SAS® Debugging 101

A presentation by

Kirk Paul Lafler
@sasNerd

Presentation Topics

- Types of Errors and Warnings
- Errors SAS Can't Help With
- Debugging Strategies and Techniques
- Macro Debugging Strategies and Techniques
- Debugging with Benchworkzz Software

Types of Errors and Warnings

Error Types

Type of Error	Description
Syntax	The rules associated with the use of a programming statement in the SAS language are violated. Examples include misspelling of dataset or variable names, and forgetting a semicolon.
Semantic	An element in a statement is specified incorrectly preventing SAS from knowing how to interpret your code. Examples include misspelling a variable name and incorrectly specifying an array's elements.
Execution-time	An error that produces a Note or Warning on the SAS Log, but allows the program to continue. Examples include observations arranged in incorrect BY-group order.
Data	An error that is generated when one or more data values do not match an INPUT statement spec.
Macro-Related	An error that occurs during macro compilation or execution. An example includes a local macro variable that should have been defined as a global macro variable.
Logic	An error that does not have a Note, Warning or Error associated with it, but contains erroneous or unexpected results. Examples include using a "<" when a ">" comparison operator should have been used.

Compilation vs. Execution Errors

Compilation Phase Errors	Execution Phase Errors
Syntax	Execution-time
Semantic	Data
Macro-Related	Macro-Related

Errors SAS Can't Help With

Errors That SAS Can't Help With

Logic errors can be difficult to locate in SAS because it's up to you to find them!

Debugging Strategies and Techniques

Examining SAS Option Settings

SAS Option settings can be examined by:

- **Using the PROC OPTIONS statement**

```
PROC OPTIONS;
RUN;
```

- **Viewing the OPTIONS window**
- **Accessing the contents of DICTIONARY.OPTIONS using SQL**

```
PROC SQL;
  SELECT  *  FROM DICTIONARY.OPTIONS;
QUIT;
```

- **Accessing the virtual table SASHELP.VOPTION**

```
PROC PRINT DATA=SASHELP.VOPTION;
RUN;
```

Debugging with SAS Options

SAS Options allow you to request:

- **Processing information to be displayed in the SAS Log**
- **Running programs with specific observations**
- **Stopping a program when certain errors are encountered**

DSNFERR/NODSNFERR Option

Determine whether processing should stop when a reference to a data set does not exist.

System Option

```
options  DSNFERR;
data processed_movies;
  set mydata.movies;
  <  other SAS statements  >;
run;
```

ERRORABEND/NOERRORABEND

Rather than issuing an error message, tell SAS to abnormally terminate for many errors including syntax errors.

System Option

```
options  ERRORABEND;
data processed_movies;
   set mydata.movies;
   <  other SAS statements  >;
run;
```

ERRORS= n (default is 20)

Tells SAS the maximum number of observations to print complete error messages.

System Option

```
options  ERROR=100;
data processed_movies;
   set mydata.movies;
   <  other SAS statements  >;
run;
```

FMTERR / NOFMTERR

Tells SAS to produce an error message when it can not find a format.

System Option

```
options  FMTERR;
data processed_movies;
   set mydata.movies;
   <  other SAS statements  >;
run;
```

MSGLEVEL=N / I

Tells SAS to print notes, warnings, errors and informational messages for merge, index and sort usage.

System Option

```
options  MSGLEVEL=I;
data processed_movies;
   set mydata.movies;
   <  other SAS statements  >;
run;
```

NOTES / NONOTES

Tells SAS to print all notes to the SAS Log.

System Option

```
options  NOTES;
data processed_movies;
   set mydata.movies;
   <  other SAS statements  >;
run;
```

DATASTMTCHK=COREKEYWORDS

Prevents a data set from being overwritten when there is a syntax error in a MERGE, SET, UPDATE or MODIFY statement.

System Option

```
options  DATASTMTCHK=COREKEYWORDS;
data mydata.movies;
   set mydata.movies;
   <  other SAS statements  >;
run;
```

REPLACE/NOREPLACE Option

You can change the REPLACE system option with NOREPLACE or REPLACE=YES with REPLACE=NO to prevent the replacement of permanent data sets.

System Option

```
options NOREPLACE;
data mydata.movies;
  set mydata.movies;
  <  other SAS statements  >;
run;
```

Data Set Option

```
data mydata.movies(replace=no);
  set mydata.movies;
  <  other SAS statements  >;
run;
```

OBS=0 and NOREPLACE

This combination of options tells SAS to execute the step and analyze the syntax of your code without reading any input data.

System Option

```
options  OBS=0  NOREPLACE;
data mydata.movies;
   set mydata.movies;
   <  other SAS statements  >;
run;
```

SOURCE / NOSOURCE

Tells SAS to write all SAS source code to the SAS Log.

System Option

```
options  SOURCE;
data processed_movies;
   set mydata.movies;
   <  other SAS statements  >;
run;
```

NOSOURCE2 / SOURCE2

Tells SAS to write all included SAS source code to the SAS Log for audit trail purposes.

System Option

```
options  SOURCE2;
data processed_movies;
   set mydata.movies;
   %include  'c:\include-sas-code.sas';
   <  other SAS statements  >;
run;
```

Macro Debugging Strategies and Techniques

Non-Macro vs. Macro Error?

A simple way to determine whether the error received is a macro error or a non-macro error is to:

- **If the message displays a number, then the error is most likely due to non-macro SAS code;**
- **Otherwise, the error can be assumed to be a macro-related error.**

Macro Debugging Options

%PUT with options:

- _ALL_
- _AUTOMATIC_
- _GLOBAL_
- _LOCAL_
- _USER_

MACRO

MERROR

MFILE

MLOGIC

MPRINT

SERROR

SYMBOLGEN

Isolating Macro Issues

- Use %PUT to isolate problems
- Useful for the following activities:
  - ✓ Inspect a macro variable's value
  - ✓ Inspect ampersand resolution
  - ✓ Verify that a specified condition was met
  - ✓ Verify leading or trailing blanks in a value

MLOGIC and Flow of Execution

OPTIONS MLOGIC;

```
%MACRO statsproc (PROC, DSN);
  %IF %UPCASE(&proc)=MEANS %THEN %DO;
        PROC MEANS DATA=&dsn; RUN;
  %END;
  %ELSE %DO;
        PROC UNIVARIATE DATA=&dsn; RUN;
  %END;
%MEND statsproc;
%statsproc (means, SASUSER.movies);
```

MLOGIC and the SAS Log

```
1   OPTIONS MLOGIC;
2   %MACRO statsproc (PROC, DSN);
3     %IF %UPCASE(&proc)=MEANS %THEN %DO;
4        PROC MEANS DATA=&dsn; RUN;
5     %END;
6     %ELSE %DO;
7        PROC UNIVARIATE DATA=&dsn; RUN;
8     %END;
9   %MEND statsproc;
10  %statsproc (means, SASUSER.movies);
```

```
MLOGIC(STATSPROC):  Beginning execution.
MLOGIC(STATSPROC):  Parameter PROC has value means
MLOGIC(STATSPROC):  Parameter DSN has value SASUSER.movies
MLOGIC(STATSPROC):  %IF condition %UPCASE(&proc)=MEANS is TRUE
```

```
NOTE: There were 22 observations read from the data set SASUSER.MOVIES.
NOTE: PROCEDURE MEANS used (Total process time):
      real time          0.01 seconds
      cpu time           0.01 seconds

MLOGIC(STATSPROC):  Ending execution.
```

MPRINT – Generated Statements

```
OPTIONS  MPRINT;
%MACRO statsproc (PROC, DSN);
  %IF %UPCASE(&proc)=MEANS %THEN %DO;
        PROC MEANS DATA=&dsn; RUN;
  %END;
  %ELSE %DO;
        PROC UNIVARIATE DATA=&dsn; RUN;
  %END;
%MEND statsproc;
%statsproc (means, SASUSER.movies);
```

MPRINT and the SAS Log

```
1   OPTIONS MPRINT;
2   %MACRO statsproc (PROC, DSN);
3     %IF %UPCASE(&proc)=MEANS %THEN %DO;
4        PROC MEANS DATA=&dsn; RUN;
5     %END;
6     %ELSE %DO;
7        PROC UNIVARIATE DATA=&dsn; RUN;
8     %END;
9   %MEND statsproc;
10   %statsproc (means, SASUSER.movies);
```

MPRINT(STATSPROC): PROC MEANS DATA=SASUSER.movies;
MPRINT(STATSPROC): RUN;

NOTE: There were 22 observations read from the data set SASUSER.MOVIES.
NOTE: PROCEDURE MEANS used (Total process time):
real time 0.00 seconds
cpu time 0.01 seconds.

SYMBOLGEN – Macro Variables

OPTIONS SYMBOLGEN;

```
%MACRO statsproc (PROC, DSN);
  %IF %UPCASE(&proc)=MEANS %THEN %DO;
        PROC MEANS DATA=&dsn; RUN;
  %END;
  %ELSE %DO;
        PROC UNIVARIATE DATA=&dsn; RUN;
  %END;
%MEND statsproc;
%statsproc (means, SASUSER.movies);
```

SYMBOLGEN and the SAS Log

```
1   OPTIONS SYMBOLGEN;
2   %MACRO statsproc (PROC, DSN);
3     %IF %UPCASE(&proc)=MEANS %THEN %DO;
4        PROC MEANS DATA=&dsn; RUN;
5     %END;
6     %ELSE %DO;
7        PROC UNIVARIATE DATA=&dsn; RUN;
8     %END;
9   %MEND statsproc;
10   %statsproc (means, SASUSER.movies);
```

SYMBOLGEN: Macro variable PROC resolves to means

SYMBOLGEN: Macro variable DSN resolves to SASUSER.movies

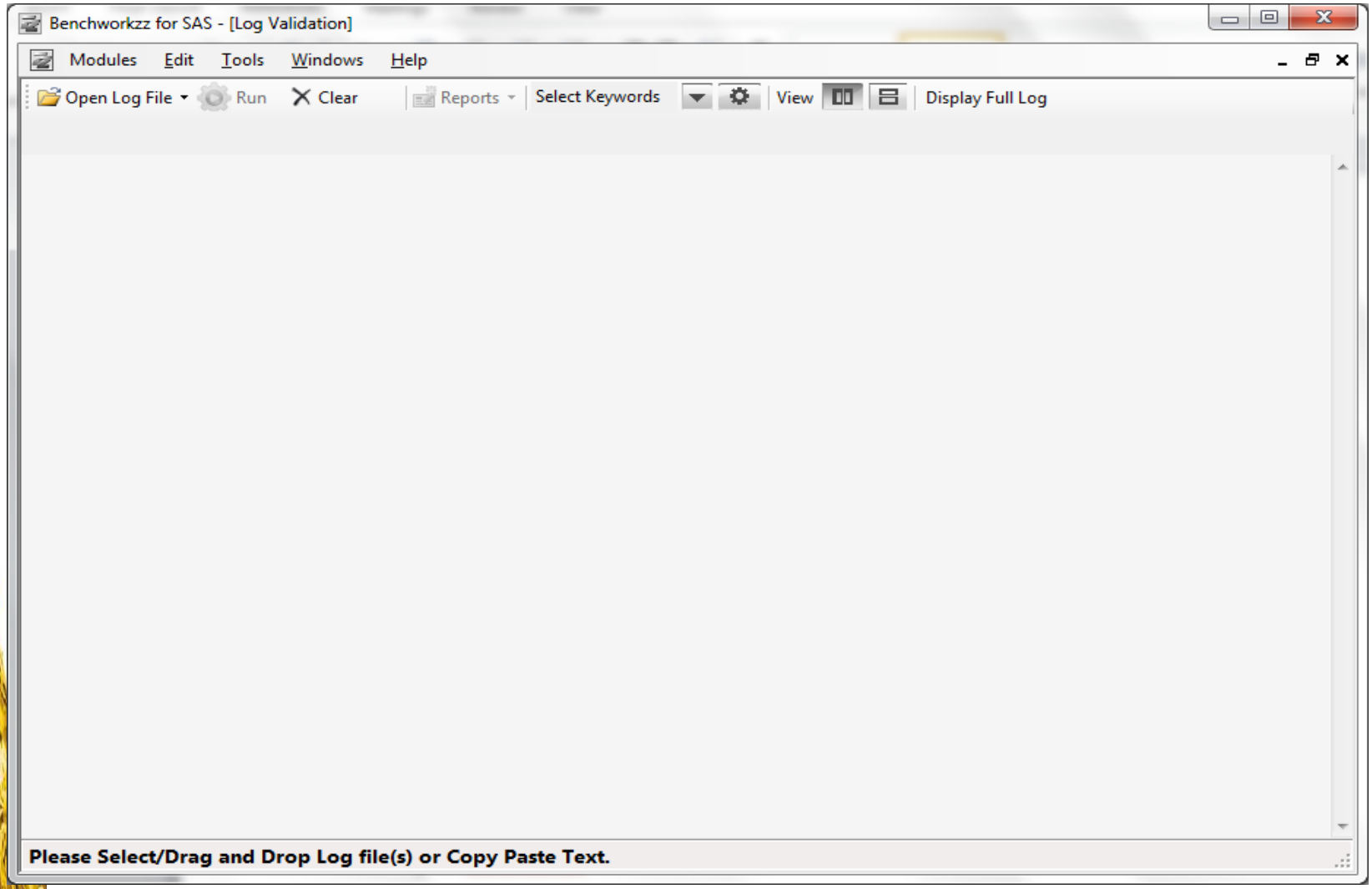
```
NOTE: There were 22 observations read from the data set SASUSER.MOVIES.
NOTE: PROCEDURE MEANS used (Total process time):
      real time          0.00 seconds
      cpu time           0.01 seconds.
```

Debugging with Benchworkzz Software

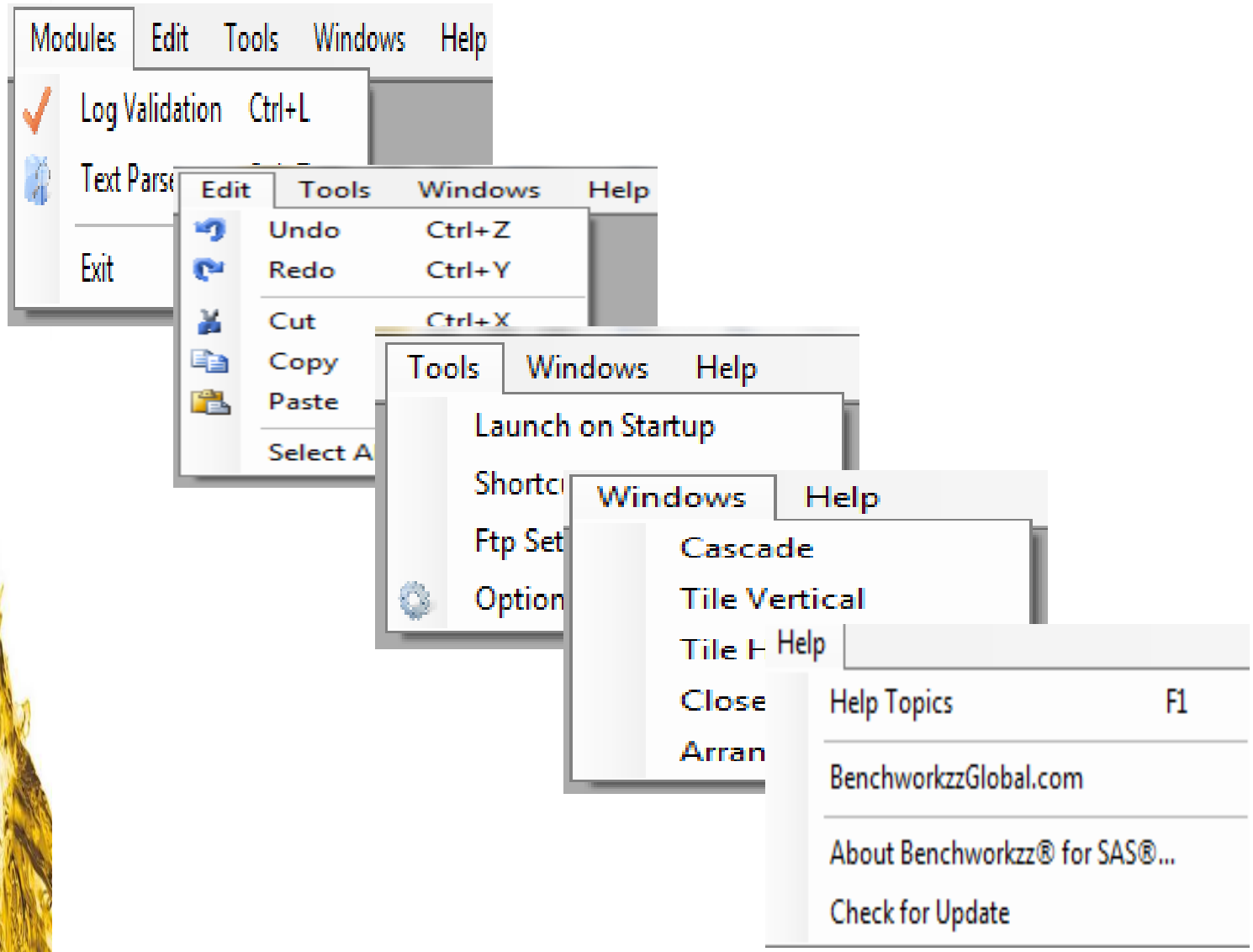
Benchworkzz Features

- Easy to use Productivity and Diagnostic Tool
- All-in-one Tool
- Identify NOTE, WARNING, ERROR and INFO
- Scrub and Sanitize Program Code
- Contains powerful Search Capabilities
- Logging and Reporting Capabilities

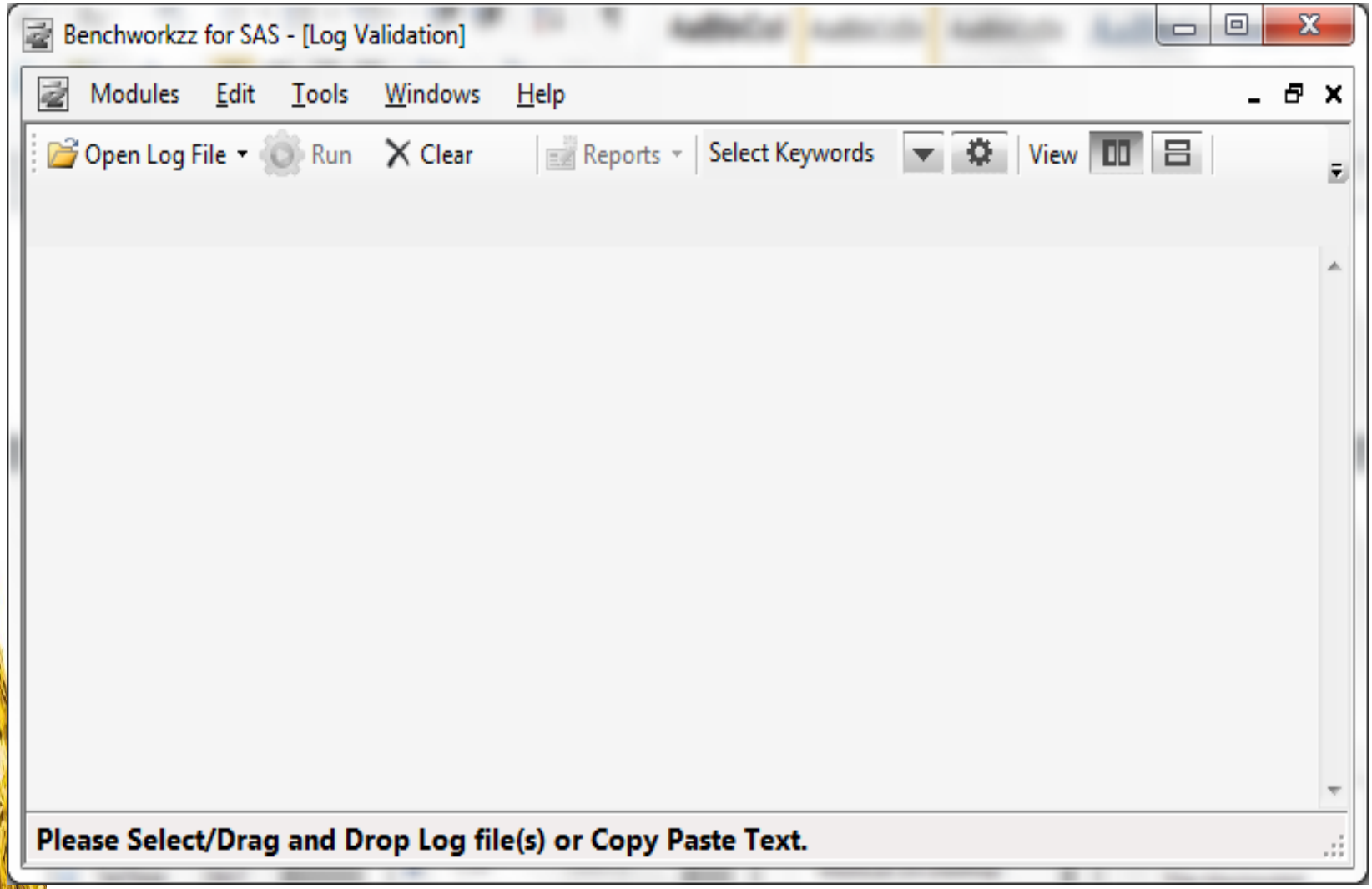
Benchworkzz Interface



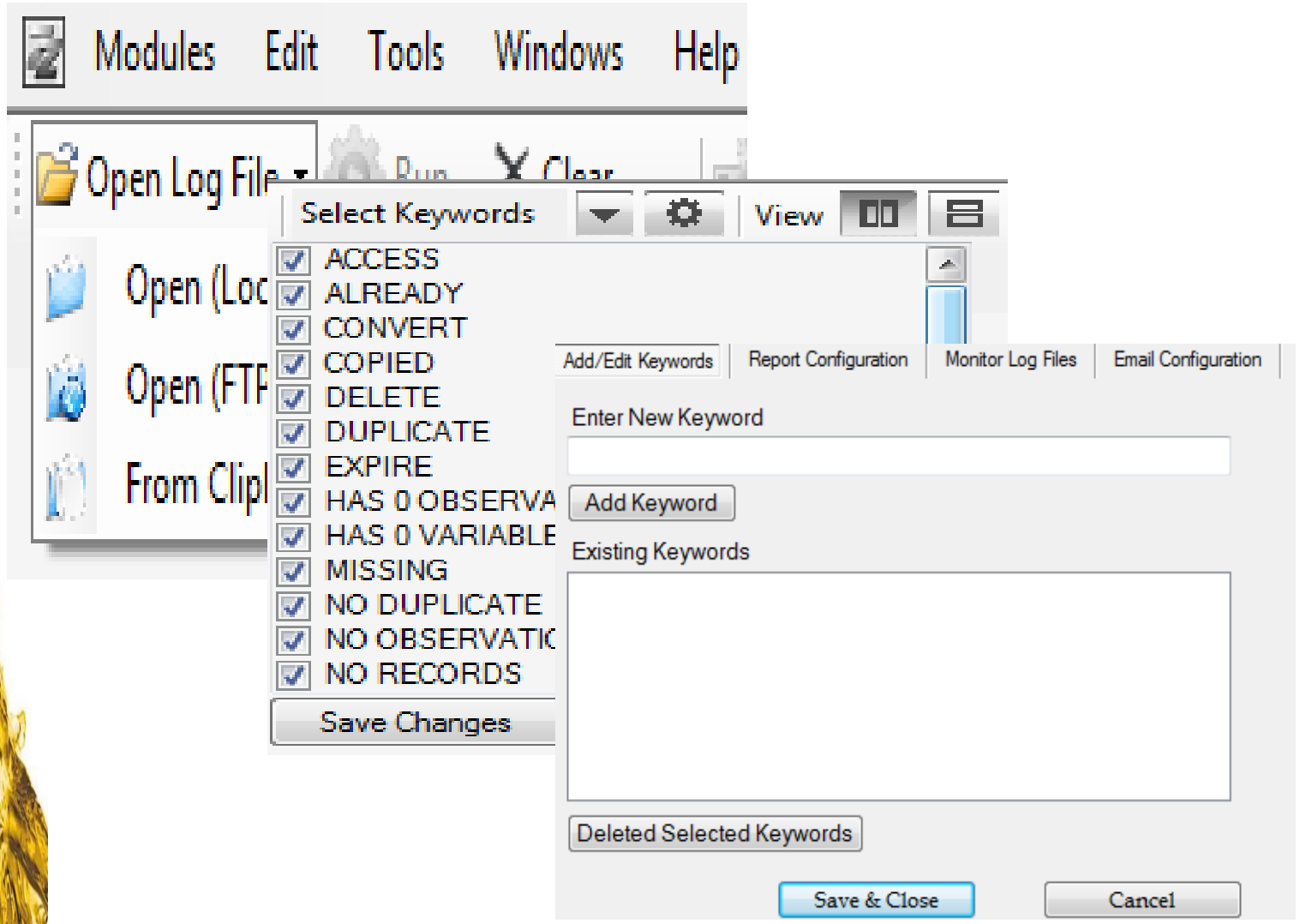
Navigation and Drop-down Menus



Benchworkzz Log Validation Module



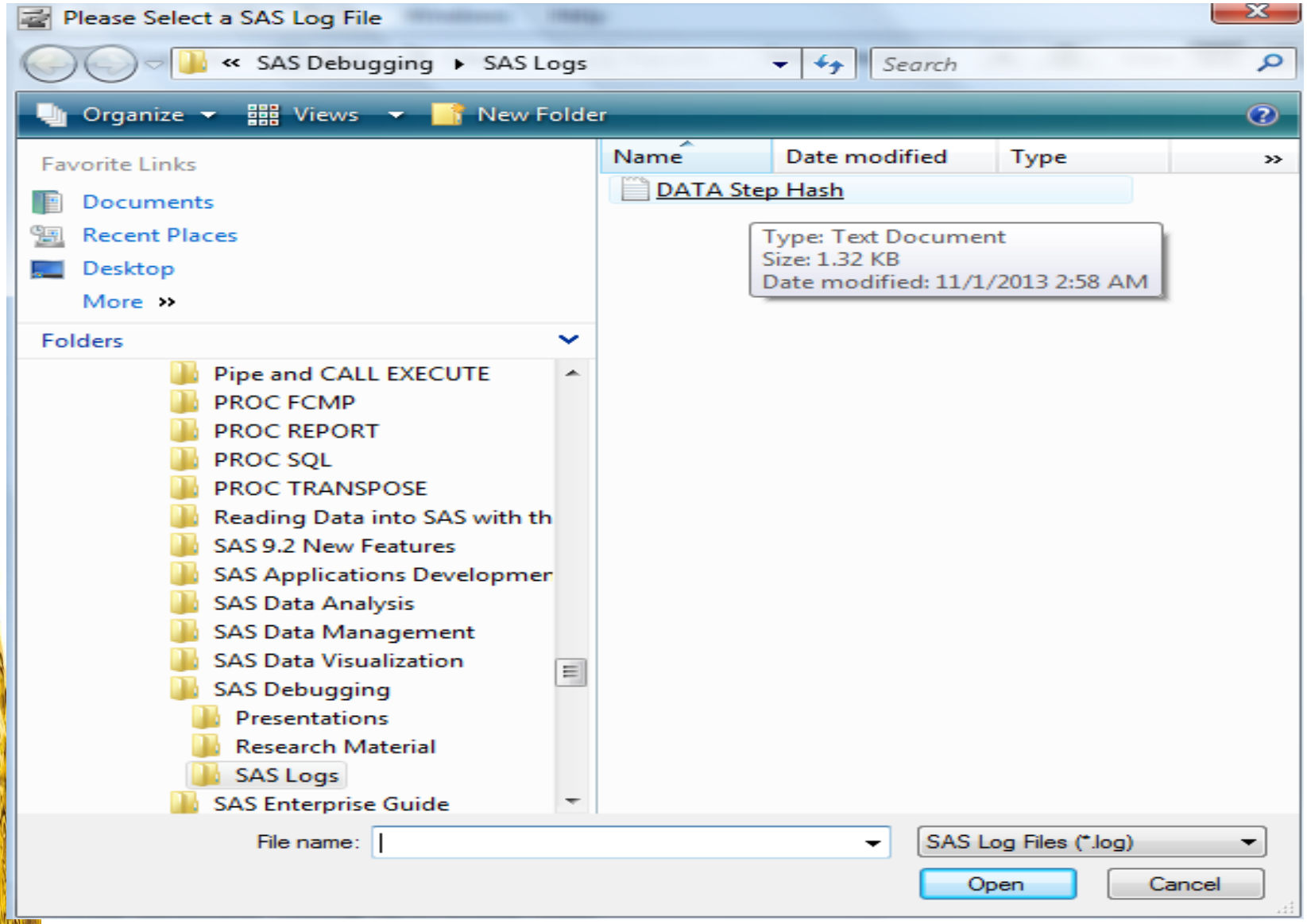
Log Validation Drop-down Menus



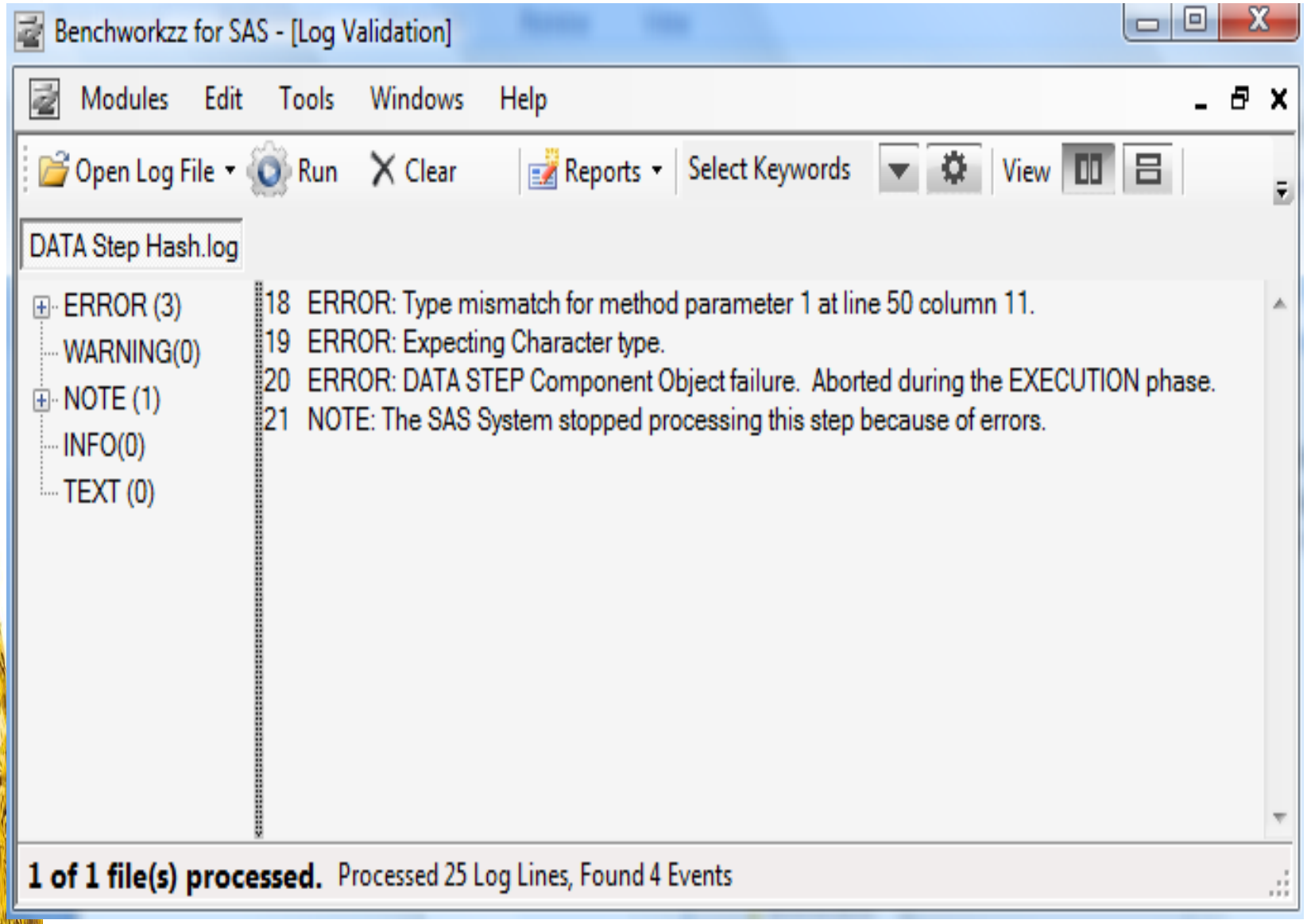
DATA Step Hash Code Example

```
data _null_;
 if 0 then set mydata.actors;  /* load variable properties into hash tables */
 if _n_ = 1 then do;
   declare Hash MatchTitles (dataset:'mydata.actors'); /* declare the name MatchTitles
                                                          for hash */
   MatchTitles.DefineKey ('Title'); /* identify variable to use as key */
   MatchTitles.DefineData ('Actor_Leading',
                           'Actor_Supporting'); /* identify columns of data */
   MatchTitles.DefineDone (); /* complete hash table definition */
 end;
 set mydata.movies end=eof;
 if eof and
   MatchTitles.find(key:title) = 0 then   /* write data using hash MatchTitles */
      MatchTitles.output(dataset:match_on_titles);
run;
```

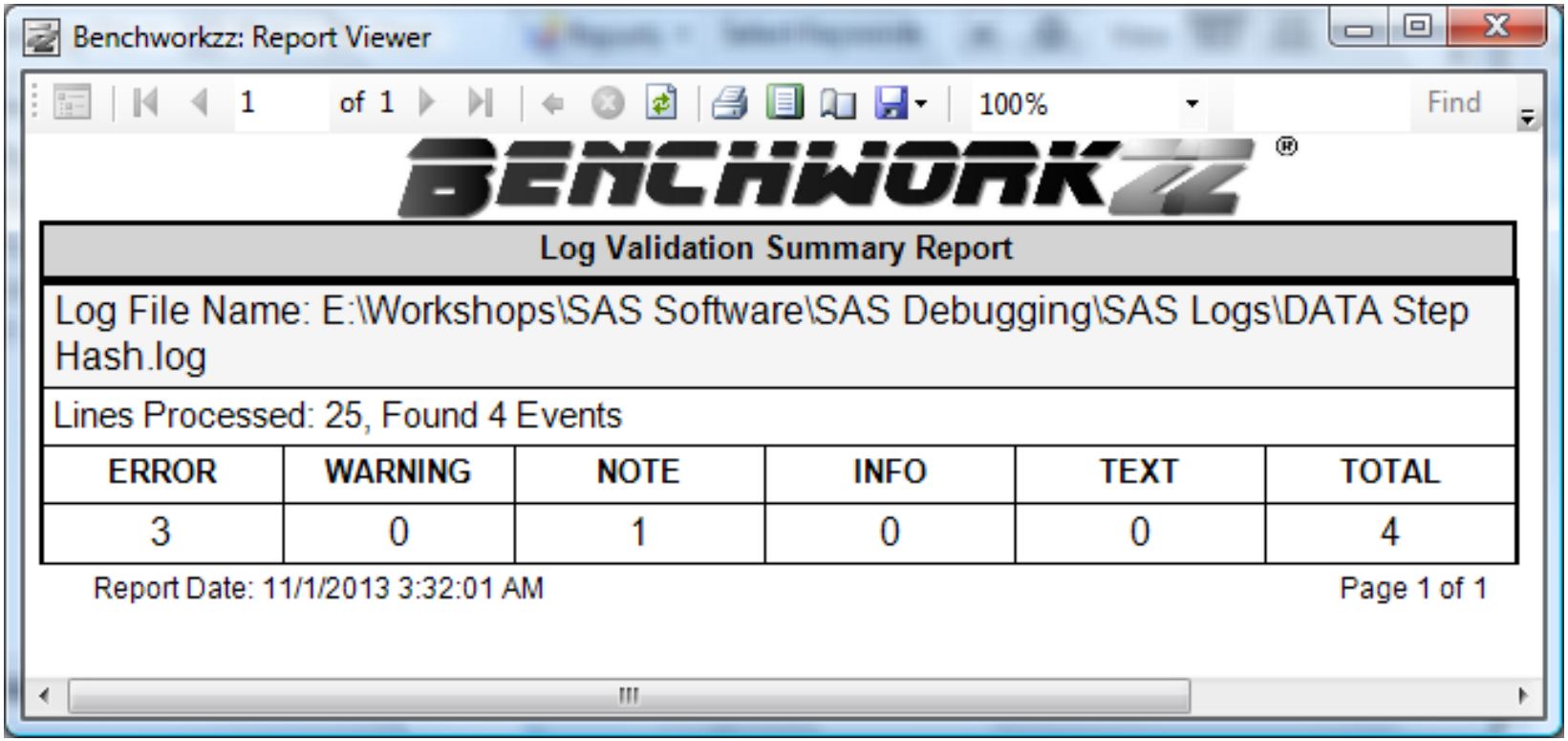
Selecting a SAS Log File



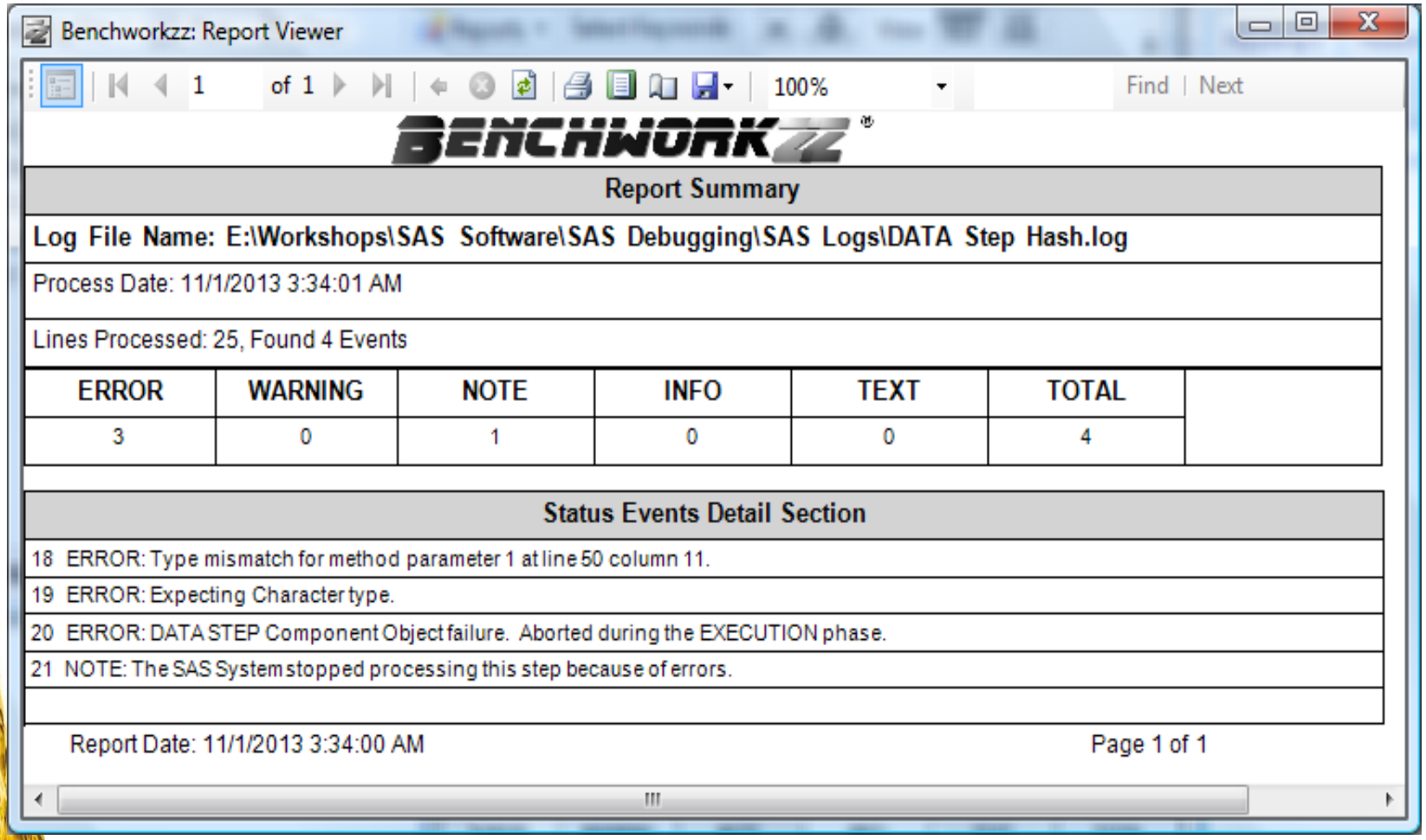
Log Validation Results



Log Validation Summary Report



Log Validation Detailed Report



Report Summary

Log File Name: E:\Workshops\SAS Software\SAS Debugging\SAS Logs\DATA Step Hash.log

Process Date: 11/1/2013 3:34:01 AM

Lines Processed: 25, Found 4 Events

ERROR	WARNING	NOTE	INFO	TEXT	TOTAL
3	0	1	0	0	4

Status Events Detail Section

18 ERROR: Type mismatch for method parameter 1 at line 50 column 11.

19 ERROR: Expecting Character type.

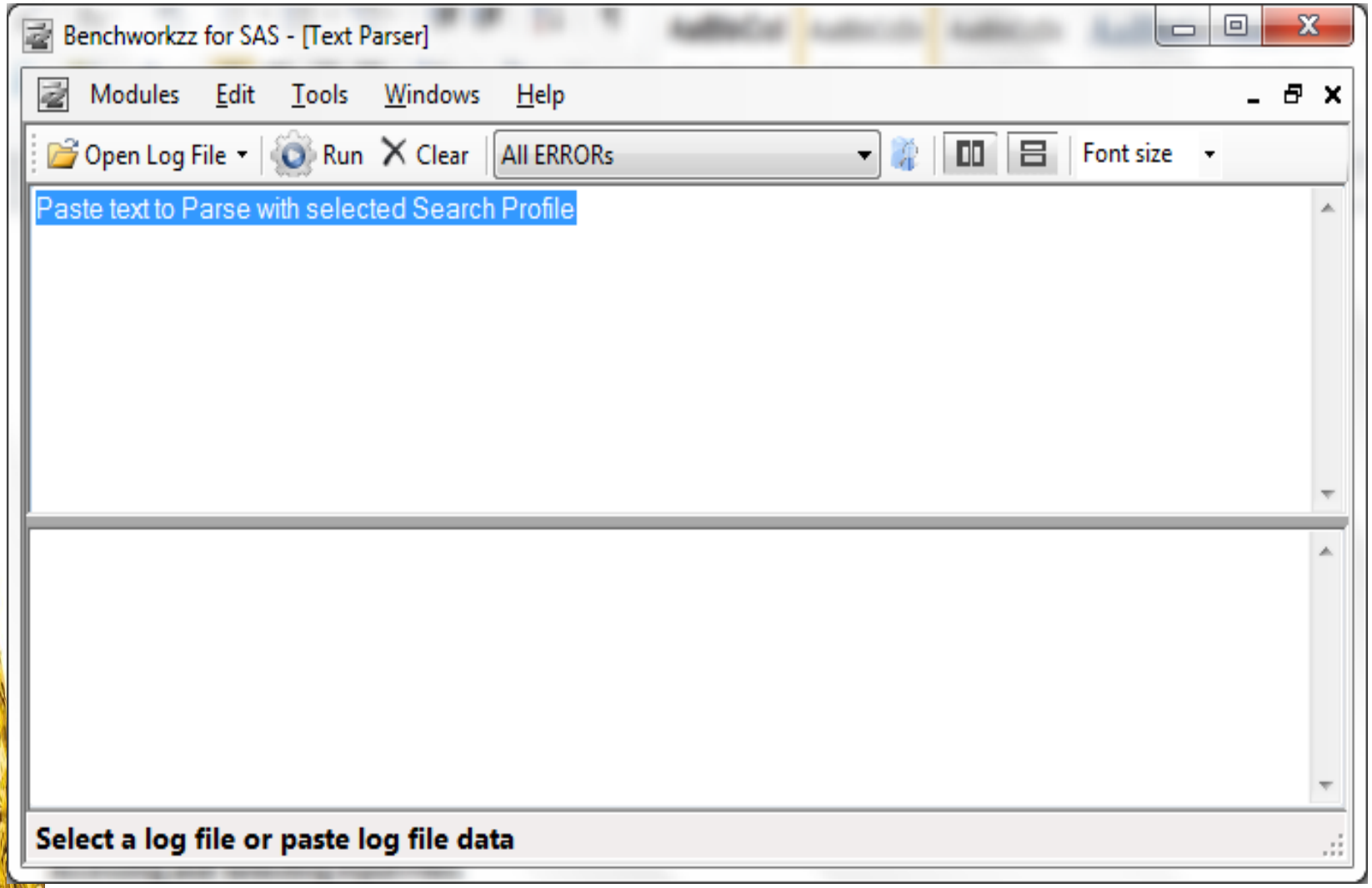
20 ERROR: DATA STEP Component Object failure. Aborted during the EXECUTION phase.

21 NOTE: The SAS System stopped processing this step because of errors.

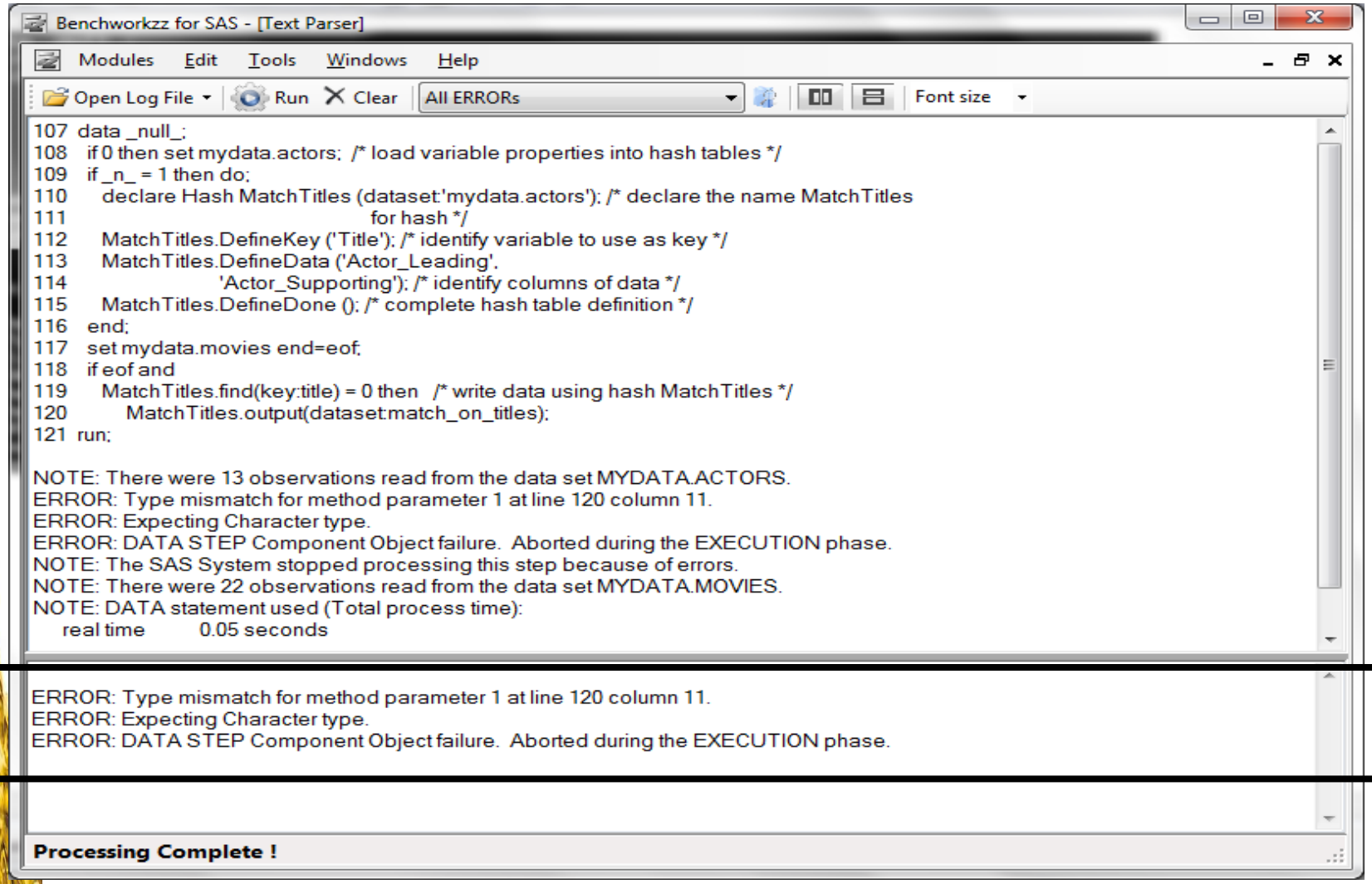
Report Date: 11/1/2013 3:34:00 AM

Page 1 of 1

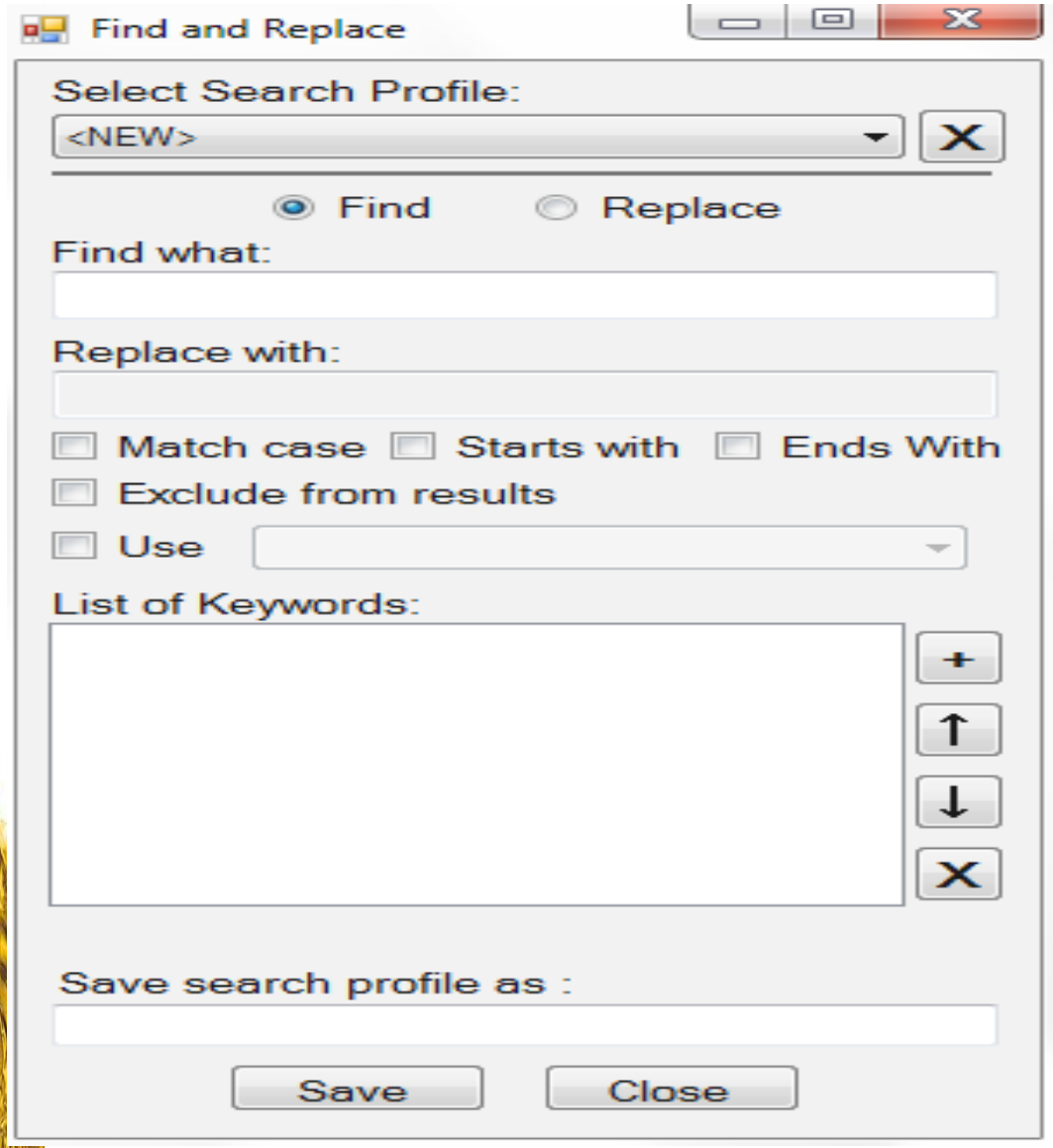
Benchworkzz Text Parser Module



Text Parser and Displaying Errors

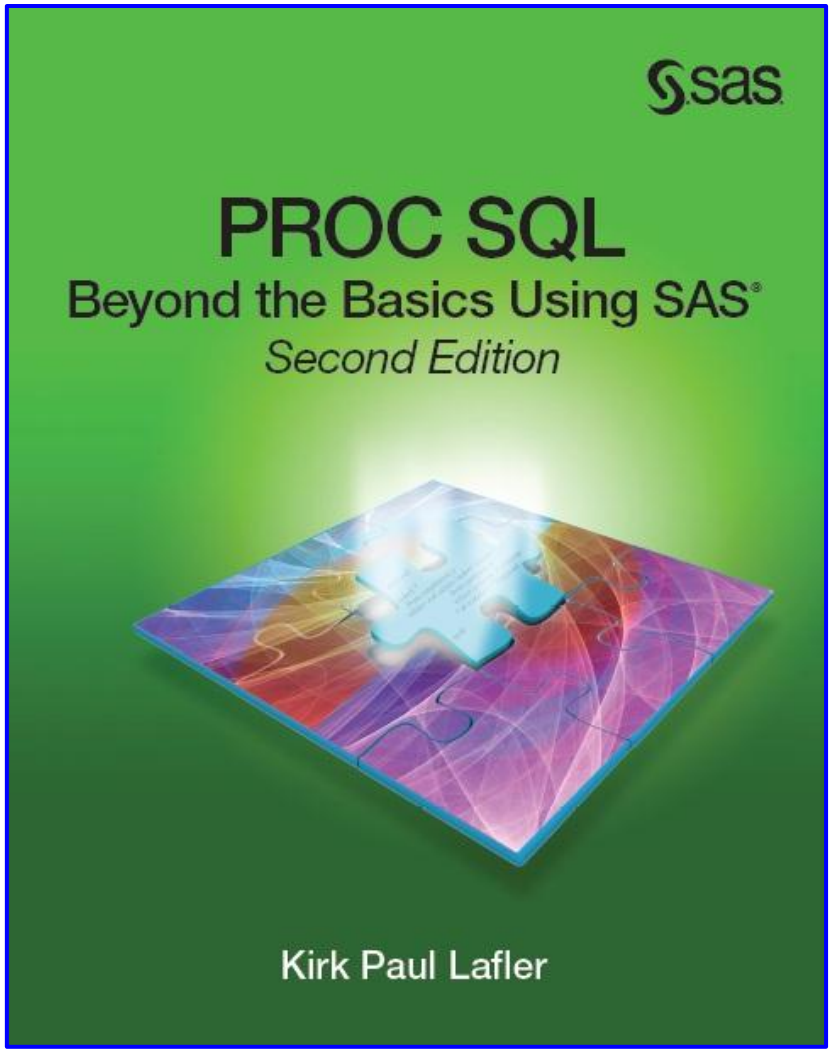


Search Profiler Editor



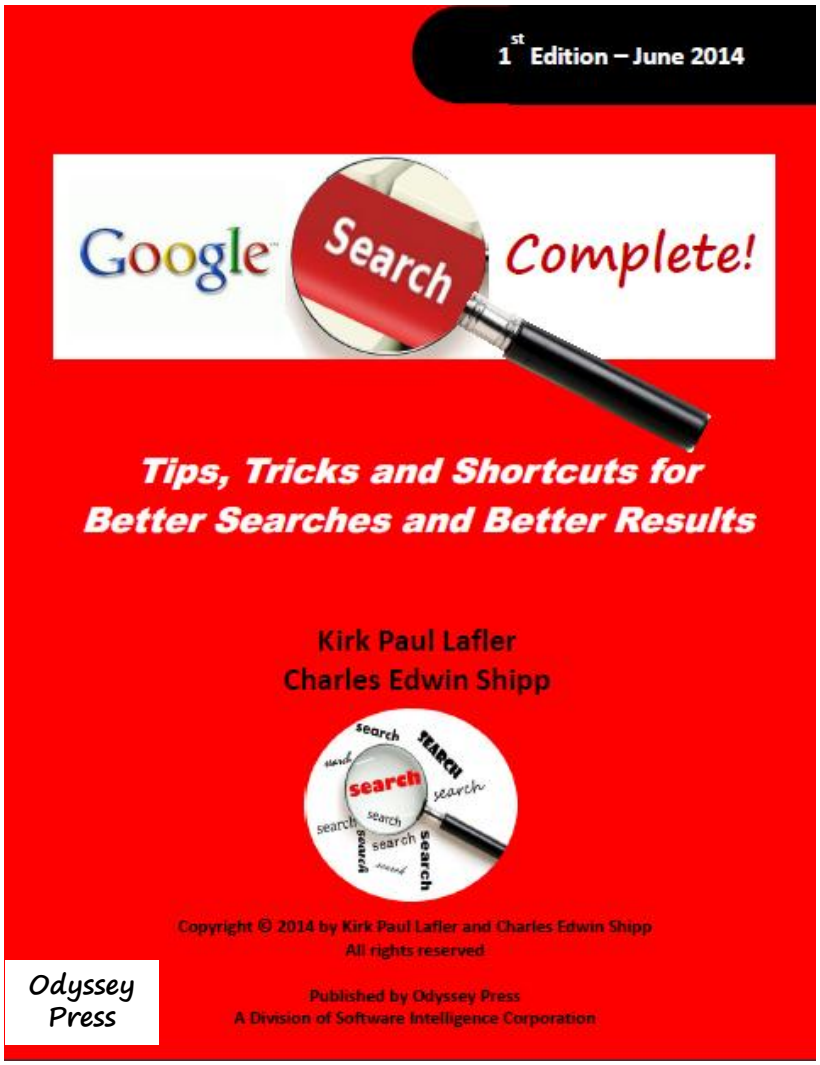
Conclusion

- Types of Errors and Warnings
- Errors SAS Can't Help With
- Debugging Strategies and Techniques
- Macro Debugging Strategies and Techniques
- Debugging with Benchworkzz Software



An SQL Book with "behind the scenes" details, explanations and lots of examples

Available from SAS Press!



Search like a Pro!
Filled with
Tips, Tricks and
Shortcuts
for Better Searches
and Better Results

Available June 2014!

Thank you for attending!

Questions?

A presentation by

Kirk Paul Lafler

KirkLafler@cs.com

@sasNerd