TAP TO GO BACK TO KIOSK MENU

# SAS® GLOBAL FORUM 2020

MARCH 29 – APRIL 1
WASHINGTON, DC

USERS PROGRAM

# Writing Data Quality rules using SAS Meta-programming

TAP TO GO BACK TO KIOSK MENU

**Shreyas Dalvi**
SunTrust Bank now Truist



## Abstract


- Data quality/Data Governance is important for any organization's data office, considering the major impact of bad data in all of the downstream applications like Regulatory reporting, advanced analytics.
- There are many tools in market to asses and govern the data quality of data warehouse and data marts. This paper explains how to implement data quality check for data marts using SAS® Meta-programming technique.
- Meta-programming is way of programming where a program is used to write a program. In SAS this can be easily achieved by using SAS Macros and Metadata table. In this paper we have demonstrated how we can store and maintain the data quality rules in one metadata table and run the rules on various tables/extracts in data mart using SAS metadata driven approach. Data quality rules are applied on multiple extracts which are created by joining dimension and fact tables. Final report provides list of variables with rule name, description and percentage of error as per the rule. With all rules in single table we can achieve more flexibility in rule writing, maintenance and process automation.
- This paper is intended for Data science and Data management professional with intermediate level of SAS expertise.



Shreyas Dalvi

## Intro

- Data quality/Data Governance is important for any organization's data office, considering the major impact of bad data in all of the downstream applications like reporting, advanced analytics.
- There are many tools in market for identifying the data quality issues in data marts. With SAS we can monitor data quality more flexibly and applications can be custom made using SAS programming.
- In this ePoster we have used the meta-data driven programming technique to check errors in data warehouse.

**Why Meta-data driven technique ?**

Meta data driven SAS programming can generate the SAS code using a single metadata table and macro programming. There are various benefits using this technique.

- Excellent change management
- Eliminates hard coding
- Clean and short code
- Easy to increase the scope of code
- Easy to make changes in code

## Objective

To implement various data quality checks for Data warehouse

- Column-level testing
- Business rule testing
- Structural level-testing

# Writing Data Quality rules using SAS Meta-programming

Shreyas Dalvi
SunTrust Bank now Truist



## Metadata driven Architecture

- Metadata table drives the SAS program
- The entries in table acts as a parameters for the main macro in SAS program.
- Table name and field name are used for identifying the input for checking Data quality
- RuleDataset is output dataset where all the incorrect data is captured from input extract data.
- Rulecode is a SQL where clause condition to capture errors
- No of entries = No of Rules on a data warehouse
- SAS Macro will iterate N no of times to calculate the errors as per each DQ rule


Data Source

Metadata Table

| Extract | TableName | FieldName | RuleID | RuleDesc | RuleDataset | Rulecode | Dimension |
|---|---|---|---|---|---|---|---|
| Extract1 | table1 | field1 | tab1_field1 | Field1 should not be NULL | Field1_NULL | Field1 NE NULL | Completeness |
| … | … | … | … | … | … | … | .. |
| ExtractN | tableN | fieldN | tabn_fieldn | FieldN should not be negative | FieldN_NEG | fieldN < 0 | Validity |

Set of Codes


Summarized Data Quality Report

DQ Dashboards

# Writing Data Quality rules using SAS Meta-programming

Shreyas Dalvi
SunTrust Bank now Truist



## Code

**Extracting the data from Data warehouse/Data marts or any other sources**

```
%let monyy =%sysfunc(today(),date7.);
libname CSVOUT  "/.../path/../CSV/&monyy.";
libname SASOUT  "/.../path/../SASDATA/&monyy.";
libname SASDATA "/.../path/../EXTRACT/&monyy.";
libname RULES   "/.../path/../code";
libname DWLIB sqlsvr user="ABCD" password="******" datasrc='ORAC_DB' schema='SCH';

PROC SQL;
CREATE TABLE SASDATA.EXTRACT_1 AS  SELECT var1, var2, ..., var_n FROM DWLIB.table_name WHERE conditions;
QUIT;
```

**Creating the empty table one for one extract to hold the errors**

```
proc sql;
create table ERR_Summary_1 (Extract char(50), Ruleid char(250), RuleDesc varchar(256),Application char(50), TableName
varchar(50),FieldName varchar(50), TotalCount int, FailureCounts int, FailurePercentage int, RuleRunDate date
format=datetime20.);
quit;
```

**Creation of Macro variables using the Metadata table**

```
PROC SQL;
select (count(*)) into :n trimmed from RULES.RULES_META where Extract = "Extract1";
select trim(TableName) into :Tablenm1-:Tablenm&n. from RULES.RULES_META where Extract = "Extract1";
select trim(FieldName) into :Fieldnm1-:Fieldnm&n. from RULES.RULES_META where Extract = "Extract1";
select trim(Ruleid) into :Ruleidm1-:Ruleidm&n. from RULES.RULES_META where Extract = "Extract1";
select trim(RuleDesc) into :Ruledescm1-:Ruledescm&n. from RULES.RULES_META where Extract = "Extract1";
select trim(RuleDataset) into :RuleDatasetm1-:RuleDatasetm&n. from RULES.RULES_META where Extract = "Extract1";
select trim(Rulecode) into :Rulecode1-:Rulecode&n. from RULES.RULES_META where Extract = "Extract1";
select trim(Dimension) into :Dimension1-:Dimension&n. from RULES.RULES_META where Extract = "Extract1";
quit;
```

# Writing Data Quality rules using SAS Meta-programming

Shreyas Dalvi
SunTrust Bank now Truist



## Code

**Creating the main Macro which iterates n no of times depending on no of rules on metadata:**

```
%macro doit();
%do i=1 %to &n.;
proc sql;
create table SASOUT.&&RuleDatasetm&i. as
select "&&ruleidm&i." as Ruleid, *, put(datetime(),datetime18.) as Timestamp from SASData.Extract_1 where &&rulecode&i.;
quit;
```

**Putting summary of errors for each rule:**

```
proc sql;
insert into ERR_Summary_1 (Extract, Ruleid, RuleDesc, Application, SchemaName, TableName, FieldName, TotalCount,
FailureCounts,FailurePercentage,Dimension, Frequency,As_of_Date,RuleRunDate)
select  "Extract_1" as Extract, "&&Ruleidm&i." as Ruleid,
"&&Ruledescm&i." AS RuleDesc, (select distinct application from SASData.Extract_1) as Application,
"&&Tablenm&i." AS TableName, "&&Fieldnm&i." as FieldName,
(SELECT count(*) FROM SASData.Extract_1) AS TotalCount,
 count(*) as FailureCounts,
(count(*)/(SELECT count(*) FROM SASData.Extract_1)*100) as FailurePercentage,
"Monthly" as Frequency, %sysfunc(datetime()) from SASOUT.&&RuleDatasetm&i..;
quit;
```

**Exporting the Error Records for each Rule into CSV:**

```
proc export data=SASOUT.&&RuleDatasetm&i.
outfile="/../path/../CSV/&monyy./&&RuleDatasetm&i.._&monyy..CSV"
   dbms=csv
   replace;
run;
%end;
%mend;

%doit();
```

**Exporting the Summary Error Report for each Rule into CSV:**

```
proc export data=ERR_Summary_1_&monyy.
outfile="/../path/../CSV/&monyy./ERR_Summary_1_&monyy..CSV"
   dbms=csv
   replace;
run;
```

# Writing Data Quality rules using SAS Meta-programming

Shreyas Dalvi
SunTrust Bank now Truist



## Final Report

| Extract | Ruleid | RuleDesc | TableName | FieldName | TotalCount | FailureCounts | FailurePercentage | Dimension | RuleRunDate |
|---|---|---|---|---|---|---|---|---|---|
| Extract_1 | CUST_INFO.FICO_SCORE.VALID_FICO_SCORE | FICO Score should be in range 300-899 | CUST_INFO | FICO Score | 559739 | 768 | 0.137% | Validity | 07FEB2020:15:45:00 |
| Extract_1 | CUST_INFO.ZIP.BLANK_ZIP | ZIP Should not be missing | CUST_INFO | ZIP | 559739 | 233 | 0.042% | Completeness | 07FEB2020:15:45:00 |
| … | … | … | … | … | … | … | … | … | .. |
| Extract11 | PROD_INFO.LOAN_NUM.VALID_LO_NUM | Loan number should not be missing | PROD_INFO | LOAN_NUM | 40000 | 12 | 0.030% | Completeness | 07FEB2020:15:45:00 |

- Final Report looks like table shown above with all the DQ metrics
- This table is created concatenating all the outputs from each extract and gives out summary of all errors
- Can be used as input to the DQ Dashboards
- SAS jobs can be scheduled every day/month/week without any changes in code using SAS Macro parameters
- Any addition/ deletion or modification of Rule can be done by changing only the metadata table
- No need to change the actual main program code

## References

Kirk Paul Lafler. 2018 “Data-driven Programming Techniques Using SAS® Metadata, Software Intelligence Corporation” https://support.sas.com/content/dam/SAS/support/en/sas-global-forum-proceedings/2018/1653-2018.pdf

Cassell, D. L. 2007. Don't Be Loopy: Re-Sampling and Simulation the SAS Way. in SAS Global Forum 2007 Conference. S. I. Inc, ed. SAS Institute Inc.

SAS Institute Inc, SAS Macro Language Reference, First Edition

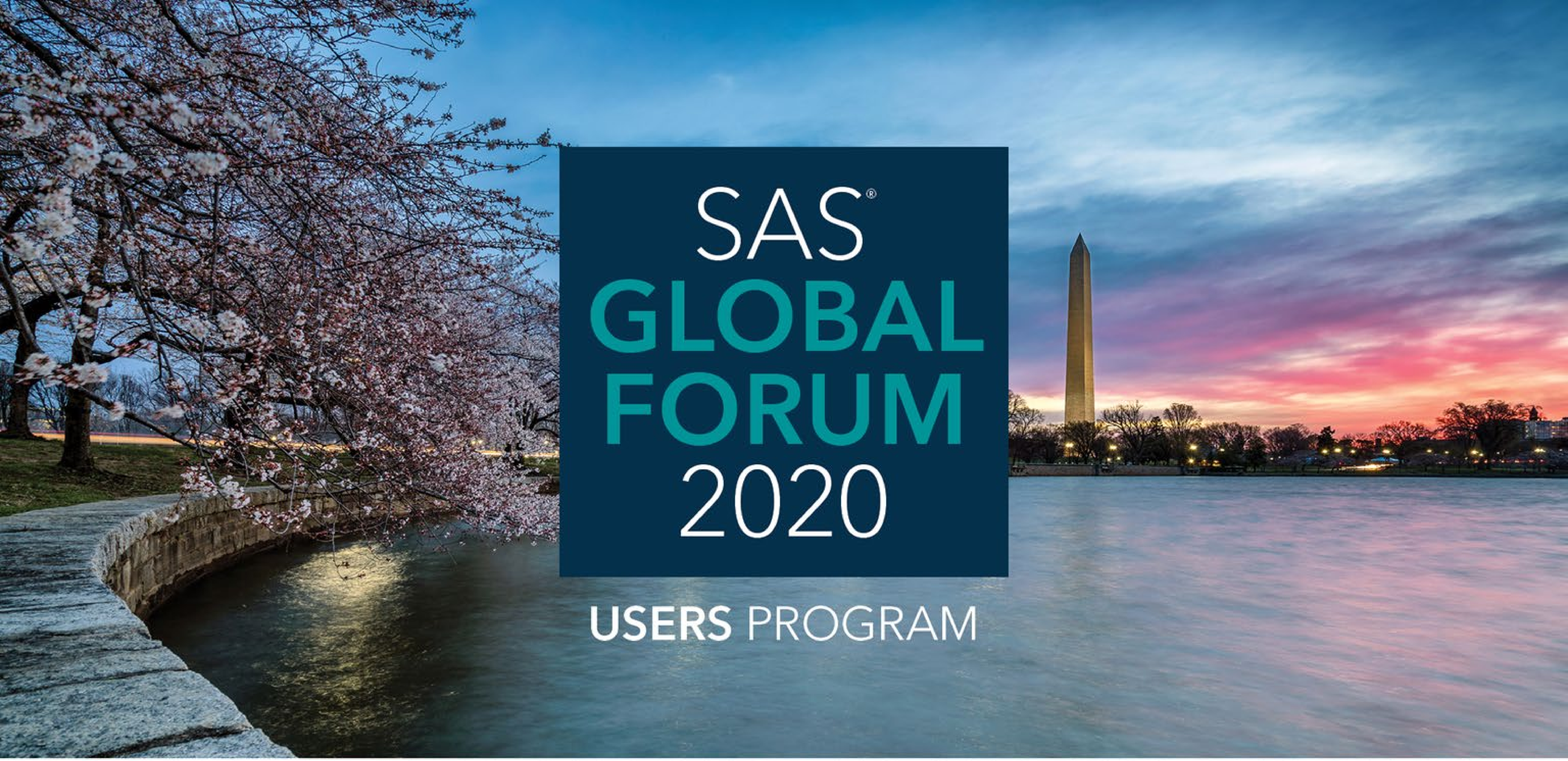
SAS Institute Inc, Getting Started with the SQL Procedure, Version 6, First Edition

## Acknowledgment

I would like to thank SAS for an excellent opportunity to present my ideas. Thank you Risk and Finance Data management Team(Michelle, Abhay, Sudhesh, Siva, Ketki, Sanjeev) , SunTrust Now Truist for supporting me.

## Contact

Shreyas Dalvi
AVP – SunTrust Bank Now Truist
Phone: 813-573-4627
E-mail: shreyasdalvi@mail.usf.edu
LinkedIn: https://www.linkedin.com/in/shreyasdalvi/




MARCH 29 – APRIL 1 | WASHINGTON, DC | #SASGF