

Disclaimer

I'm not a native speaker but I'm doing my best.

If I say something that may be interpreted as either negative or positive (e.g. a joke vs. an offense) I always, always, mean the positive.



SQLinDS and evExpress packages a tribute to my sas rock-stars!

the SAS Users Day

sas innovate
2026

Bartosz Jabłoński

✉ yabwon@gmail.com
[linkedin.com/in/yabwon](https://www.linkedin.com/in/yabwon)

April 27th, 2026
Grapevine, TX



Bart Jabłoński, PhD

yabwon (2022)

SASing since 2009



**WYDZIAŁ
MATEMATYKI I NAUK
INFORMACYJNYCH**

Faculty of
Mathematics
and Information
Science

Warsaw
University
of Technology

www.mini.pw.edu.pl



Takeda
Pharmaceutical
www.takeda.com



Polish SAS
Users Group
www.polsug.com



Rock Stars



Rock Stars



Rock Stars



Rock Stars

Paper 004-2012

Use the Full Power of SAS® in Your Function-Style Macros

Mike Rhoads, Westat, Rockville, MD

ABSTRACT

Function-style macros are macros that can be used within a SAS statement. Although such macros are extremely flexible, they can only use macro language code.

So w
requi
SAS
interf
neces
bene

Paper 032-2013

Submitting SAS® Code On The Side

Rick Langston, SAS Institute Inc., Cary NC

ABSTRACT

This paper explains the new DOSUBL function and how it can submit SAS code to run "on the side" while your DATA step is still running. It also explains how this function differs from invoking CALL EXECUTE or invoking the RUN_COMPILE function of FCMP. Several examples are shown that introduce new ways of writing SAS code.



Problem Statement

Two SAS data sets, LEFT and RIGHT, are provided. They have merging variables X and Y with repetitive BY-group values in both data sets...

Can I use SQL to combine LEFT and RIGHT data inside a DATA step?



Problem Statement - Example

X	A	B	Y	C	D
A	17	42	A	71	24
A	100	101	A	001	010
B	1	0	C	0	1

```
data WANT;  
  set (SELECT L.*, R.*  
        FROM left as L,  
            right as R  
        WHERE L.x = R.y);  
  T=sum(of a-NUMERIC-d);  
  keep _NUMERIC_;  
run;
```



Problem Statement - Example

X	A	B	Y	C	D
A	17	42	A	71	24
A	100	101	A	001	010
B	1	0	C	0	1

A	B	C	D	T
17	42	71	24	154
17	42	001	010	70
100	101	71	24	296
100	101	001	010	212

```
data WANT;
  set (SELECT L.*, R.*
        FROM left as L,
        right as R
        WHERE L.x = R.y);
  T=sum(of a-NUMERIC-d);
  keep _NUMERIC_;
run;
```



Proc sql in data step

Ask Question

Asked 25 days ago · Modified 23 days ago · Viewed 83 times



I would like to ask if it is possible to include a proc sql in data step like this:

```
data mydata;
  set have;

  proc sql;
  create table adrs01 as /*This line I suppose should be removed*/
  select *
  .....;
run;
```



If yes, could you please provide me an example?

Thank you in advance.

sas

asked Feb 27 at 15:25



NewUsr_stat

2,635 ● 5 ● 30 ● 41



Proc sql in data step

[Ask Question](#)

Asked 25 days ago · Modified 23 days ago · Viewed 83 times

I would like to ask if it is possible to include a proc sql in data step like this:

SAS® 9.4 and SAS® Viya® 3.5 Programming Documentation SAS 9.4 / Viya 3.5

DS2 Reference

- What's New
- Introduction
- Getting Started
- DS2 Language Reference
- Appendixes
 - Data Type Reference
 - DS2 Expressions
 - DS2 Example Programs
 - Example: Find Minimums
 - Example: SQL in a DS2 Program**

Example Overview: SQL in a DS2 Program

[Example Overview: SQL in a DS2 Program](#)

[Example Code: SQL in a DS2 Program](#)

[Example Output: SQL in a DS2 Program](#)

Example Overview: SQL in a DS2 Program

Thank you in advance.



asked Feb 27 at 15:25



NewUsr_stat

2,635 ● 5 ● 30 ● 41



Problem Statement

SAS expressions are provided in the form of text strings, say:

"A and B or C",

"A * B + C", or

"42 = A + B + C",

where A, B , and C are variables from some data set HAVE.

How can I evaluate those string expressions using vales from HAVE?



Problem Statement - Example

code	A	B
A + B	17	42
A = B	100	101
A or B	1	0



Problem Statement - Example

code	A	B	result
A + B	17	42	
A = B	100	101	
A or B	1	0	

```
result = "EVALUATE"(code);
```



Problem Statement - Example

code	A	B	result
A + B	17	42	59
A = B	100	101	0
A or B	1	0	1

```
result = "EVALUATE"(code);
```



Problem Statement - Example

code	Pharmaceuticals Industry Applications 263
A + B	Parsing a Compound Boolean Condition to Evaluate Its Clauses Individually and in Logical Groups <i>Daniel M. Himmel, Merck & Co., Inc.</i>
A = B	
A or B	

TE" (code);



Problem Statement - Example

code

A + B

A = B

A or B

Paper SM12

Using the full power of SAS to resolve Algorithmic MedDRA SMQs

Steve Prust, elderbrook solutions GmbH, Biberach an der Riß, Germany

ABSTRACT

This paper presents a method to resolve MedDRA algorithmic SMQs. The method involves using a user-written SAS function that allows data operations that might normally involve several DATA steps or PROCs to be performed in a single macro call (and within a DATA step) as well as hash tables and a little-used SAS MACRO interface. The method has also been extended to include weighted SMQs and flexible contemporaneous requirements and is a comprehensive solution to current algorithmic SMQ definitions. The resultant code demonstrates how diverse elements of the SAS language can be combined to produce powerful solutions.

E" (code) ;



Problem Statement - Example

[Yet another reason to love hash tables](#)
Bart Jablonski <syabwon@GMAIL.COM>
Thu, 4 Jul 2019 16:47:41 -0400
[text/plain](#) (138 lines)

Hi SAS-Lers,

I had this cool discussion at the office today and the result was very interesting (at lest for me) code which I like to share.

Idea was as old as programming: you have a dataset which contain both data and selection conditions (i.e. if a condition is satisfied than output an observation). I know that keeping both data and logic in the same place is not very "good practice" but that was the setup.



Are there SAS packages
for this job?

Of course there are!

The SQLinDS and
the evExpress
packages...





the way to share

`github.com/yabwon/SAS_PACKAGES`





the way to share

`github.com/SASPAC/sqlinds`

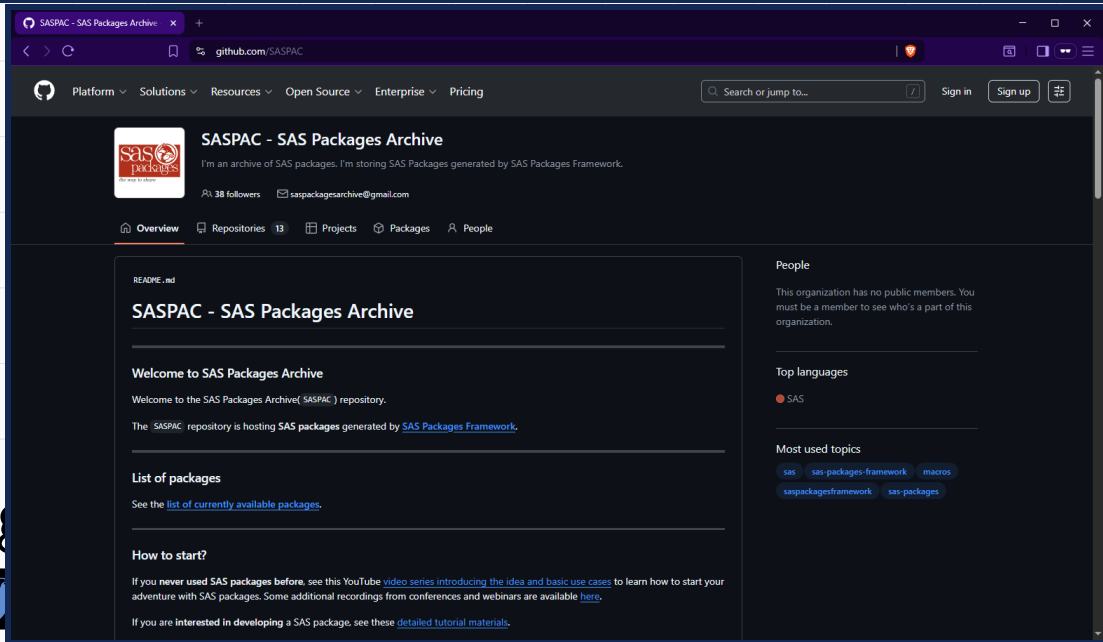




the way to share

`github.com/SASPAC/evexpress`





The screenshot shows the GitHub repository page for 'SASPAC - SAS Packages Archive'. The page is in dark mode. At the top, there is a navigation bar with links for Platform, Solutions, Resources, Open Source, Enterprise, and Pricing. A search bar and 'Sign in'/'Sign up' buttons are also present. The repository header includes the organization name, a description: 'I'm an archive of SAS packages. I'm storing SAS Packages generated by SAS Packages Framework.', and contact information: '38 followers' and 'saspackagesarchive@gmail.com'. Below the header, there are tabs for Overview, Repositories (13), Projects, Packages, and People. The main content area displays the README.md file, which contains the following text:

```
README.md
```

SASPAC - SAS Packages Archive

Welcome to SAS Packages Archive

Welcome to the SAS Packages Archive(`SASPAC`) repository.

The `SASPAC` repository is hosting `SAS packages` generated by [SAS Packages Framework](#).

List of packages

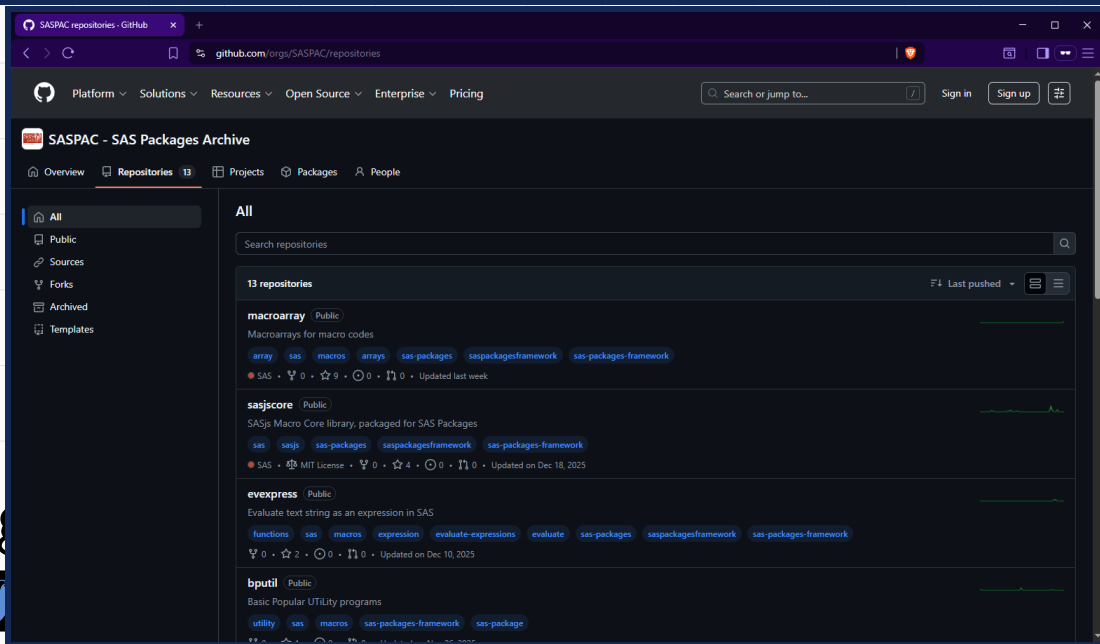
See the [list of currently available packages](#).

How to start?

If you **never used SAS packages before**, see this [YouTube video series introducing the idea and basic use cases](#) to learn how to start your adventure with SAS packages. Some additional recordings from conferences and webinars are available [here](#).

If you are interested in **developing a SAS package**, see these [detailed tutorial materials](#).

On the right side of the repository page, there is a 'People' section stating: 'This organization has no public members. You must be a member to see who's a part of this organization.' Below that is a 'Top languages' section with a single entry: 'SAS'. At the bottom, there is a 'Most used topics' section with tags for 'sas', 'sas-packages-framework', 'macros', 'saspackagesframework', and 'sas-packages'.



The screenshot shows the GitHub interface for the SASPAC repository. The browser address bar shows `github.com/orgs/SASPAC/repositories`. The page title is "SASPAC - SAS Packages Archive". The navigation menu includes "Overview", "Repositories" (with a count of 13), "Projects", "Packages", and "People". A search bar is present with the text "Search or jump to...".

On the left sidebar, there are filters for "All", "Public", "Sources", "Forks", "Archived", and "Templates".

The main content area is titled "All" and contains a search bar for repositories. Below it, there are 13 repositories listed, sorted by "Last pushed". The visible repositories are:

- macroarray** (Public): Macroarrays for macro codes. Tags: `array`, `sas`, `macros`, `arrays`, `sas-packages`, `saspackagesframework`, `sas-packages-framework`. Stats: 0 forks, 9 stars, 0 issues, 0 pull requests. Updated last week.
- sasjscore** (Public): SASjs Macro Core library, packaged for SAS Packages. Tags: `sas`, `sasjs`, `sas-packages`, `saspackagesframework`, `sas-packages-framework`. License: MIT License. Stats: 0 forks, 4 stars, 0 issues, 0 pull requests. Updated on Dec 18, 2025.
- evexpress** (Public): Evaluate text string as an expression in SAS. Tags: `functions`, `sas`, `macros`, `expression`, `evaluate-expressions`, `evaluate`, `sas-packages`, `saspackagesframework`, `sas-packages-framework`. Stats: 0 forks, 2 stars, 0 issues, 0 pull requests. Updated on Dec 10, 2025.
- bputil** (Public): Basic Popular UTILITY programs. Tags: `utility`, `sas`, `macros`, `sas-packages-framework`, `sas-package`.



The screenshot shows the GitHub repository page for `SASPAC/evexpress`. The repository is public and has 2 stars and 0 forks. The main branch is `main`, with 1 branch and 2 tags. The repository contains a `hist` directory and files `README.md`, `evexpress.md`, and `evexpress.zip`. The `README` file is expanded, showing the title `evExpress - Evaluate text string as an expression in SAS`. The description states: "The `evExpress` package is a bunch of macros and functions design to help evaluate SAS expressions that are provided in a form of a text string." It also provides the SHA256 digest for the latest version: `F*19CC8C0C14CEBD6962F96FA114430A5044ABD02972DFAAE3F5BE1878FDBEA221` and a link to the [Documentation for evExpress](#). The right sidebar shows the repository's activity, including 2 stars, 0 watching, 0 forks, and 1 release. The latest release is titled "The evExpress package, version: ..." and was published on Dec 9, 2025.

GitHub - SASPAC/evexpress: Ev... x +

github.com/SASPAC/evexpress

Platform Solutions Resources Open Source Enterprise Pricing

SASPAC / evexpress Public

Notifications Fork 0 Star 2

Code Issues Pull requests Actions Projects Security Insights

main 1 Branch 2 Tags

Go to file Code

yabwon Update README.md 3353798 · last month 4 Commits

hist	Package: evExpress, Version: 0.0.2	last month
README.md	Update README.md	last month
evexpress.md	Package: evExpress, Version: 0.0.2	last month
evexpress.zip	Package: evExpress, Version: 0.0.2	last month

README

evExpress - Evaluate text string as an expression in SAS

The `evExpress` package is a bunch of macros and functions design to help evaluate SAS expressions that are provided in a form of a text string.

SHA256 digest for the latest version of `evExpress` :
F*19CC8C0C14CEBD6962F96FA114430A5044ABD02972DFAAE3F5BE1878FDBEA221

[Documentation for evExpress](#)

About

Evaluate text string as an expression in SAS

functions sas macros expression

evaluate-expressions evaluate

sas-packages saspackagesframework

sas-packages-framework

Readme

Activity

Custom properties

2 stars

0 watching

0 forks

Report repository

Releases 2

The evExpress package, version: ... Latest

on Dec 9, 2025

+ 1 release



The Code

```
%loadPackage(SQLinDS)
```

X	A	B	X	C	D
A	1	2	A	10	20
A	3	4	B	30	40
B	5	6	B	50	60
B	7	8	C	70	80

```
data want;  
set %SQL(  
    SELECT * FROM left  
    NATURAL FULL JOIN right  
);  
T=sum(of _NUMERIC_);  
keep X _NUMERIC_;  
run;
```



The Code

```
%loadPackage(SQLinDS)
```

X	A	B	X	C	D
A	1	2	A	10	20
A	3	4	B	30	40
B	5	6	B	50	60
B	7	8	C	70	80

```
data want;
set %SQL(
  SELECT * FROM left
  NATURAL FULL JOIN right
);
T=sum(of _NUMERIC_);
keep X _NUMERIC_;
run;
```

Data set: WORK.WANT

Obs	X	C	D	A	B	T
1	A	10	20	1	2	33
2	A	10	20	3	4	37
3	B	50	60	5	6	121
4	B	30	40	5	6	81
5	B	50	60	7	8	125
6	B	30	40	7	8	85
7	C	70	80	.	.	150



The Code

```
1 data want;
2 set %SQL(
3     SELECT * FROM left
4     NATURAL FULL JOIN right
5 );
NOTE:*** the query ***
      "SELECT * FROM left NATURAL FULL JOIN right"
      *****
*** executed as ***
>     select COALESCE(RIGHT.X, LEFT.X) as X, RIGHT.C, RIGHT.D, LEFT.A, LEFT.B
>     from WORK.LEFT full outer join WORK.RIGHT on RIGHT.X = LEFT.X;
>
*****
6     T=sum(of _NUMERIC_);
7     keep X _NUMERIC_;
8 run;

NOTE: There were 4 observations read from the data set WORK.LEFT.
NOTE: There were 4 observations read from the data set WORK.RIGHT.
NOTE: There were 7 observations read from the data set DSSQL.DSSQLTMPVIEW32.
NOTE: The data set WORK.WANT has 7 observations and 6 variables.
NOTE: DATA statement used (Total process time):
      real time           0.03 seconds
      user cpu time       0.01 seconds
      system cpu time     0.01 seconds
      memory              6773.87k
      OS Memory          33540.00k
```



The Code

```
%loadPackage(evExpress)
```

code	X	Y	C
(x > 1) and c="A"	3	11	A
x < 1 and (9 < y < 13)	-3	17	B
x < 1 or (81 < y*y)	3	11	C
(3 - sqrt(x+y))	5	11	D
12 + sum(of x--y)	3	9	E
sin(x)**2 + cos(x)**2	-3	11	F

```
%evExpressDS(have,exp=code,want=work.want)
```



The Code

```
%loadPackage(evExpress)
```

```
code
(x > 1) and c="A"
x < 1 and (9 < y < 13)
x < 1 or (81 < y*y)
(3 - sqrt(x+y))
12 + sum(of x--y)
sin(x)**2 + cos(x)**2
```

```
%evExpressDS(have, exp=code)
```

Data set: WORK.WANT

Obs	code	x	y	c	errorCheck	result
1	(x > 1) and c="A"	3	11	A	0	1
2	x < 1 and (9 < y < 13)	-3	17	B	0	0
3	x < 1 or (81 < y*y)	3	11	C	0	1
4	(3 - sqrt(x+y))	5	11	D	0	-1
5	12 + sum(of x--y)	3	9	E	0	24
6	sin(x)**2 + cos(x)**2	-3	11	F	0	1



The Code

```
%loadPackage(evExpress)
```

code	X	Y	C
(x > 1) and c="A"	3	11	A
x < 1 and (9 < y < 13)	-3	17	B
x < 1 or (81 < y*y)	3	11	C
(3 - sqrt(x+y))	5	11	D
12 + sum(of x--y)	3	9	E
sin(x)**2 + cos(x)**2	-3	11	F

```
data want;
```

```
set have
```

```
  curobs=curobs
```

```
  indiname=indiname;
```

```
  value = evExpress(code, indiname, curobs);
```

```
run;
```



The Code

```
%loadPackage(evExpress)
```

```
code
(x > 1) and c="A"
x < 1 and (9 < y < 13)
x < 1 or (81 < y*y)
(3 - sqrt(x+y))
12 + sum(of x--y)
sin(x)**2 + cos(x)**2
```

```
data want;
set have
curobs=curobs
indsname=indsname;
```

```
value = evExpress(code, indsname, curobs);
run;
```

Data set: WORK.WANT

Obs	code	x	y	c	value
1	(x > 1) and c="A"	3	11	A	1
2	x < 1 and (9 < y < 13)	-3	17	B	0
3	x < 1 or (81 < y*y)	3	11	C	1
4	(3 - sqrt(x+y))	5	11	D	-1
5	12 + sum(of x--y)	3	9	E	24
6	sin(x)**2 + cos(x)**2	-3	11	F	1



The Code

```
%loadPackage(evExpress)
```

```
code
```

```
1 data want2;  
2   set have2  
3   curobs=curobs  
4   indcname=indcname;  
5  
6   value = evExpress(code, indcname, curobs);  
7   run;
```

```
NOTE: There were 6000 observations read from the data set WORK.HAVE2.
```

```
NOTE: The data set WORK.WANT2 has 6000 observations and 5 variables.
```

```
NOTE: DATA statement used (Total process time):
```

real time	18.85 seconds
user cpu time	8.73 seconds
system cpu time	9.48 seconds
memory	2925.09k
OS Memory	26876.00k

```
value = evExpress(code, indcname, curobs);  
run;
```



- Bartosz Jabłoński, *"SAS packages - the way to share (a how to) - extended"*, SAS GF 2020 Proceedings (updated version as of November 2025), https://github.com/yabwon/SAS_PACKAGES/tree/master/SPF/Documentation
- Bartosz Jabłoński, *evExpress packages documentation*, (as of January 2026), <https://github.com/SASPAC/evexpress>
- Mike Rhoads, *"Use the Full Power of SAS in Your Function-Style Macros"*, SAS Global Forum 2012 Proceedings, 004-2012, <https://support.sas.com/resources/papers/proceedings12/004-2012.pdf>
- Rick Langston, *"Submitting SAS Code On The Side"*, SAS Global Forum 2013 Proceedings, 032-2013, <https://support.sas.com/resources/papers/proceedings13/032-2013.pdf>
- Quentin McMullen, *"A Close Look at How DOSUBL Handles Macro Variable Scope"*, SAS Global Forum 2020 Proceedings, 4958-2020, <https://support.sas.com/resources/papers/proceedings13/032-2013.pdf>
- Steve Prust, *"Using the full power of SAS to resolve Algorithmic MedDRA SMQs"*, PHUSE EU 2025 Proceedings, SM12, https://www.lexjansen.com/phuse/2025/SM/PAP_SM12.pdf
- Daniel M. Himmel, *"Parsing a Compound Boolean Condition to Evaluate Its Clauses Individually and in Logical Groups"*, NESUG 1990 Proceedings, <https://www.lexjansen.com/nesug/nesug90/NESUG90042.pdf>
- Bartosz Jabłoński et al., *"Yet another reason to love hash tables"*, SAS-L discussion group archive, 2019-07-04, <https://listserv.uga.edu/scripts/wa-UGA.exe?A2=ind1907A&L=SAS-L&P=R18571>



dziękuję bardzo

