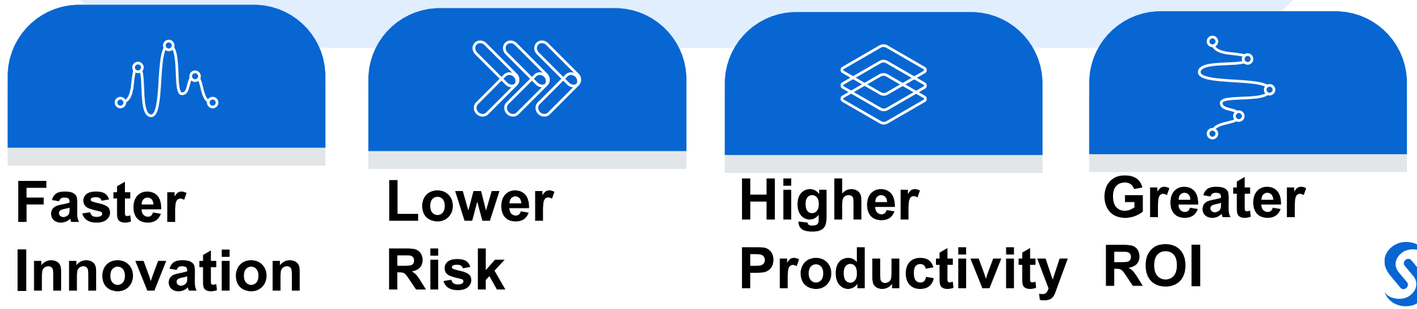
Why we need a single, unified analytics platform?

Challenges with disconnected tools

- Data silos
- Duplicate infrastructure
- Multiple versions of the truth
- Delayed decision-making
- Higher compliance risk
- Difficult in scaling AI initiatives

Benefits with the unified analytics platform

- Accelerate time-to-insight
- Break down organizational silos
- Increase trust in decisions
- Scale AI responsibly
- Maximize the value of enterprise data
- Reduce total cost of ownership



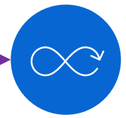
Copyright © SAS Institute Inc. All rights reserved.

Introducing PROC R

Copyright © SAS Institute Inc. All rights reserved.

Why use Open-source and SAS together?

Option	
Use only SAS	Reliable and compliant, but may adopt new analytical methods more slowly
Use only open-source (e.g., R, Python)	Access to new methods quickly, but can create challenges around governance, validation, reproducibility, and operationalization
Use SAS + open-source together	Balances innovation and compliance



Explore New Methods



Preserve Regulatory Confidence



Attract and Retain Talent



Avoid Analytics Silos

Copyright © SAS Institute Inc. All rights reserved.

Key Features

PROC R

- Execute any R code inside a block in a SAS program
- Execute SAS code blocks in the R code
- Pass macro variables from SAS to R and back
- Call SAS functions in the R code
- Move sas7bdat datasets in and out of R data frames
- Create R graphics and display in results window

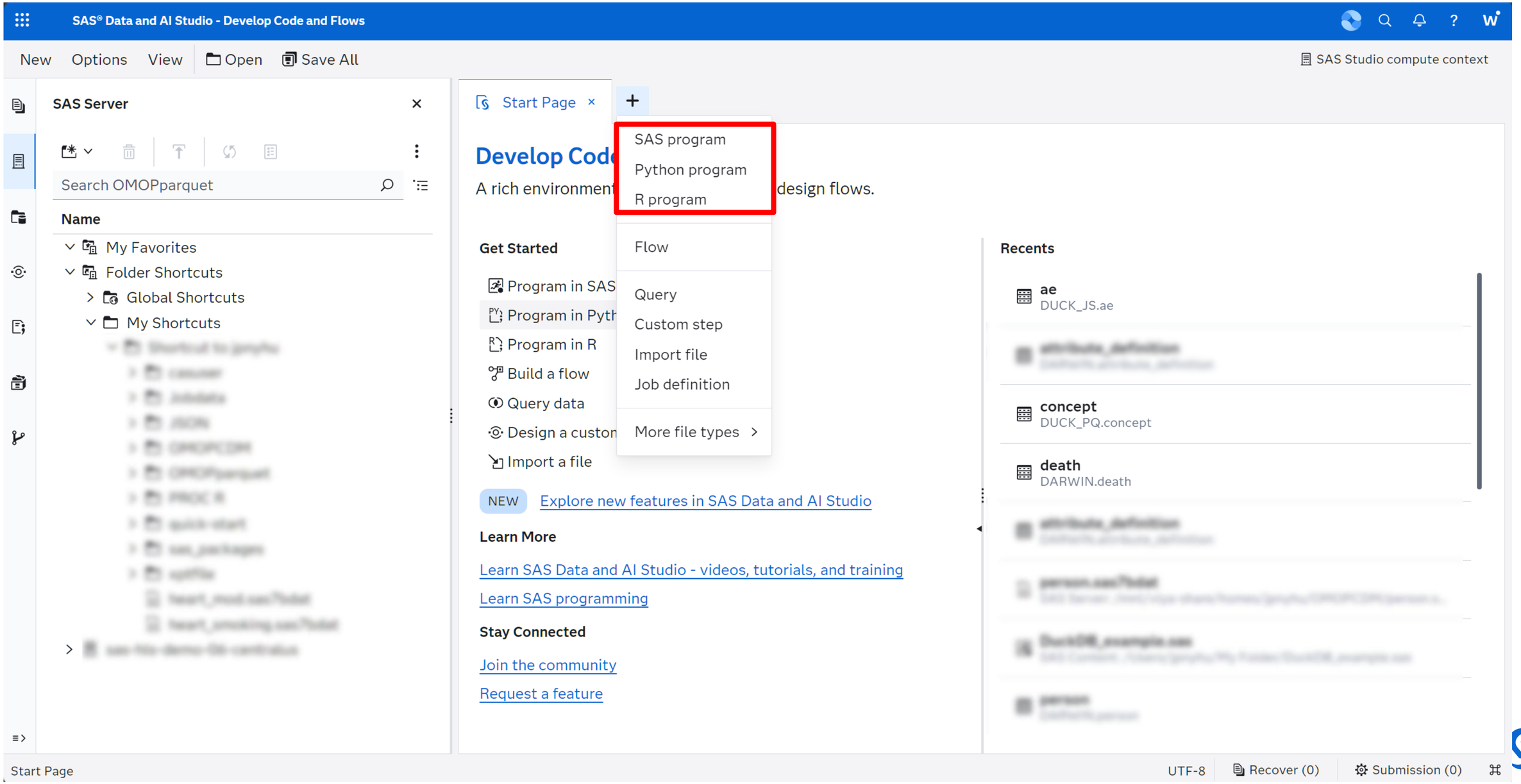
Copyright © SAS Institute Inc. All rights reserved.

SAS-specific functions in PROC R

Function	Description
`sd2df("sas_dataset")`	Exports a SAS data set to a local R data frame.
`df2sd(df, "sas_dataset")`	Transfers an R data frame to a SAS data set.
`symget("sas-macro-variable")`	Retrieves the value of a SAS macro variable.
`symput("sas-macro-variable", "R-variable")`	Sets the value of a SAS macro variable.
`sasfnc("function","arg1","arg2",…)`	Calls a SAS Function with the list of passed arguments.
`submit("sas-code")`	Submits SAS code from within the R session.
`rplot(r-plot-object)`	Prints R graphs in the Results window.
`show(r-object, <title="title">, <count=number>)`	Displays R objects in the Results window.

Copyright © SAS Institute Inc. All rights reserved.

Run SAS, R and Python program in the same environment



Combine R and SAS in the same program

Start Page | R_example.sas ×

Run | Cancel | Clear

Code

SAS Content: /Users/[redacted]/My Folder/R_example.sas

```
1
2   PROC R;
3   SUBMIT;
4
5   ver = R.version.string
6   location = R.home()
7   pkg = as.data.frame(installed.packages()[,c(1,3)])
8
9   df2sd(pkg, "packages", "work")
10  symput("version_s", ver)
11  symput("location_s", location)
12
13  endsubmit;
14  run;
15
16  TITLE "Package Listing from &version_s";
17  TITLE2 "R Location: &location_s";
18
19  PROC SQL;
20     Select Package, version
21     from packages
22     order by upcase(Package);
23  quit;
24
```

R code

SAS code

Create a df “pkg” to store the list of all installed R packages in the SAS Viya environment.

Convert “pkg” df to the SAS dataset “packages” in WORK library — **df2sd()**

Push R var. “ver” to SAS macro var. “version_s” — **symput()**

Use SAS macro var. “version_s” in SAS code

Query the “packages” SAS dataset, which converted from “pkg” df

sas

Start Page | R_example.sas

Run | Cancel | Clear | Debug | Copy to flow | Snippet

Code

SAS Content: /Users[redacted] Folder/R_example.sas

```
1
2   PROC R;
3   SUBMIT;
4
5   ver = R.version.string
6   location = R.home()
7   pkg = as.data.frame(installed.packages()[,c(1,3)])
8
9   df2sd(pkg, "packages", "work")
10  symput("version_s", ver)
11  symput("location_s", location)
12
13  endsubmit;
14  run;
15
16  TITLE "Package Listing from &version_s";
17  TITLE2 "R Location: &location_s";
18
19  PROC SQL;
20     Select Package, version
21     from packages
22     order by upcase(Package);
23  quit;
24
25
```

Log | Results | Output Data (1)

Package Listing from R version 4.4.3 (2025-02-28)
R Location: /opt/sas/viya/home/sas-[redacted]/R-4.4.3.1780067812199582/lib64/R

Package	Version
abind	1.4-8
admiral	1.5.0
admiraldev	1.5.0
arrow	25.0.0
askpass	1.2.1
assertthat	0.2.1
backports	1.5.1
bartCause	1.0-10
base	4.4.3
base64enc	0.1-6
bbmle	1.0.25.1
BCEA	2.4.83
bdsmatrix	1.3-7
BH	1.90.0-1
bigD	0.3.1
BiocManager	1.30.27

SAS macro var. “version_s” resolved as the value of R var. “ver”

Log | Results | Output Data (1)

WORK.PACKAGES

Columns: 2 Rows: 378

Enter expression

	Package	Version
1	abind	1.4-8
2	admiral	1.5.0
3	admiraldev	1.5.0
4	arrow	25.0.0
5	askpass	1.2.1
6	assertthat	0.2.1
7	backports	1.5.1
8	bartCause	1.0-10
9	base	4.4.3
10	base64enc	0.1-6
11	bbmle	1.0.25.1
12	BCEA	2.4.83
13	bdsmatrix	1.3-7
14	BH	1.90.0-1

Success | Ln 25, Col 1 | UTF-8 | Recover (17) | Submission (0)

View the content of “pkg” df in SAS Results window

View the content of “pkg” df in SAS Output Data window

Copyright © SAS Institute Inc. All rights reserved.

SAS code

```
%let inputTable = SASHELP.HEART;

proc R;
submit;
```

R code

```
# Set up libraries and read in data
library(dplyr)

df = sd2df(symget("inputTable"))          # Move SAS dataset to dataframe
dead <- df %>% filter(Status == "Dead")   # Filter

# Use SAS function LOG() to create log value of cholestrol
dead$log_cholesterol <- sapply(dead$Cholesterol, function(x) sasfnc("LOG", x))

show(dead, "Put title here", 15)          # Show dead df to Results window

df2sd(dead, "Dead_sas")                       # Create SAS dat
symput("numDead", nrow(dead))             # Extract row numbe

avg_d_age <- mean(dead$AgeAtDeath, na.rm = TRUE)
symput("Average_death_age", avg_d_age)    # Calculate the mean and Create SAS macro variable

submit("%put &=numDead;")                 # Put numDead in the Log
submit("%put &=Average_death_age;")       # Put Average_death_age in the Log

hello = sasfnc("lowcase", "CDISC SDTM")
print(hello)
```

```
endsubmit;
RUN;
```

Setup a SAS macro var. “inputTable”

Retrieve the value of the SAS macro var. “inputTable” to use in R script. Convert the SAS dataset “HEART” to R df

symget()
sd2df()

Leverage the capability of SAS function “LOG” in R script

sasfnc()

Display “dead” df in the Results window. Put a title and only show the first 15 rows

show()

submits SAS code from within the R session to put the value of “numDead” (calculated in R) back to the SAS Log using %put

submit()

sas

```
116! ;          data _null_; rc = fdelete('_rmfile'); run;          filename _rmf
116! ;%put NOTE: Character variable lengths adjusted to match maximum observed va
NOTE: Character variable lengths adjusted to match maximum observed values in the
117  %put &=numDead;
NUMDEAD=1991
118  %put &=Average_death_age;
AVERAGE_DEATH_AGE=70.5364138623807
[df2sd] SUCCESS: R data frame (15 x 18) exported to SAS dataset work._r_show
[df2sd] SUCCESS: R data frame (1991 x 18) exported to SAS dataset work.Dead_sas
[1] "cdisc sdtm"
NOTE: The PROCEDURE R printed page 3.
NOTE: PROCEDURE R used (Total process time):
      real time           0.38 seconds
      cpu time            0.15 seconds
```

Run | Cancel | Clear | Debug | Copy to flow | Snippet

Code | Log | Results | Output Data (1)

Put title here

Obs	Status	DeathCause	Sex	Chol_Status	BP_Status	Weight_Status	Smoking_Status	AgeCHDdiag	AgeAtStart	Height	Wei
1	Dead	Other	Female		Normal	Overweight	Non-smoker	.	29	62.50	1
2	Dead	Cancer	Female	Desirable	High	Overweight	Non-smoker	.	41	59.75	1
3	Dead	Other	Male	High	Normal	Normal	Non-smoker	.	53	65.50	1
4	Dead	Cerebral Vascular Disease	Male	High	Normal	Normal	Light (1-5)	.	52	62.50	1
5	Dead	Cancer	Male	Desirable	Normal	Underweight	Moderate (6-15)	56	56	67.25	1
6	Dead	Coronary	Male	Borderline	High	Overweight	Very Heavy (>	74	46	66.50	1

Start Page | * R_example.sas

Run | Cancel | Clear

Code | Log | Results | Output Data (1)

WORK.DEAD_SAS

Columns: 18 Rows: 1,991

Enter expression

#	Status	DeathCause	Sex	Chol_Status
1	Dead	Other	Female	
2	Dead	Cancer	Female	Desirable
3	Dead	Other	Male	High
4	Dead	Cerebral Vascular Disease	Male	High
5	Dead	Cancer	Male	Desirable
6	Dead	Coronary Heart Disease	Male	Borderline
7	Dead	Coronary Heart Disease	Female	Borderline
8	Dead	Unknown	Female	High
9	Dead	Coronary Heart Disease	Female	High
10	Dead	Cancer	Male	Desirable
11	Dead	Coronary Heart Disease	Male	High
12	Dead	Coronary Heart Disease	Male	High
13	Dead	Coronary Heart Disease	Male	Borderline
14	Dead	Cancer	Male	Borderline
15	Dead	Cerebral Vascular Disease	Female	High
16	Dead	Cerebral Vascular Disease	Male	Borderline
17	Dead	Cancer	Female	High
18	Dead	Cerebral Vascular Disease	Female	

Copyright © SAS Institute Inc. All rights reserved.

Leverage R's graphics capability

SAS code

```
60  /* Generate R scatterplot*/
61  %let inputTable = SASHELP.HEART;
62  %let X = Systolic;
63  %let Y = Cholesterol;
64  %let Color = Status;
65  %let shape = Sex;
66
67  PROC R;
68  submit;
```

Setup SAS macro variables for R graphs

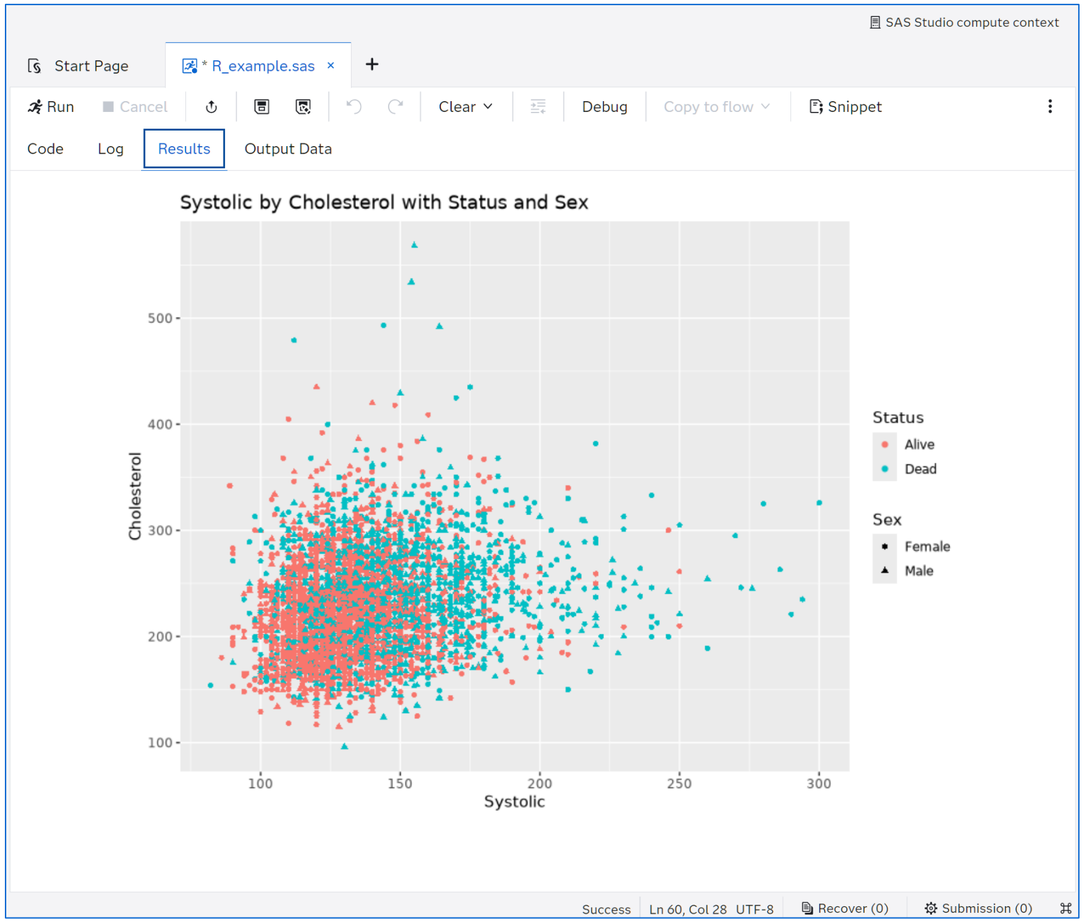
R code

```
69
70  library(ggplot2)
71  df = sd2df(symget("inputTable"))
72  plotX = sym(symget("X"))
73  plotY = sym(symget("Y"))
74  plotC = sym(symget("Color"))
75  plotS = sym(symget("shape"))
76
77  plt1 <- ggplot(df,
78              aes(x = !!plotX,
79                  y = !!plotY,
80                  color = !!plotC,
81                  shape = !!plotS)) +
82        geom_point() +
83        labs(title = paste(plotX, "by", plotY, "with", plotC, "and", plotS),
84             x = plotX,
85             y = plotY)
86
87  rplot(plt1)
88
89  endsubmit;
90  run;
```

Retrieve the value of the SAS macro var. and use it in R script.

Prints the R graph in the Results window

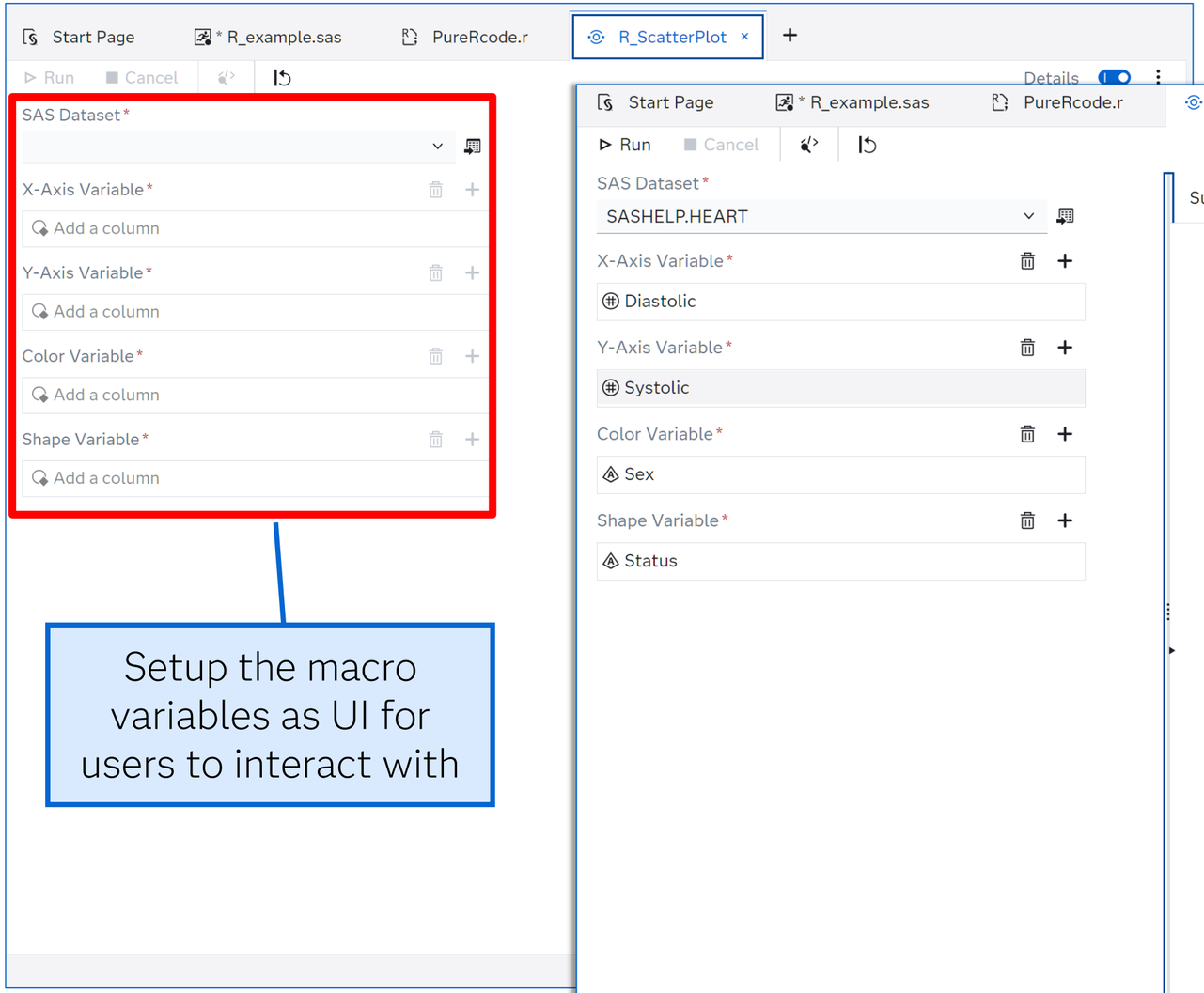
rplot()



Copyright © SAS Institute Inc. All rights reserved.

Build reuseable analytic templates for better efficiency

SAS Custom Step



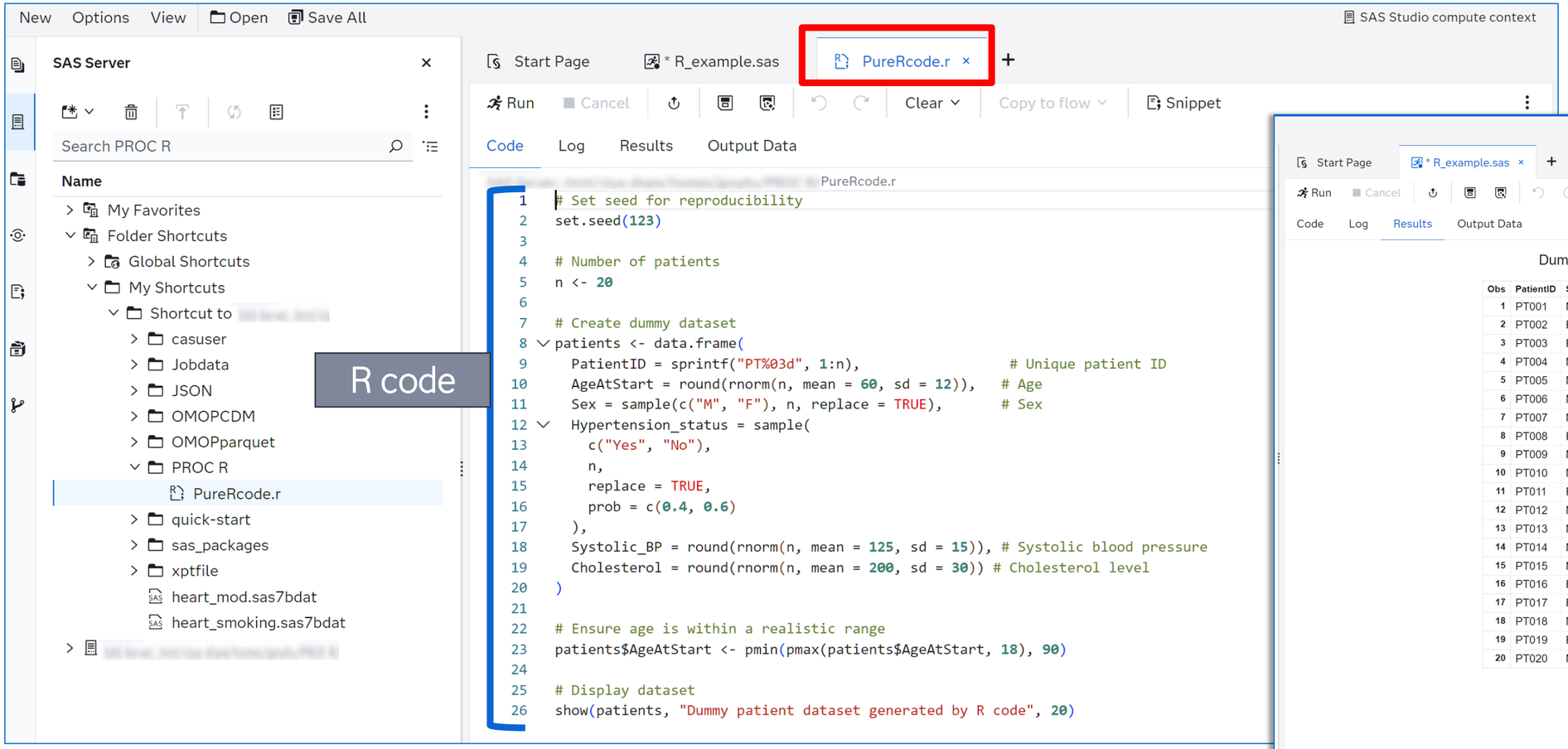
Setup the macro variables as UI for users to interact with

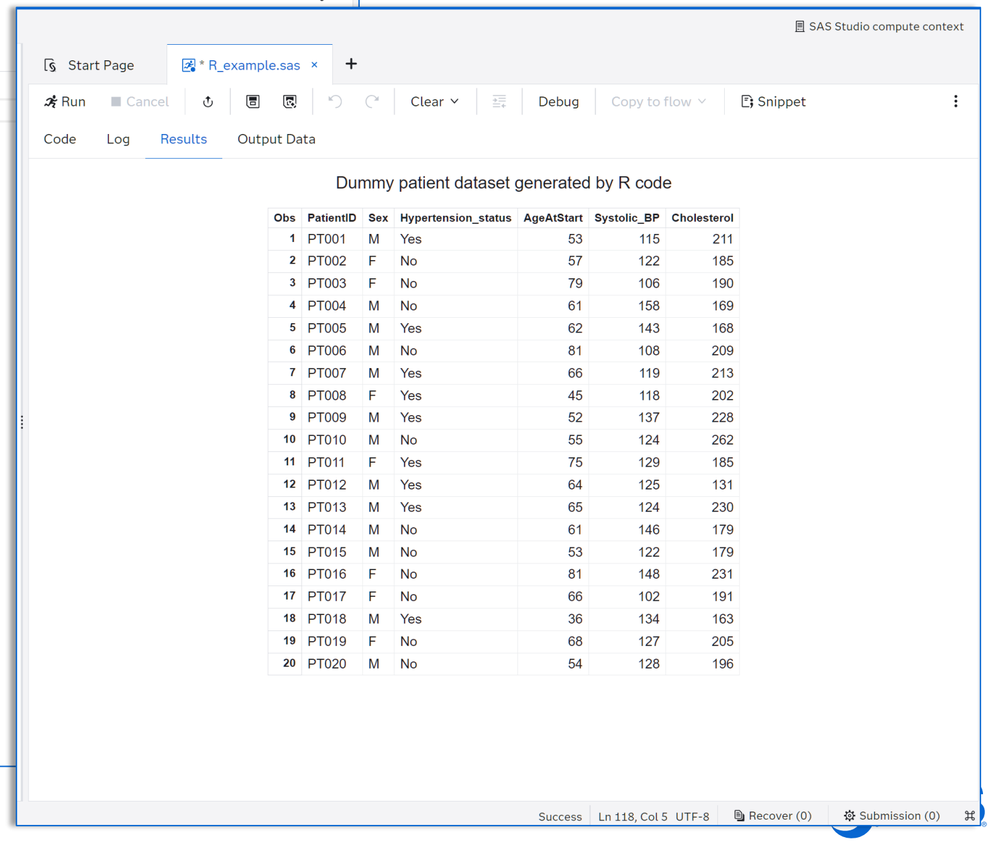


Copyright © SAS Institute Inc. All rights reserved.

Execute R program file (.r) directly via PROC R

```
116   /* Use infile to directly run a external R pgm file stored in Server side */
117   proc R infile="/mnt/viya-share/homes [redacted] /PROC R/PureRcode.r";
118   run;
```





Obs	PatientID	Sex	Hypertension_status	AgeAtStart	Systolic_BP	Cholesterol
1	PT001	M	Yes	53	115	211
2	PT002	F	No	57	122	185
3	PT003	F	No	79	106	190
4	PT004	M	No	61	158	169
5	PT005	M	Yes	62	143	168
6	PT006	M	No	81	108	209
7	PT007	M	Yes	66	119	213
8	PT008	F	Yes	45	118	202
9	PT009	M	Yes	52	137	228
10	PT010	M	No	55	124	262
11	PT011	F	Yes	75	129	185
12	PT012	M	Yes	64	125	131
13	PT013	M	Yes	65	124	230
14	PT014	M	No	61	146	179
15	PT015	M	No	53	122	179
16	PT016	F	No	81	148	231
17	PT017	F	No	66	102	191
18	PT018	M	Yes	36	134	163
19	PT019	F	No	68	127	205
20	PT020	M	No	54	128	196

Copyright © SAS Institute Inc. All rights reserved.

Introducing SAS/ACCESS to DuckDB

Working with Open File Formats

Copyright © SAS Institute Inc. All rights reserved.

Data Landscape



RDBMS



Cloud Warehouses



Lakehouse/Delta Lakes



Open File Formats

Company Confidential – For Internal Use Only
Copyright © SAS Institute Inc. All rights reserved.

Why Open File Formats (e.g., Parquet)?

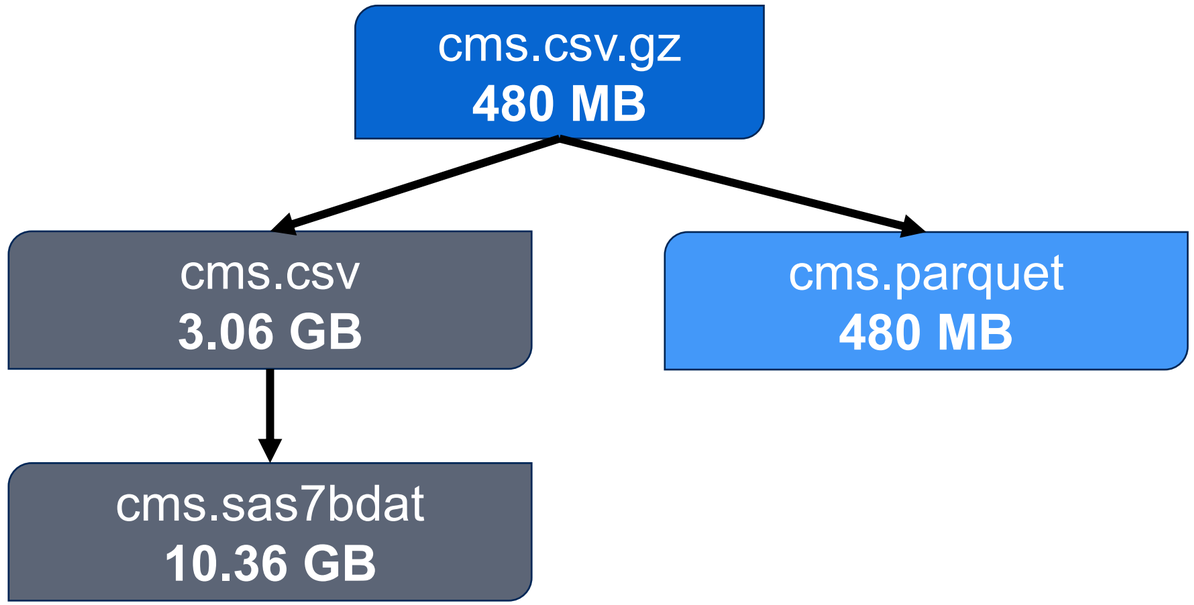
Benefits

- Advanced compression
- Lower storage costs
- Faster query performance
- Reduces I/O
- Better interoperability across tools and languages

Parquet

Test Scenario

-Raw data distributed by CMS for Provider Summary



Data Source: CMS Provider Summary by Type of Service

Company Confidential – For Internal Use Only
Copyright © SAS Institute Inc. All rights reserved.

What is SAS/ACCESS to DuckDB?

Access to, Query and Analyze Open File Formats

- DuckDB has become a favored analytics engine for modern open file formats such as **Parquet**, Iceberg, **JSON** and Delta Lake.
- High-performance from an in-process query engine
- Optimized for analytics
- Minimize data movement
- Avoid duplicating data

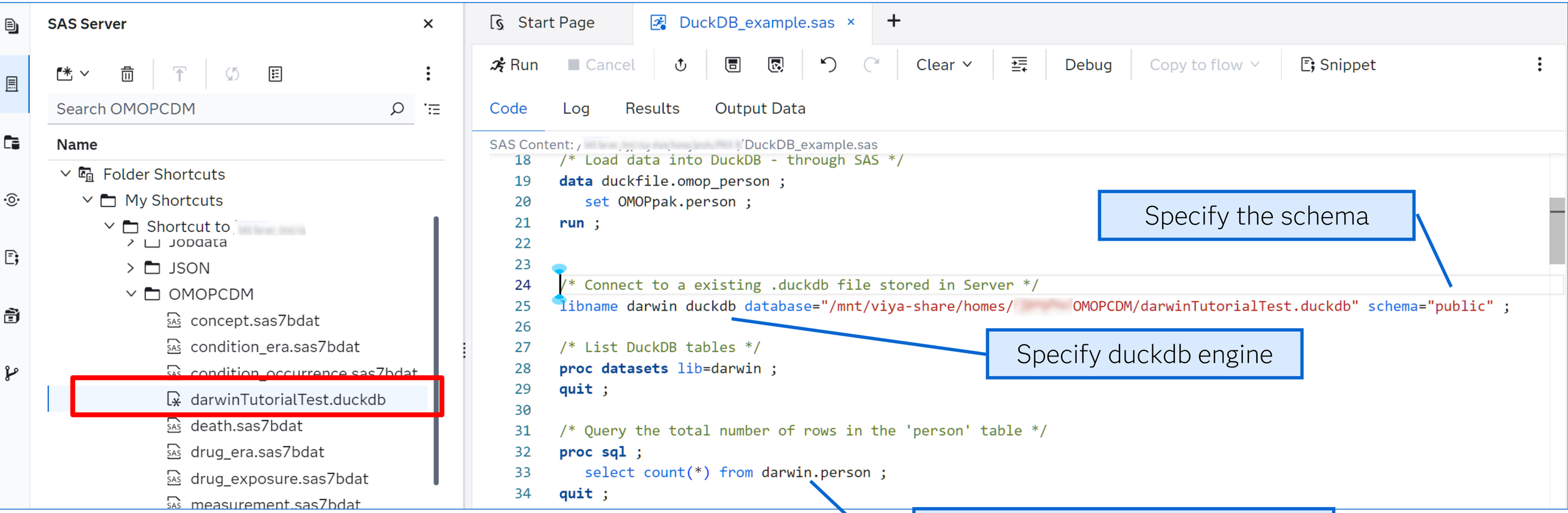
Test Scenario (continued)

- Sort the data by Provider Zip, Type and Submitted Charges
- Perform Geographic Analysis by State & RUCA (Rural/Urban)
- Analyze using:
  - ***BASE*** *engine using* ***sas7bdat*** *as file format*
  - ***DuckDB*** *engine using* ***parquet*** *as file format*

	DuckDB	BASE	Difference
Import CSV.GZ	60 sec	780 sec	**13.0x**
PROC SORT	43.0 sec	144.2 sec	**3.3x**
SORT: PROC SQL	17.7 sec	153.0 sec	**8.6x**
PROC FREQ	2.2 sec	33 sec	**15x**
PROC MEANS	2.3 sec	42 sec	**14x**
Total	**126 sec**	**1,152 sec**	

Company Confidential – For Internal Use Only
Copyright © SAS Institute Inc. All rights reserved.

Access parquet files in duckdb database via DuckDB engine



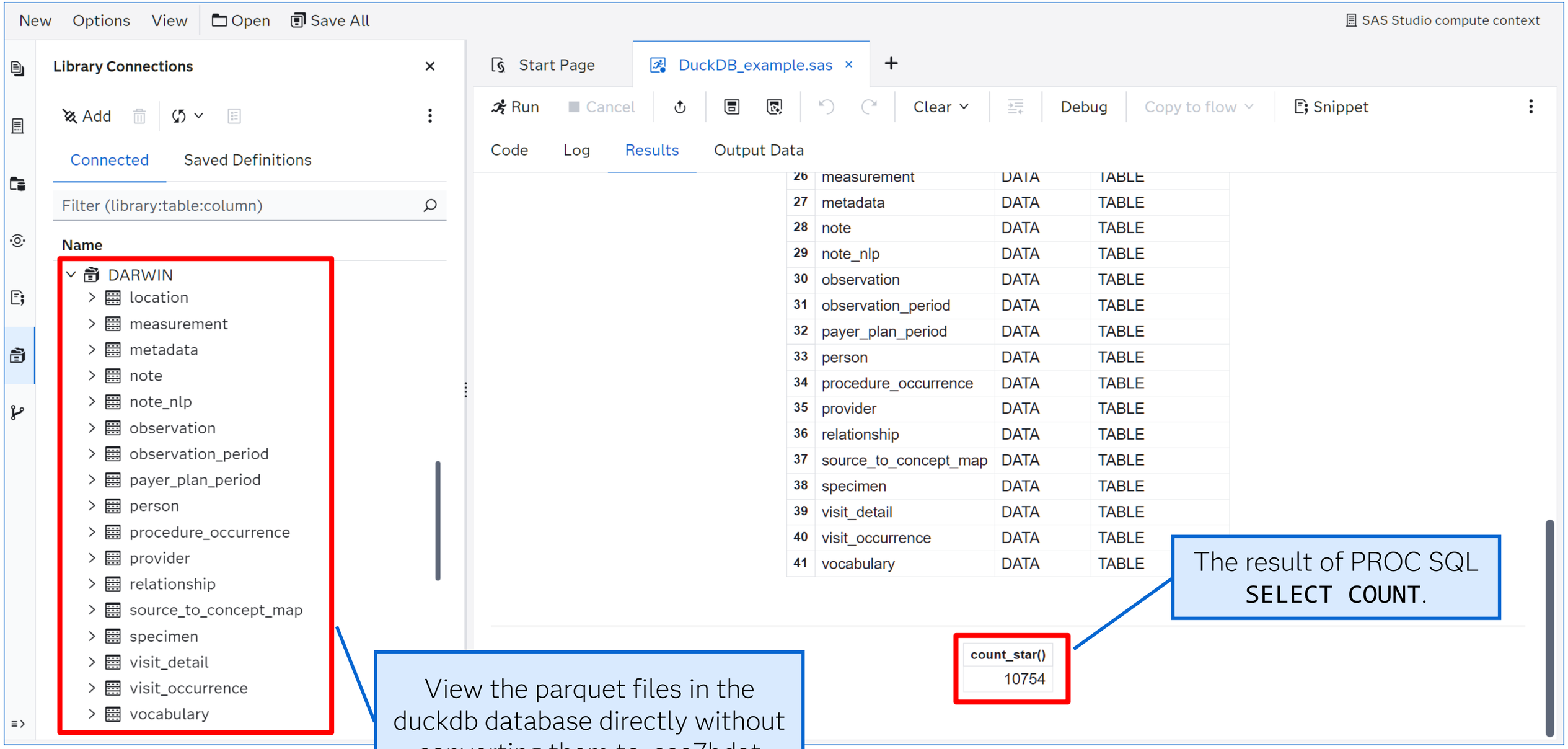
Specify the schema

Specify duckdb engine

Query the parquet files in the duckdb database directly using PROC SQL without converting them to .sas7bdat

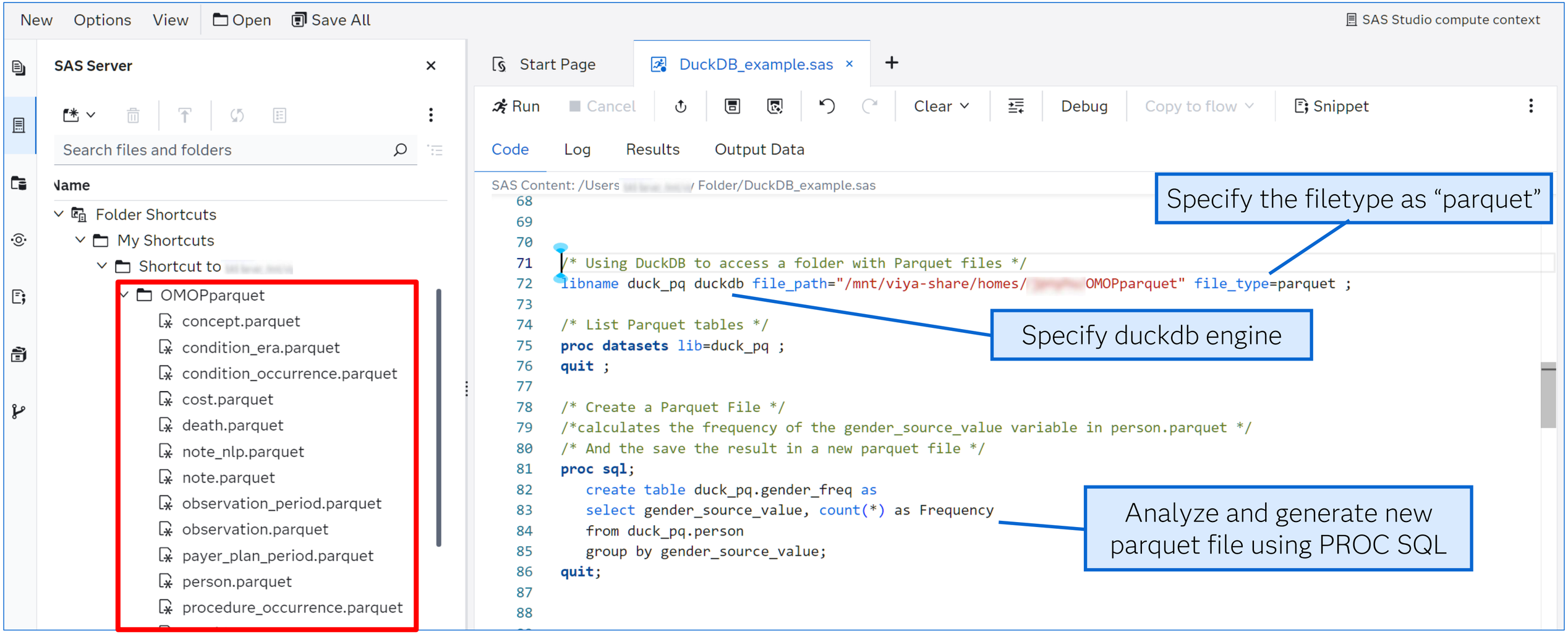
You can leverage other PROC SQL capabilities (e.g., **FROM, WHERE, ORDER BY, JOIN**) to analyze parquet files directly without duplicating data

Copyright © SAS Institute Inc. All rights reserved.



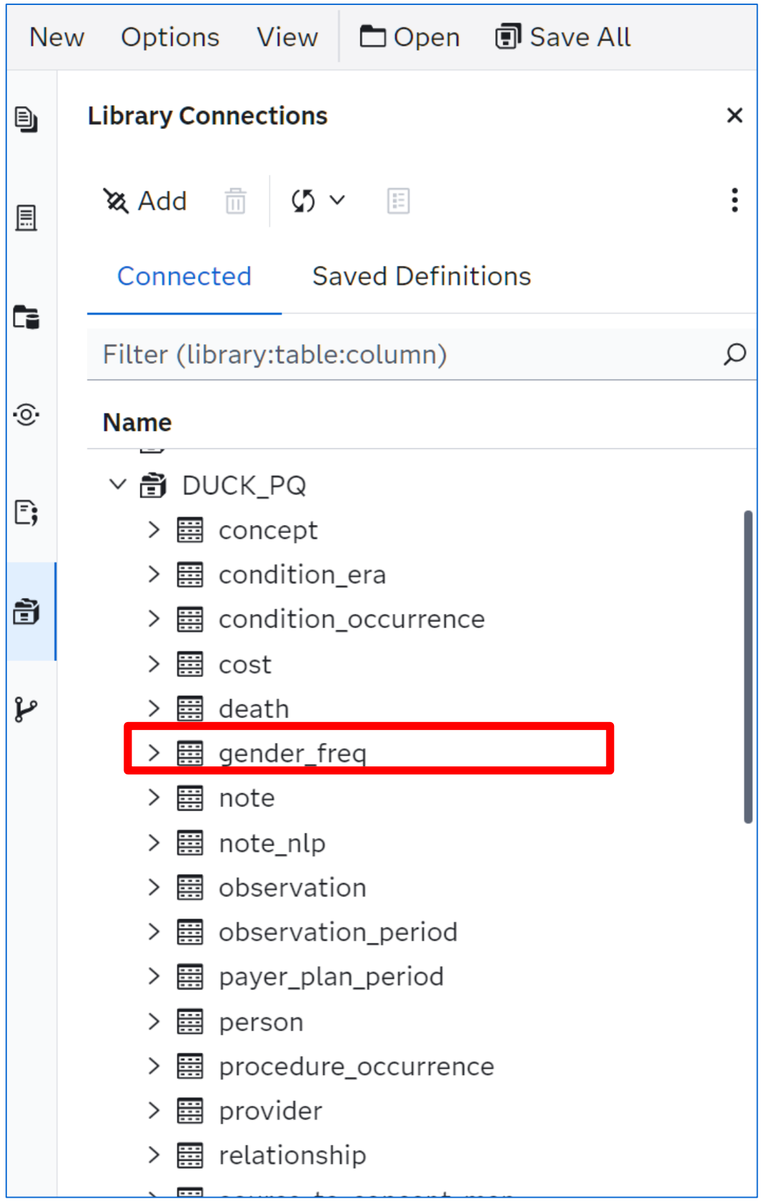
Copyright © SAS Institute Inc. All rights reserved.

Access parquet files in a folder

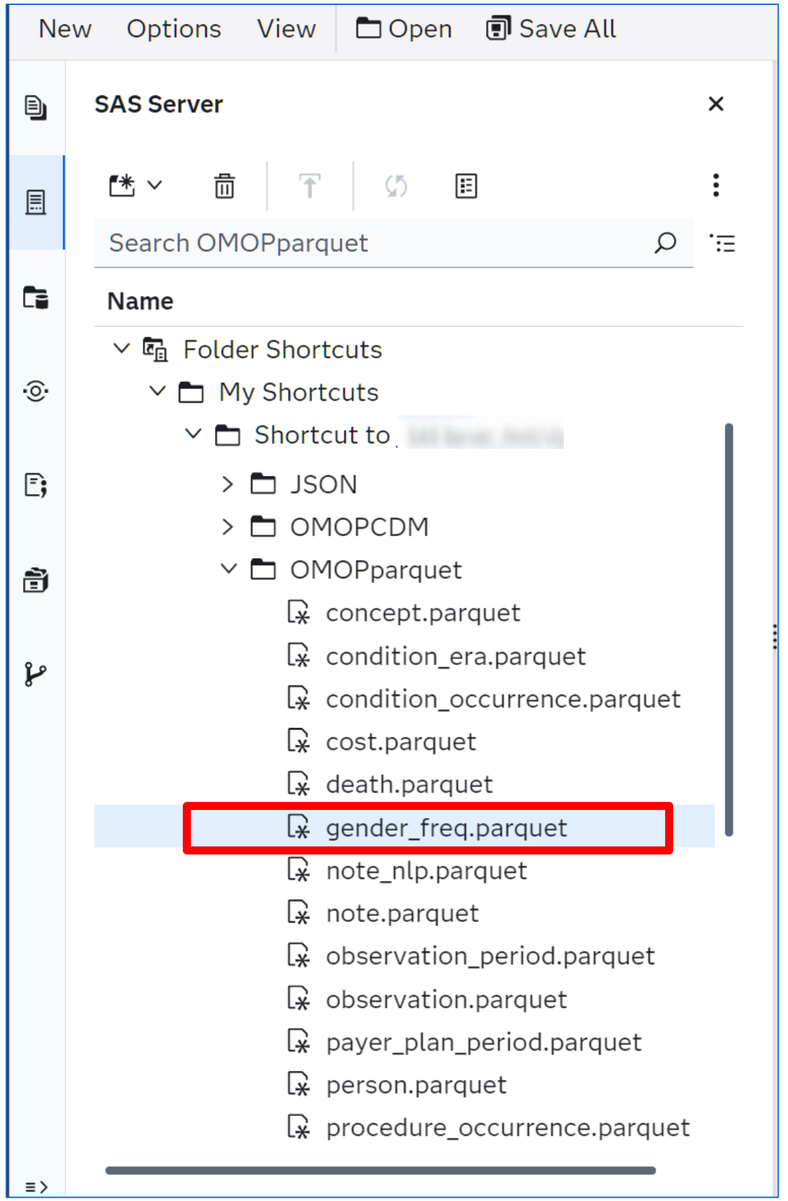


Copyright © SAS Institute Inc. All rights reserved.

SAS Library

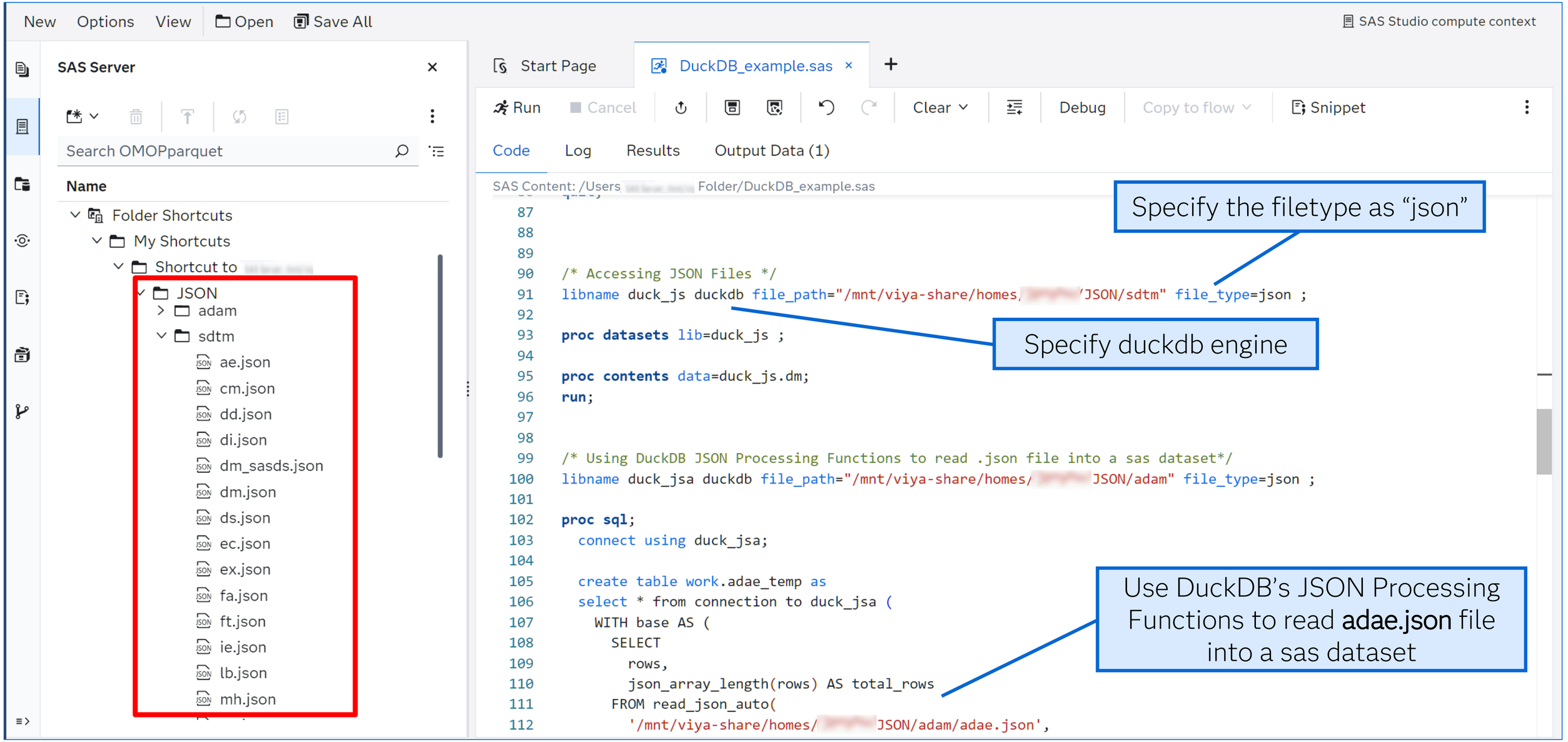


SAS Server

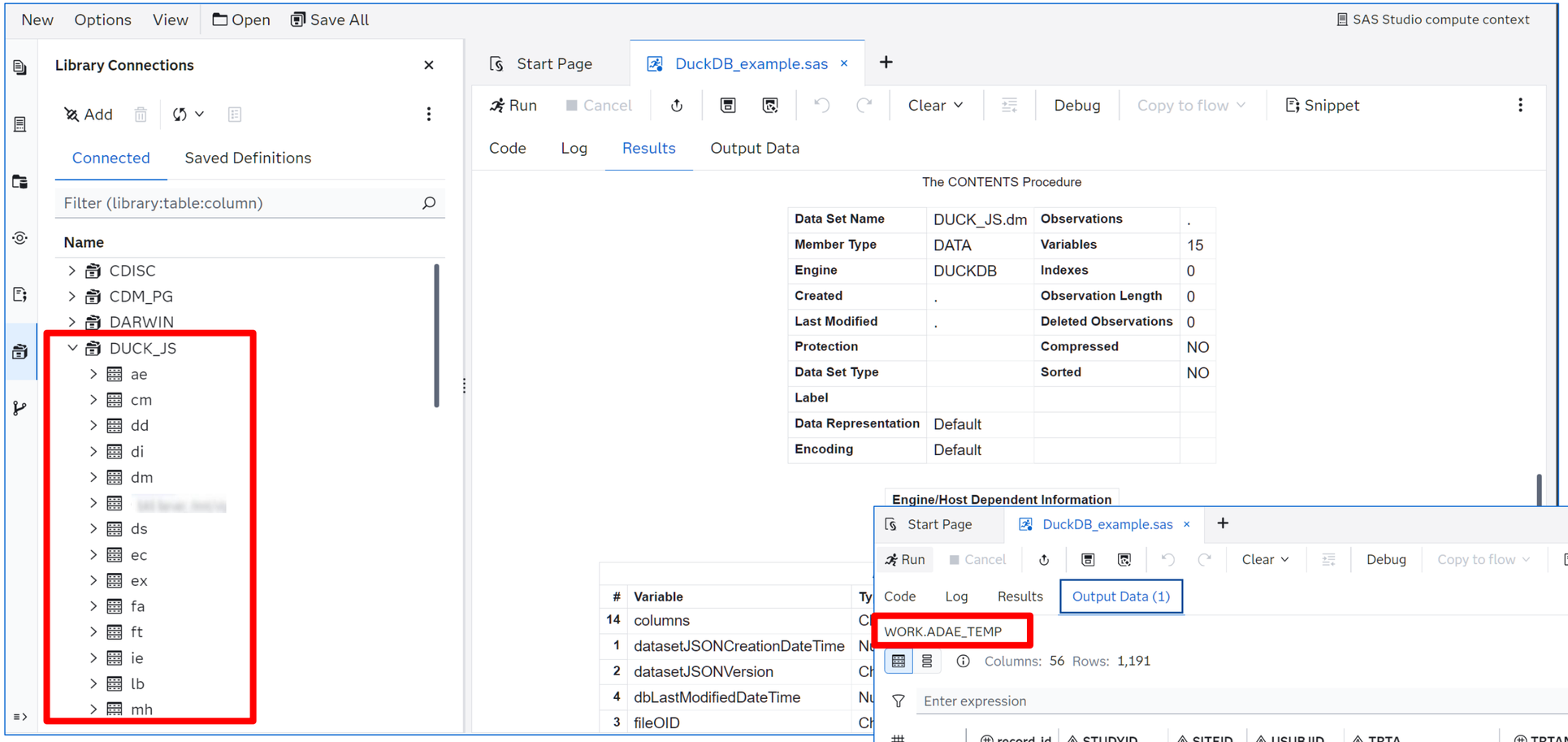


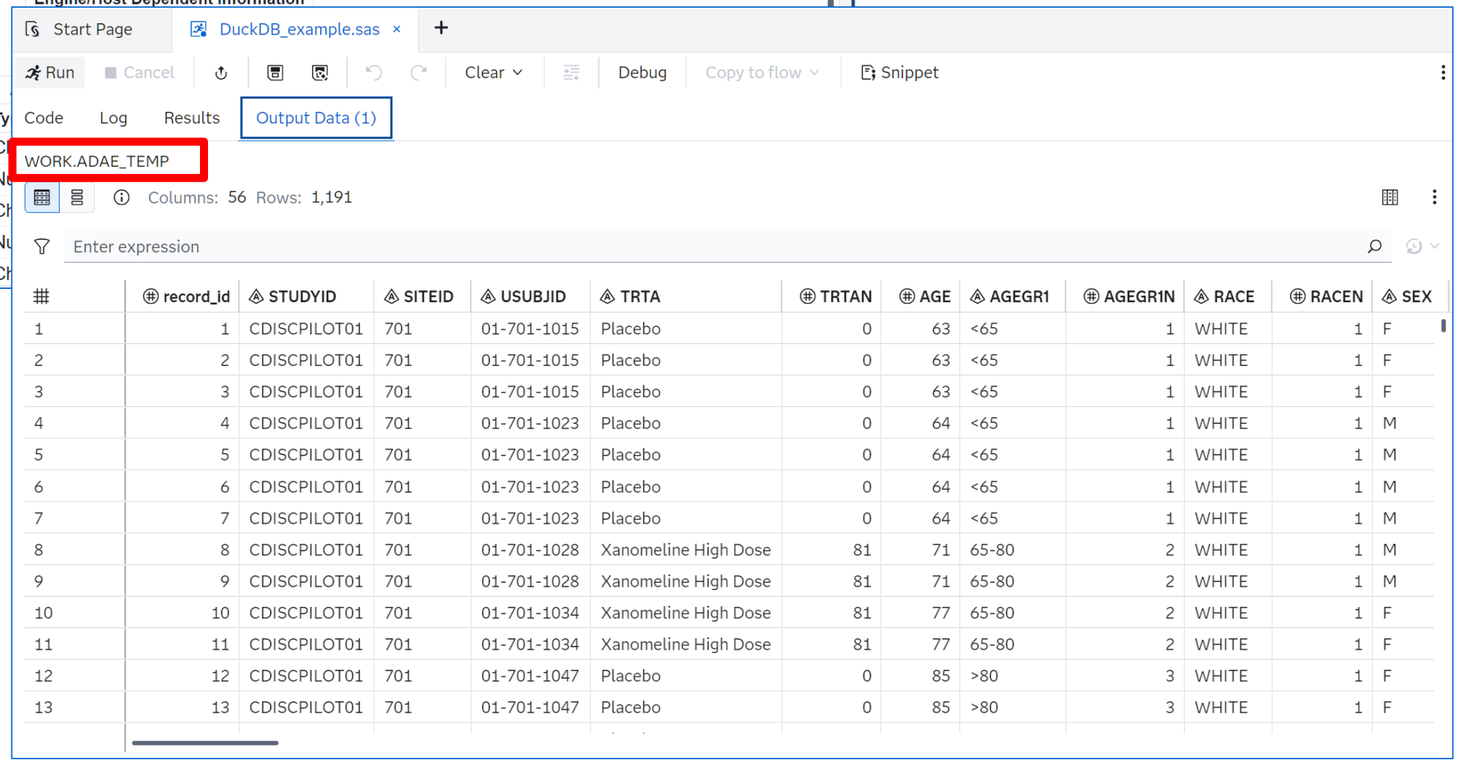
Copyright © SAS Institute Inc. All rights reserved.

Access JSON files



Copyright © SAS Institute Inc. All rights reserved.





	record_id	STUDYID	SITEID	USUBJID	TRTA	TRTAN	AGE	AGEGR1	AGEGR1N	RACE	RACEN	SEX
1	1	CDISCPILOT01	701	01-701-1015	Placebo	0	63	<65	1	WHITE	1	F
2	2	CDISCPILOT01	701	01-701-1015	Placebo	0	63	<65	1	WHITE	1	F
3	3	CDISCPILOT01	701	01-701-1015	Placebo	0	63	<65	1	WHITE	1	F
4	4	CDISCPILOT01	701	01-701-1023	Placebo	0	64	<65	1	WHITE	1	M
5	5	CDISCPILOT01	701	01-701-1023	Placebo	0	64	<65	1	WHITE	1	M
6	6	CDISCPILOT01	701	01-701-1023	Placebo	0	64	<65	1	WHITE	1	M
7	7	CDISCPILOT01	701	01-701-1023	Placebo	0	64	<65	1	WHITE	1	M
8	8	CDISCPILOT01	701	01-701-1028	Xanomeline High Dose	81	71	65-80	2	WHITE	1	M
9	9	CDISCPILOT01	701	01-701-1028	Xanomeline High Dose	81	71	65-80	2	WHITE	1	M
10	10	CDISCPILOT01	701	01-701-1034	Xanomeline High Dose	81	77	65-80	2	WHITE	1	F
11	11	CDISCPILOT01	701	01-701-1034	Xanomeline High Dose	81	77	65-80	2	WHITE	1	F
12	12	CDISCPILOT01	701	01-701-1047	Placebo	0	85	>80	3	WHITE	1	F
13	13	CDISCPILOT01	701	01-701-1047	Placebo	0	85	>80	3	WHITE	1	F

Copyright © SAS Institute Inc. All rights reserved.

Introducing SAS Viya Copilot for Code Assistant

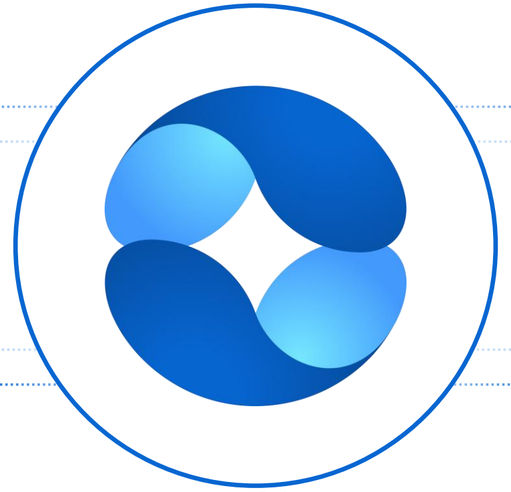
+ SAS Viya MCP Server

Copyright © SAS Institute Inc. All rights reserved.

Generate, iterate, comment and explain SAS code with AI-driven precision

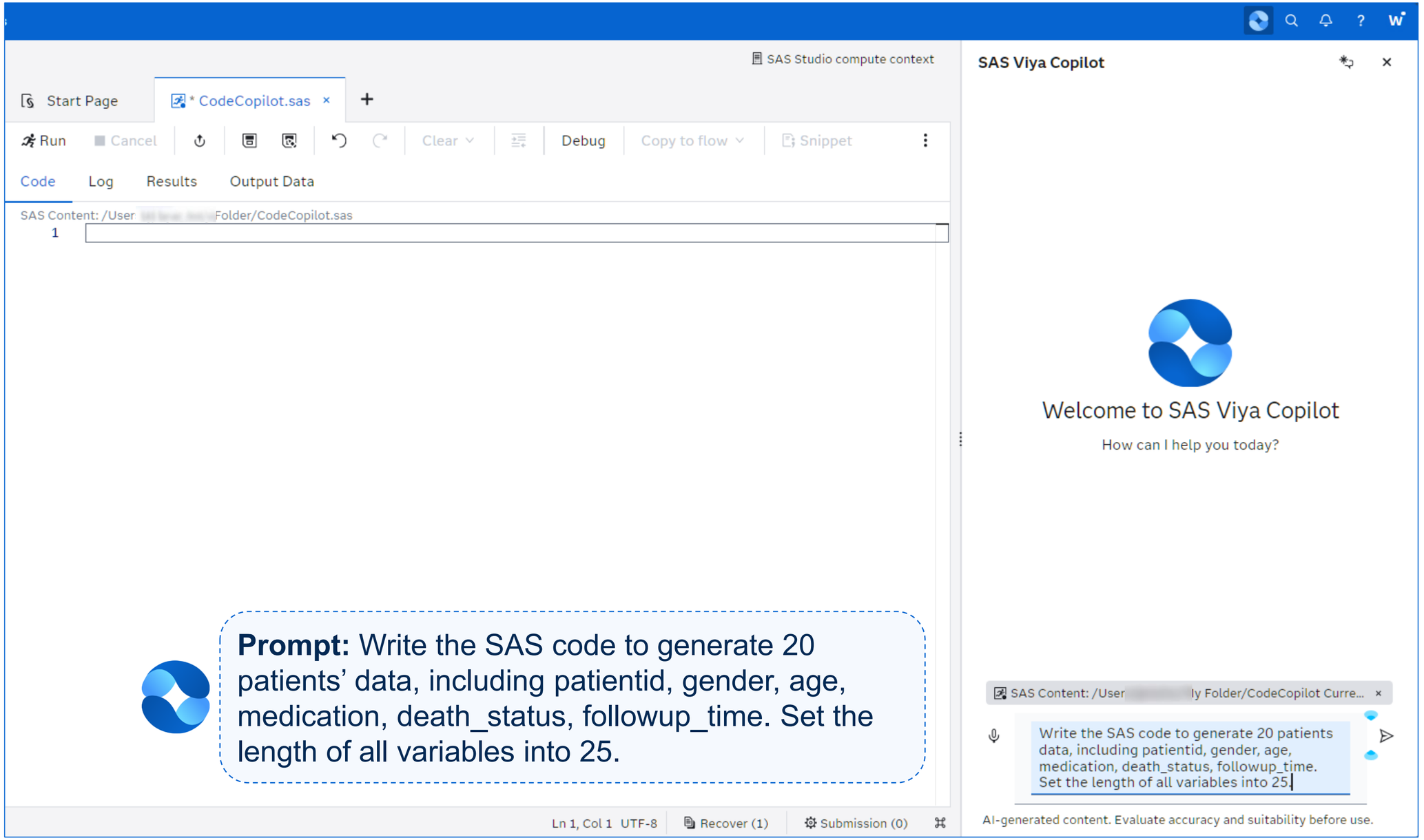
Input → **Output**

	Input	Output	
GENERATE CODE	**Plain English** Prompt written in plain English Inside /* */ + content code where available	**SAS, R or Python Code** SAS, R or Python code in Editor Window	NEW CODE
REFINE AND REFORMAT CODE	**Old Code** Code (may include inline comments)	**Improved Code** Enhanced to your requests and requirements	BETTER CODE
EXAMPLES, EXPLANATIONS, AND COMMENTS	**SAS, R or Python Code** Code (may include inline comments)	**Comments / Explanations** English inline comments in Editor Window	EASY TO UNDERSTAND



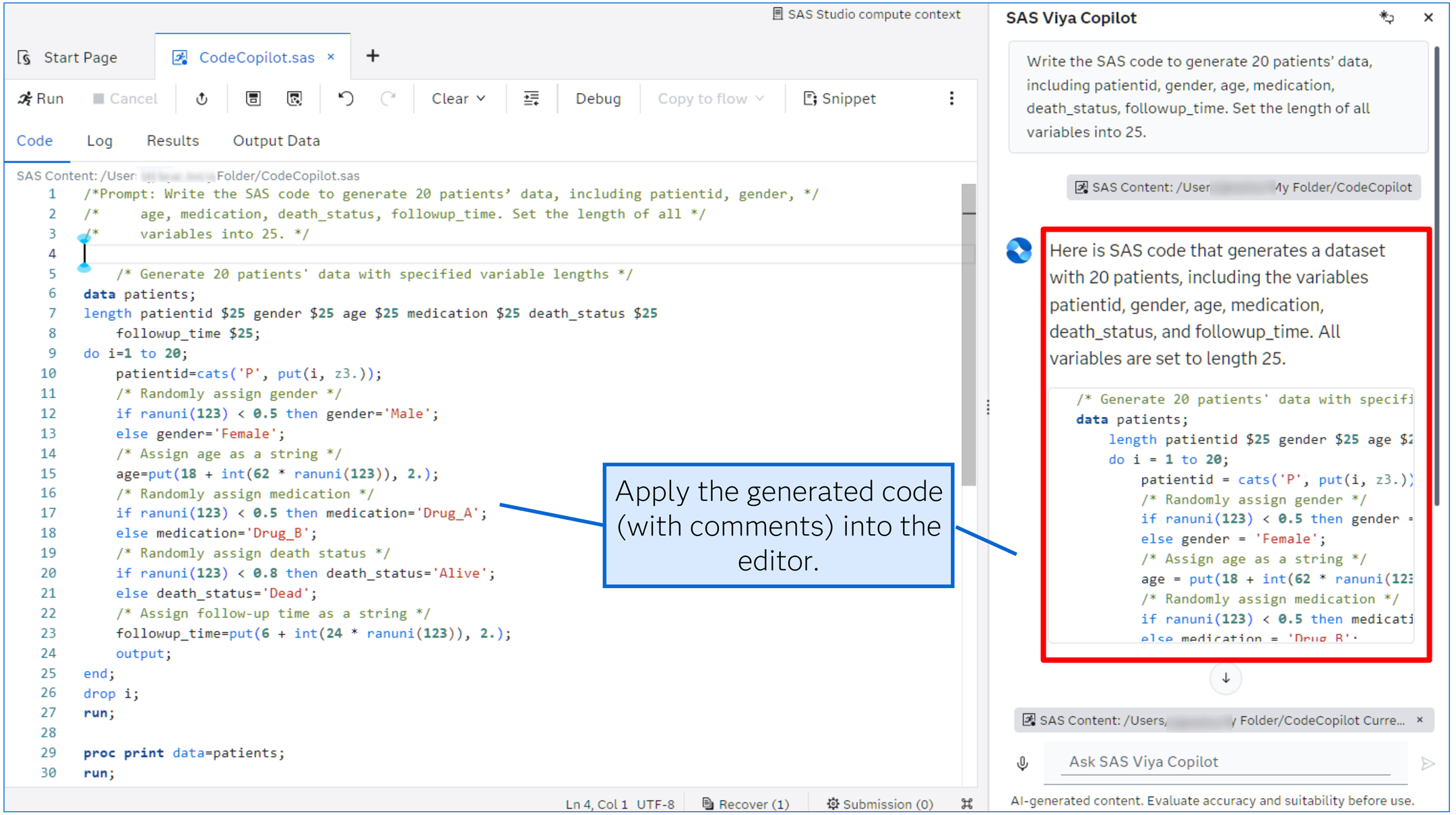
Copyright © SAS Institute Inc. All rights reserved.

Writing code using natural language



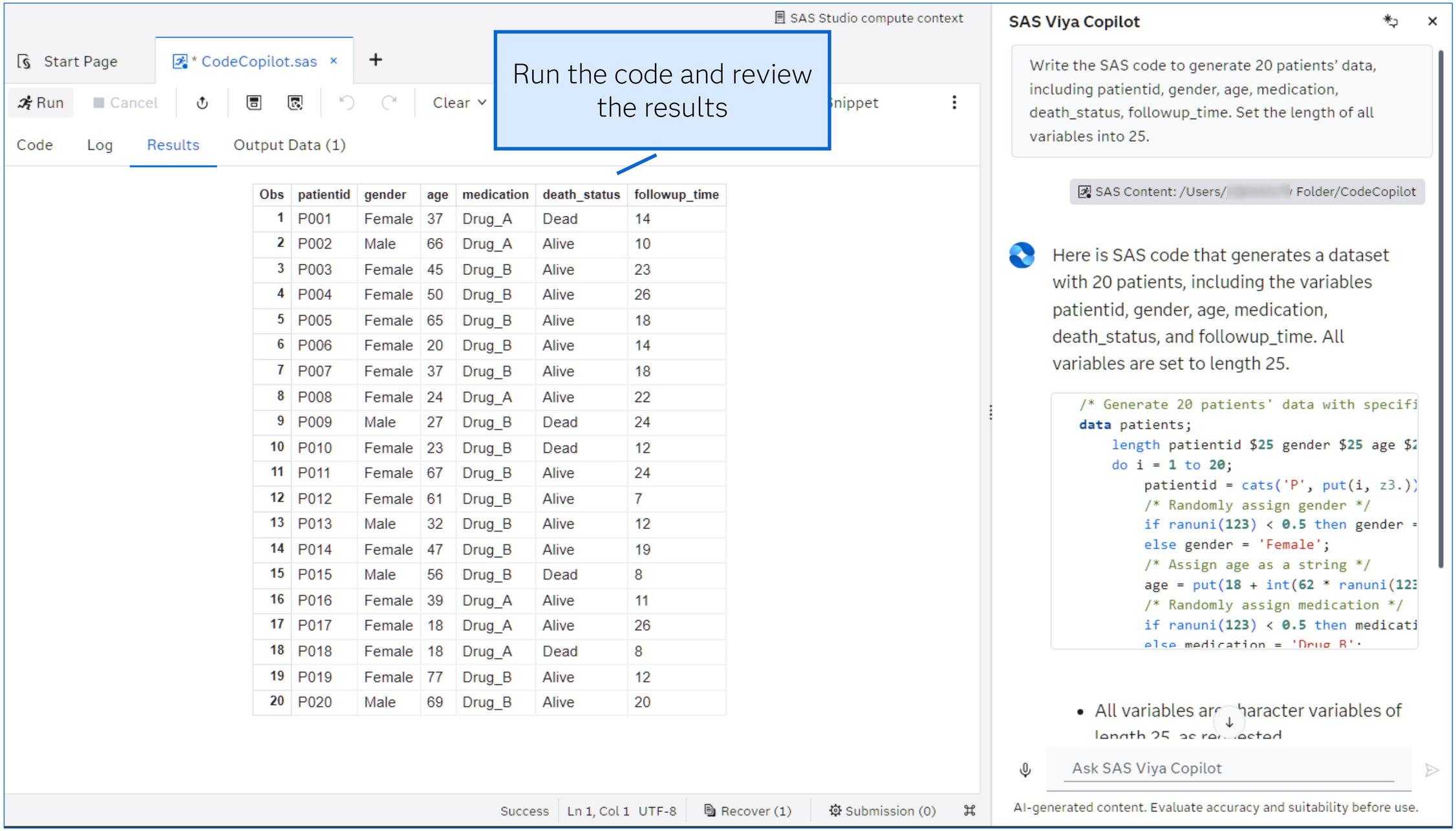
Prompt: Write the SAS code to generate 20 patients' data, including patientid, gender, age, medication, death_status, followup_time. Set the length of all variables into 25.

Copyright © SAS Institute Inc. All rights reserved.



Apply the generated code (with comments) into the editor.

Copyright © SAS Institute Inc. All rights reserved.



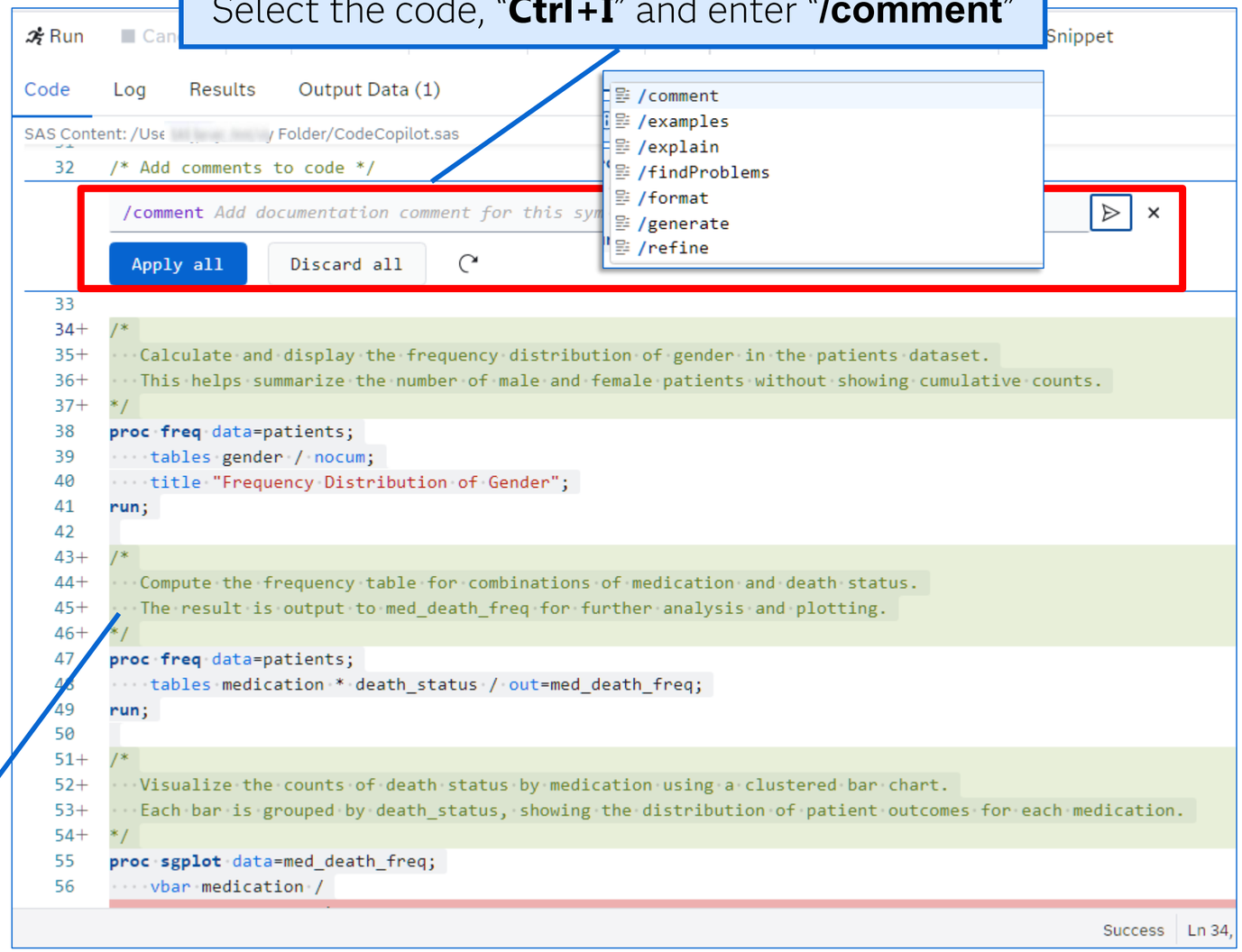
Obs	patientid	gender	age	medication	death_status	followup_time
1	P001	Female	37	Drug_A	Dead	14
2	P002	Male	66	Drug_A	Alive	10
3	P003	Female	45	Drug_B	Alive	23
4	P004	Female	50	Drug_B	Alive	26
5	P005	Female	65	Drug_B	Alive	18
6	P006	Female	20	Drug_B	Alive	14
7	P007	Female	37	Drug_B	Alive	18
8	P008	Female	24	Drug_A	Alive	22
9	P009	Male	27	Drug_B	Dead	24
10	P010	Female	23	Drug_B	Dead	12
11	P011	Female	67	Drug_B	Alive	24
12	P012	Female	61	Drug_B	Alive	7
13	P013	Male	32	Drug_B	Alive	12
14	P014	Female	47	Drug_B	Alive	19
15	P015	Male	56	Drug_B	Dead	8
16	P016	Female	39	Drug_A	Alive	11
17	P017	Female	18	Drug_A	Alive	26
18	P018	Female	18	Drug_A	Dead	8
19	P019	Female	77	Drug_B	Alive	12
20	P020	Male	69	Drug_B	Alive	20

Copyright © SAS Institute Inc. All rights reserved.

Add comments to code automatically

```
/* Add comments to code */

proc freq data=patients;
    tables gender / nocum;
    title "Frequency Distribution of Gender";
run;

proc freq data=patients;
    tables medication * death_status / out=med_death_freq;
run;

proc sgplot data=med_death_freq;
    vbar medication /
        response=count
        group=death_status
        groupdisplay=cluster
        datalabel;

    xaxis label="Medication";
    yaxis label="Number of Patients" integer;
    keylegend / title="Death Status";
    title "Death Status Counts by Medication";
run;
```

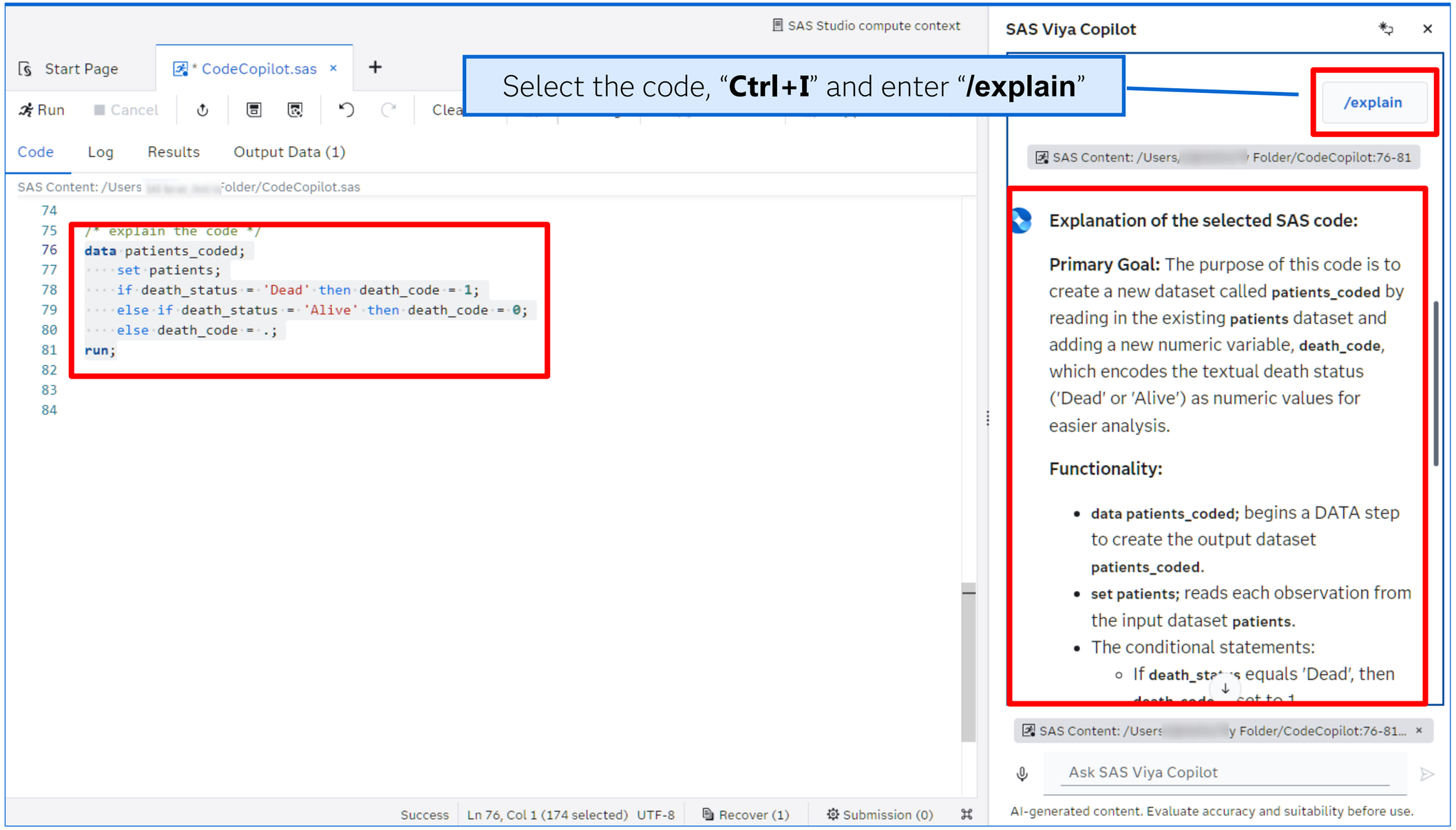
Select the code, "**Ctrl+I**" and enter "**/comment**"



Newly generated comments are highlighted in green

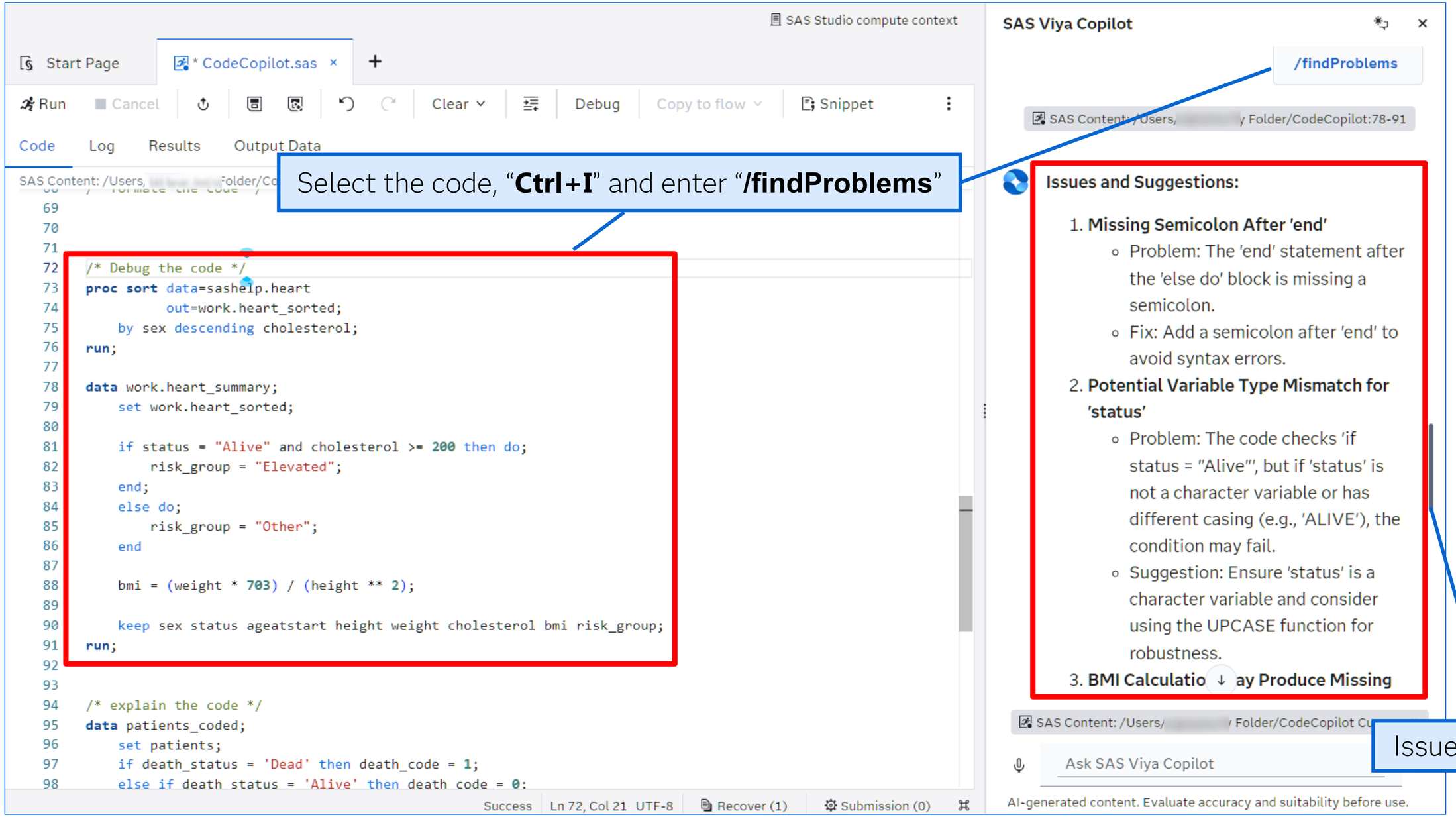
Copyright © SAS Institute Inc. All rights reserved.

Explain the code without adding comments



Copyright © SAS Institute Inc. All rights reserved.

Debug the code



Copyright © SAS Institute Inc. All rights reserved.

SAS Studio compute context

Start Page | * CodeCopilot.sas ×

Run | Cancel | Clear | Debug | Copy to flow | Snippet

Code | Log | Results | Output Data

SAS Content: /Users, Folder/CodeCopilot.sas

```
71
72  /* Debug the code */
73  proc sort data=sashelp.heart
            out=work.heart_sorted;
74+     out=work.heart_sorted;
75      by sex descending cholesterol;
76  run;
77
78  data work.heart_summary;
79      set work.heart_sorted;
80+
81+     length risk_group $10;
82
83      if status = "Alive" and cholesterol >= 200 then do;
84          risk_group = "Elevated";
85      end;
86      else do;
87          risk_group = "Other";
        end
88+     end;
89+
90+     /* Validate height before BMI calculation to avoid division by zero or missing */
91
        bmi = (weight * 703) / (height ** 2);
92+     if height > 0 then bmi = (weight * 703) / (height ** 2);
93+     else bmi = .;
94
95      keep sex status ageatstart height weight cholesterol bmi risk_group;
96  run;
97
```

Fix issues in the code based on the suggestions

Apply all | Discard all

Success | Ln 73, Col 1 (430 selected) UTF-8 | Recover (1) | Submission (0)

SAS Viya Copilot

Please modify the code based on your suggestion above.

SAS Content: /Users, ly Folder/CodeCopilot:73-91

```
proc sort data=sashelp.heart

data work.heart_summary;
    set work.heart_sorted;
    length risk_group $10;

    if status = "Alive" and cholesterol >=
        risk_group = "Elevated";
    end;
    else do;
        risk_group = "Other";
    end;
```

Key improvements applied:

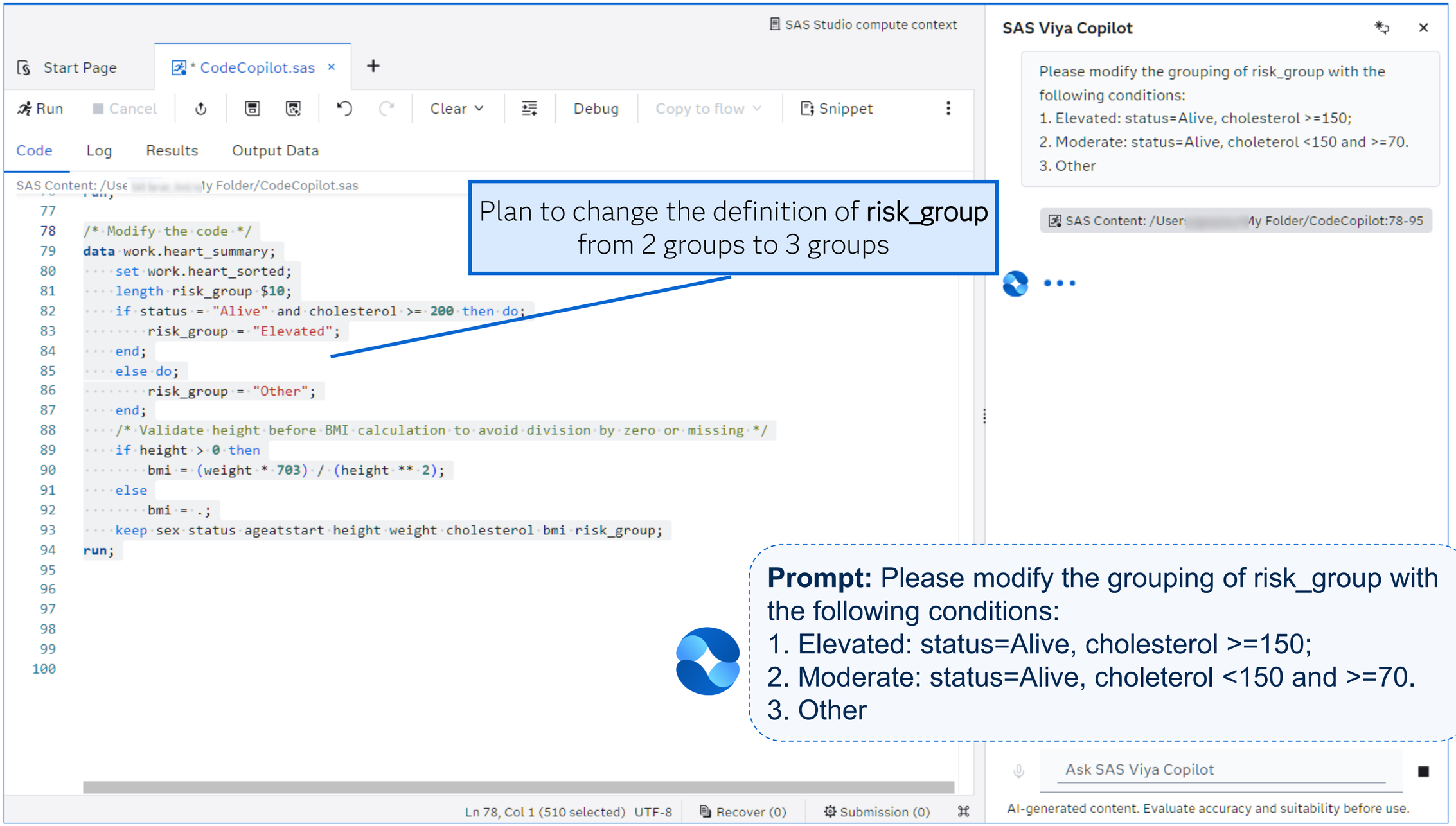
- Added **length risk_group $10;** to explicitly set the character variable's length.
- Added validation to ensure **height > 0** before calculating BMI, preventing division by zero or missing values

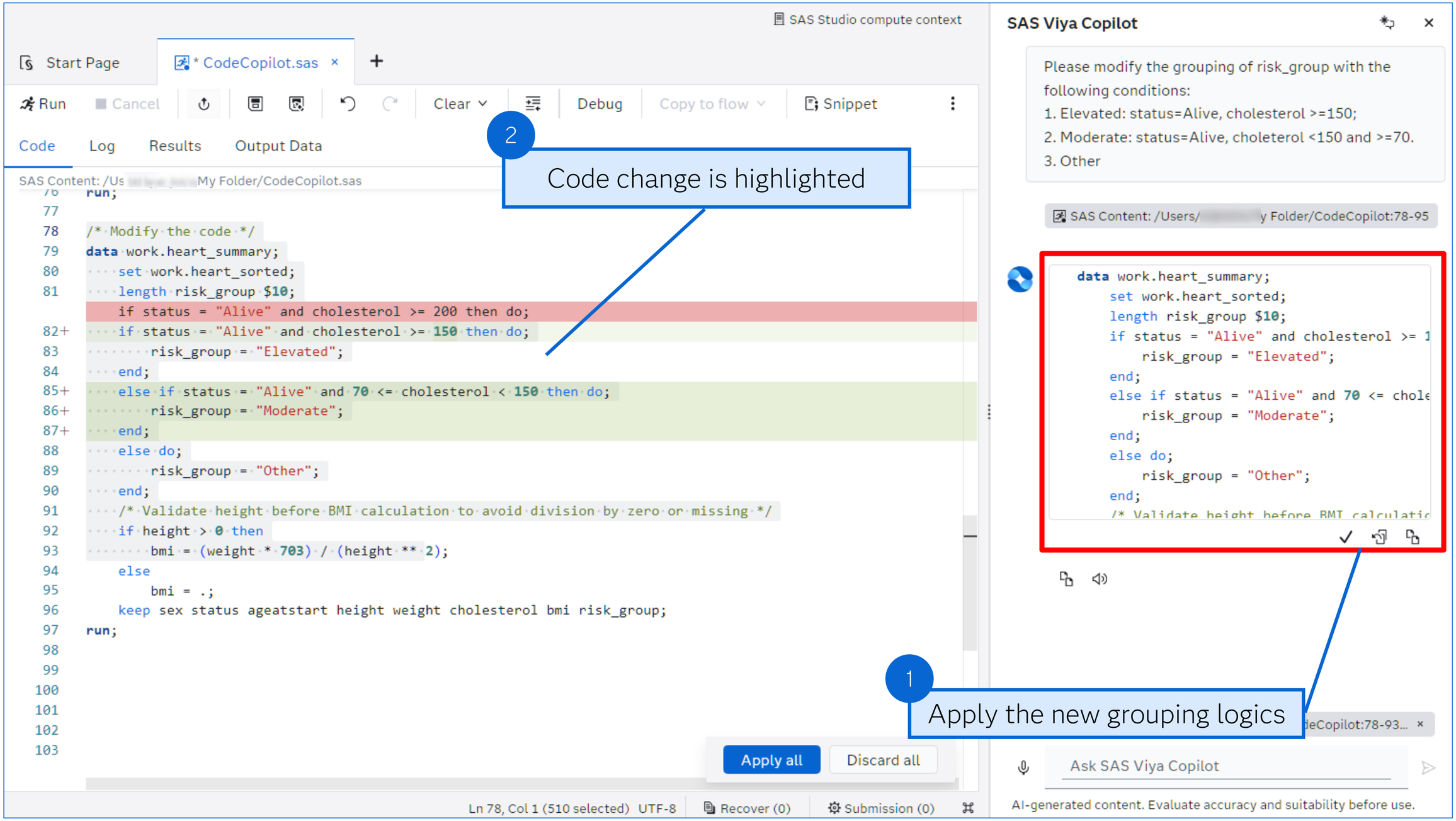
Prompt: Please modify the code based on your suggestion above.

AI-generated content. Evaluate accuracy and suitability before use.

Copyright © SAS Institute Inc. All rights reserved.

Modify the code





Copyright © SAS Institute Inc. All rights reserved.

Format the code for better readability

```
68  /* formate the code */
    /format Format the selected code
    AI-generated content. Evaluate accuracy and suitability before use.
69  proc sort data=sashelp.heart
70  out=work.heart_sorted;
71  by sex descending cholesterol;
72  run;
73  data work.heart_summary;    set work.heart_sorted;
74  length risk_group $10;
75  if status = "Alive" and cholesterol >= 200 then do;
76  risk_group = "Elevated"; end;
77  else do; risk_group = "Other";    end;
78  /* Validate height before BMI calculation to avoid division by zero or missing */
79  if height > 0 then bmi = (weight * 703) / (height ** 2);    else bmi = .;
80  keep sex status ageatstart height weight cholesterol bmi risk_group;
81  run;
```

Before

Select the code, "**Ctrl+I**" and enter "**/format**"

```
68  /* formate the code */
    /format Format the selected code
    Apply all   Discard all
69  proc sort data=sashelp.heart
    out=work.heart_sorted;
    by sex descending cholesterol;
70+     out=work.heart_sorted;
71+     by sex descending cholesterol;
72  run;
    data work.heart_summary;    set work.heart_sorted;
    length risk_group $10;
    if status = "Alive" and cholesterol >= 200 then do;
    risk_group = "Elevated"; end;
    else do; risk_group = "Other";    end;
    /* Validate height before BMI calculation to avoid division by zero or missing */
    if height > 0 then bmi = (weight * 703) / (height ** 2);    else bmi = .;
    keep sex status ageatstart height weight cholesterol bmi risk_group;
73+
74+ data work.heart_summary;
75+     set work.heart_sorted;
76+     length risk_group $10;
77+     if status = "Alive" and cholesterol >= 200 then do;
78+         risk_group = "Elevated";
79+     end;
80+     else do;
81+         risk_group = "Other";
82+     end;
```

```
68  /* formate the code */
69  proc sort data=sashelp.heart
70      out=work.heart_sorted;
71      by sex descending cholesterol;
72  run;
73
74  data work.heart_summary;
75      set work.heart_sorted;
76      length risk_group $10;
77      if status = "Alive" and cholesterol >= 200 then do;
78          risk_group = "Elevated";
79      end;
80      else do;
81          risk_group = "Other";
82      end;
83      /* Validate height before BMI calculation to avoid division by zero or missing */
84      if height > 0 then
85          bmi = (weight * 703) / (height ** 2);
86      else
87          bmi = .;
88      keep sex status ageatstart height weight cholesterol bmi risk_group;
89  run;
```

After

Copyright © SAS Institute Inc. All rights reserved.

MCP – Model Context Protocol

Structuring Memory and Action in Agentic AI Systems

Think of MCP like a USB-C port for AI applications.



USB-C provides a standardized way to connect electronic devices



MCP provides a standardized way to connect AI applications to external systems.

KEY ELEMENTS

Host (interface & controller)

Server (exposes tools/ resources securely)

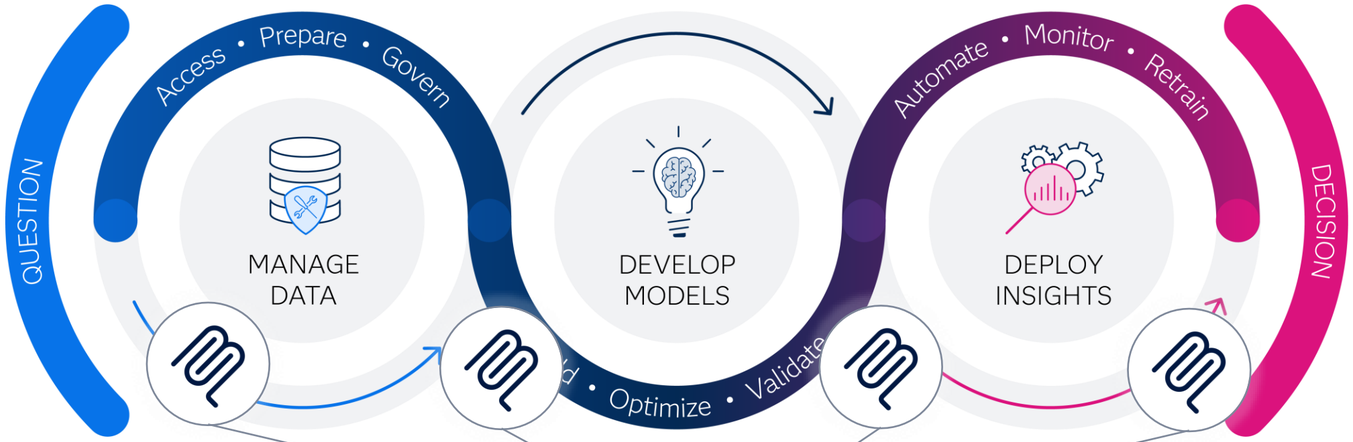
Client (manages messaging over protocol)

OPENNESS

LLM's Use any hosted or Open Source LLM that support Tools

Copyright © SAS Institute Inc. All rights reserved.

Model Context Protocol



SAS Viya MCP Server

https://github.com/sassoftware/sas-mcp-server

Claude

Microsoft Agent Framework

Google Agent Development Kit

VS Code Agent Mode

HOSTED LLM

LLM

OPEN SOURCE LLM

Copyright © SAS Institute Inc. All rights reserved.

SAS® Studio

SAS® Studio - Develop Code and Flows

Options View Open Save All

SAS Server

Name

NFScontent

projects
- Big Data Processing With CAS
- gittest
- LLM Data Mapper
- LLM SAP
- llmragsas
- newdrug
  - code
    - dm_filter.sas
  - data
- RNA_seq_demo
- Sales Forecast Demo
- sasviyaworkshop

dm_filter.sas

Clear

Code

Log

SAS Server: /nfsshare/ ts/newdrug/c...

```
/* Assign library reference to SDTM dat
libname sdtm "/nfsshar oject
/* Filter DM dataset for subjects with
data work.dm_elderly;
    set sdtm.dm;

    /* Apply age filter */
    where age > 65;

    /* Keep relevant variables for anal
    keep studyid usubjid subjid siteid
         armcd arm actarmcd actarm coun
run;

/* Print summary statistics of filtered
proc print data=work.dm_elderly (obs=10
    title "First 10 Subjects with Age >
    title2 "Demographics - Elderly Popu
run;

/* Generate frequency counts by treatme
proc freq data=work.dm_elderly;
    tables arm*sex / nocum nopercent;
    title "Distribution of Elderly Subj
run;

/* Calculate summary statistics for age
proc means data=work.dm_elderly n mean
    var age;
    class arm;
    title "Age Statistics for Elderly S
run;

/* Clear titles */
title;

/* Optional: Export filtered dataset to
/* Uncomment the following lines to sav
```

rver: /nfsshare/ s/newdrug/code/dm_filter.sas

Success Ln 11, Col 1 UTF-8

Claude — Control+Alt+Space

SAS Agent / SDTM domains inventory for clinical study

ow update that SAS code to also sort the filtered DM by AGE and save it as a new file called dm_filter_sorted.sas in the same folder.

Orchestrated SAS code restructuring with sorting and file renaming.

I'll create an updated version of the SAS program that includes sorting by AGE and save it as dm_filter_sorted.sas.

Save code

Orchestrated SAS code modification with sorting functionality.

Perfect! I've created and saved **dm_filter_sorted.sas** to

Key Updates from Original Program:

New Features Added:

1. **PROC SORT Step**: Added sorting by AGE in ascending order after filtering

```sas
proc sort data=work.dm_elderly;
    by age;
run;
```

2. **Additional Report**: Added a second PROC PRINT to display the oldest subjects (last 10 observations)
3. **Updated Documentation**:
   - Modified program name and header
   - Updated titles to reflect sorted output
   - Updated comments to indicate sorting

Program Flow:

How can I help you today?

SAS Agent

Sonnet 4.5

Key Takeaways

Harmonizing SAS, R, and GenAI on one governed platform

- **One platform, not many tools**
  - A unified analytics platform removes data silos, duplicate infrastructure and multiple versions of the truth — driving faster insight, lower risk and greater ROI.
- **SAS + Open Source, together**
  - Run R and SAS code side by side — exchanging datasets, macro variables, functions and graphics — so teams get open-source innovation without losing governance, validation or reproducibility.
- **Reusable, template-driven workflows**
  - Macro-driven R and SAS analytic templates and direct execution of .r program files let teams standardize analyses and scale them across studies.
- **Open file formats, dramatically faster**
  - SAS/ACCESS to DuckDB queries Parquet, JSON, Iceberg and Delta directly — no conversion, no duplicated data.
- **GenAI raises everyday productivity**
  - SAS Viya Copilot generates, explains, comments, debugs and reformats SAS/R code from plain English, while the SAS Viya MCP Server exposes trusted Viya analytics as a standard interface for other AI tools.

Copyright © SAS Institute Inc. All rights reserved.

Reference

- SAS Help Center- Overview- PROC R
- Introducing PROC R: The Newest Way to Integrate R and SAS
- R and SAS Working Seamlessly Together: Unlocking PROC R in Viya for Life Sciences
- SAS + DuckDB Series: What is DuckDB?
- SAS + DuckDB Series: Using the DuckDB Engine
- SAS Help Center- Overview- SAS Viya Copilot
- Getting Started with SAS Viya Copilot for Code Assistance
- SAS Viya MCP Server – GitHub
- Viya MCP: sastool with Claude Desktop
- Connecting GitHub Copilot to SAS Viya with the SAS Viya MCP Server
- Connecting Claude Code CLI to SAS Viya with the SAS Viya MCP Server

Copyright © SAS Institute Inc. All rights reserved.

Thank you
Any question?



William.Kuan@sas.com
SAS Institute Japan

Copyright © SAS Institute Inc. All rights reserved.