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

April 27th, 2026
Grapevine, TX

Bart Jabłoński, PhD

yabwon (2022)

SASing since 2009

Faculty of Mathematics and Information Science

Warsaw University of Technology

www.mini.pw.edu.pl

Takeda

Pharmaceutical

www.takeda.com

Polish SAS Users Group

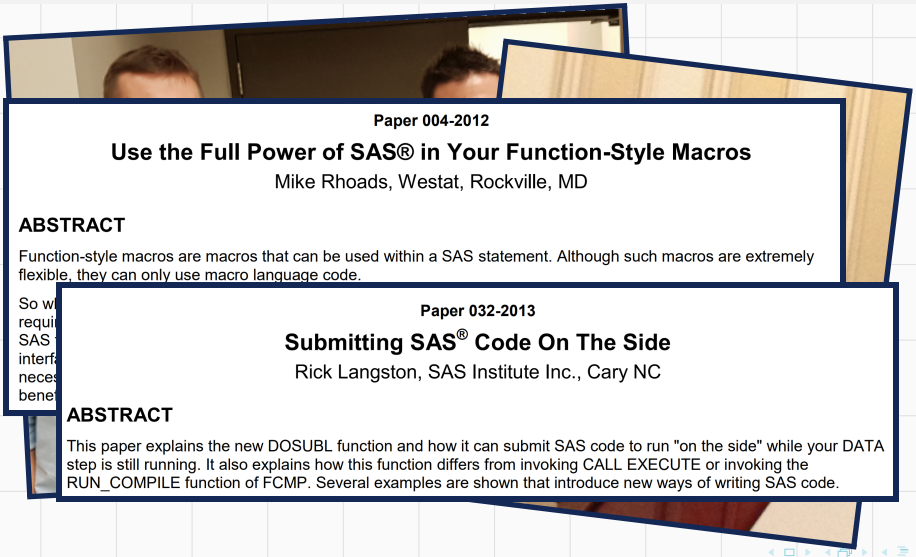
www.polsug.com

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 wh
requi
SAS
interfa
neces
benef

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
A	17	42
A	100	101
B	1	0

Y	C	D
A	71	24
A	001	010
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;
```

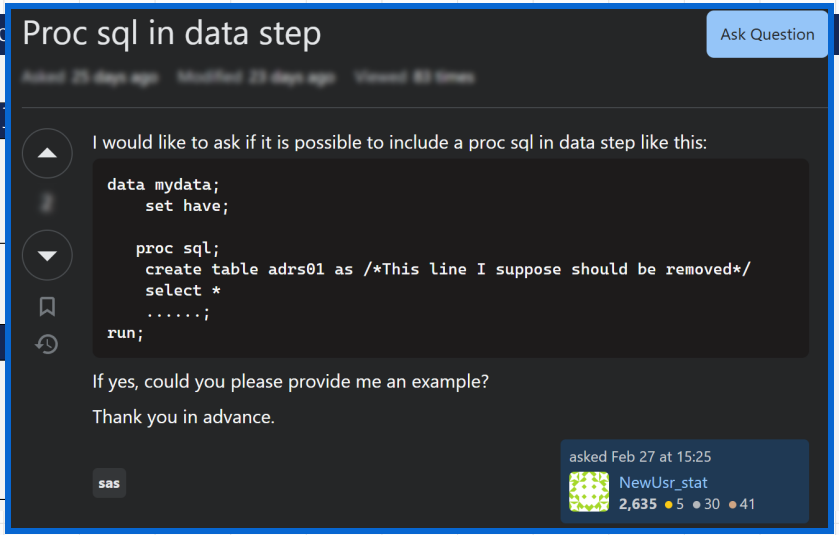
Problem Statement - Example

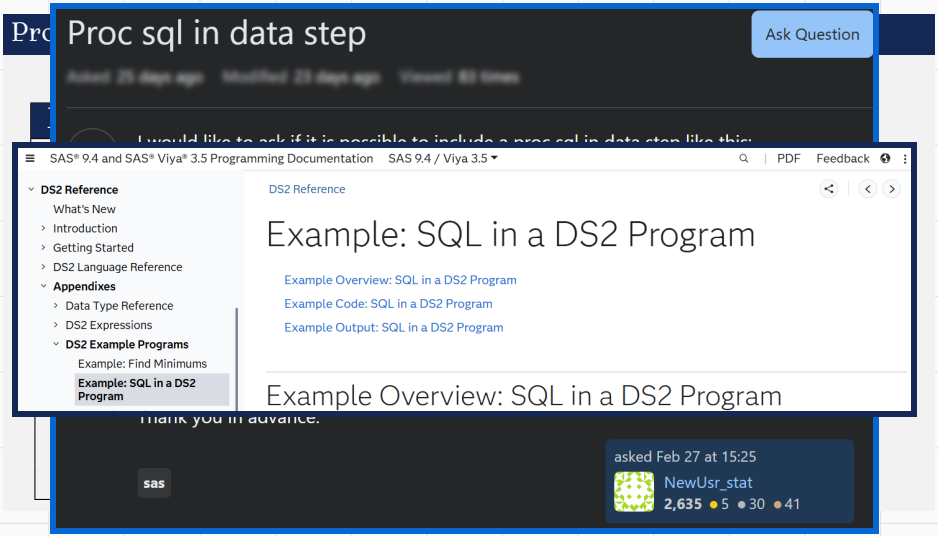
X	A	B
A	17	42
A	100	101
B	1	0

Y	C	D
A	71	24
A	001	010
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;
```

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


Proc sql in data step
Ask Question
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
SAS® 9.4 and SAS® Viya® 3.5 Programming Documentation
SAS 9.4 / Viya 3.5
PDF
Feedback
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
DS2 Reference
Example: 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
asked Feb 27 at 15:25
NewUsr_stat
2,635 5 30 41
sas

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);
```

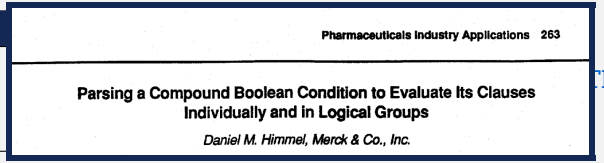
Problem Statement - Example

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
A + B
A = B
A or B



Pharmaceuticals Industry Applications 263

Parsing a Compound Boolean Condition to Evaluate Its Clauses Individually and in Logical Groups

Daniel M. Himmel, Merck & Co., Inc.

```
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 <yabwon@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!

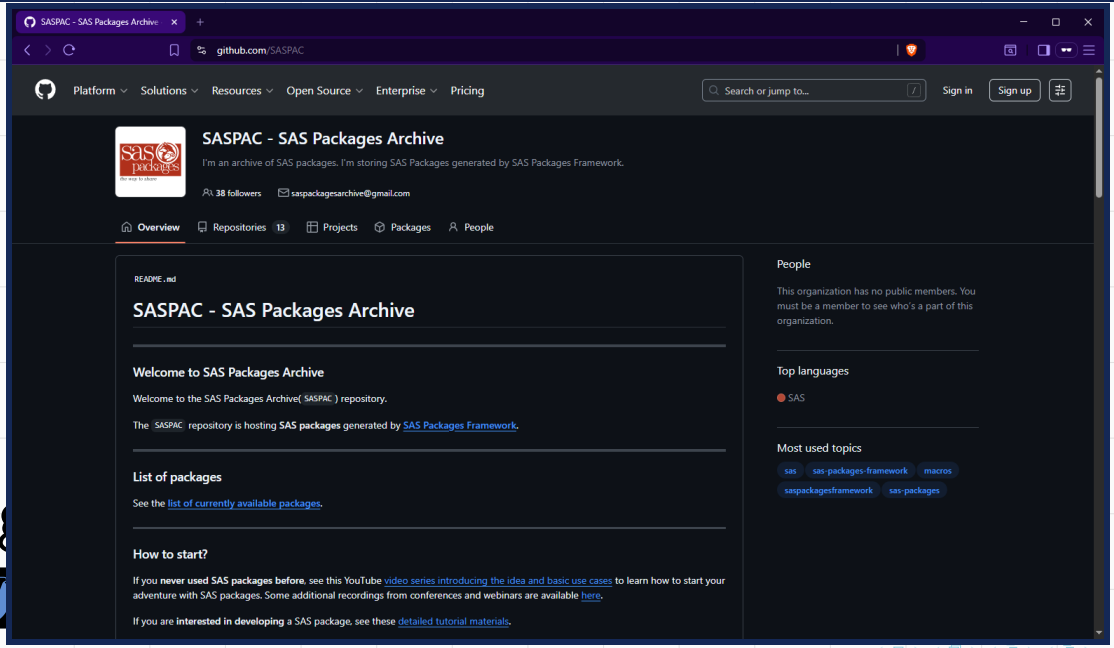
The `SQLinDS` and the `evExpress` packages...

github.com/yabwon/SAS_PACKAGES

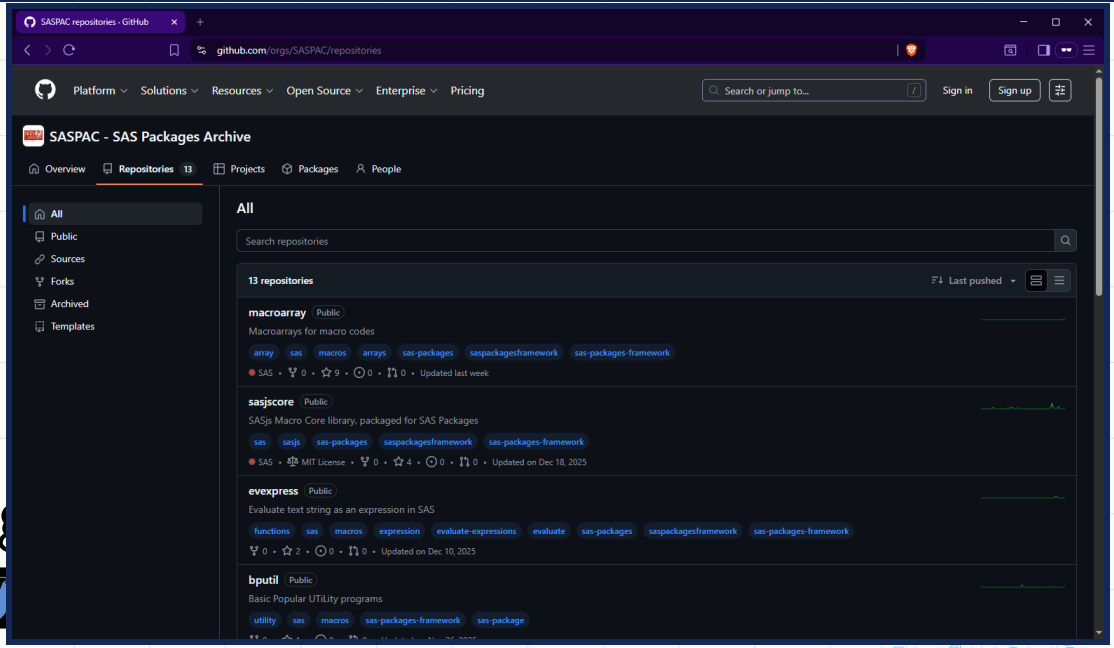
github.com/SASPAC/sqlinds

github.com/SASPAC/evexpress

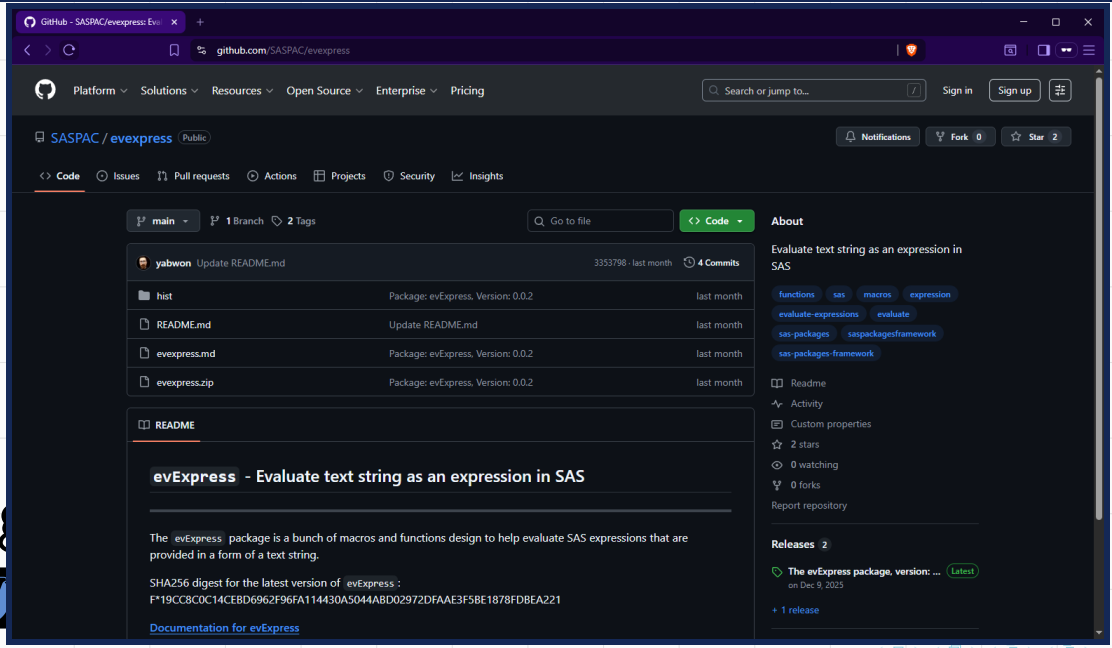
Gdzie szukać pakietu?



Gdzie szukać pakietu?



Gdzie szukać pakietu?



The Code

```
%loadPackage(SQLinDS)
```

X	A	B
A	1	2
A	3	4
B	5	6
B	7	8

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

X	C	D
A	10	20
B	30	40
B	50	60
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=cod
```

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
    indsname=indsname;

  value = evExpress(code, indsname, 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         indsname=indsname;
5
6       value = evExpress(code, indsname, 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, indsname, curobs);
run;
```

Źródła

- 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