Using SAS® to Manage And Maintain Your Data Repository

Paper # AD48

José Centeno is a Senior Data Solutions Developer at NORC at the University of Chicago.

Using SAS® to Manage And Maintain Your Data Repository

Paper # AD48

José Centeno
NORC at the University of Chicago
Chicago, IL

Background

- The goal is to have a centralized storage facility for managing data files from various formats
- A data repository increases data integrity and quality, enhances security, and facilitates research and analysis
- SAS ODBC features simplifies connections to different database applications (SQL Server, MySQL, PostgreSQL, etc.)
- SAS SQL Pass-Through Facility offers the ability to assign tasks to the database system using SQL

General workflow example

- Build a document outlining the input sources (SAS file, csv, database table, etc.)
- Connect and read source data into SAS
- Compare structure of input data against desired target schema
  - Data type, variable list, variable length
- Load data into target environment
- Apply SQL permissions to target object

Manage and Maintain your Repository

- Describing the inputs and target sources in an external file facilitates documentation, automation and QC process

source	cdr_target	var_name	var_type	var_length
sdr_2023_responses_prog	A4S_response_data	Z_EMZIP_L	2	100
sdr_2023_responses_prog	A4S_response_data	Z_PHD_DATE	2	255
sdr_2023_responses_prog	A4S_response_data	WORKPHONE_IS_VALID	1	8
sdr_2023_responses_prog	A4S_response_data	CELLPHONE_IS_VALID	1	8
sdr_2023_responses_prog	A4S_response_data	Z_YOB_PLUS_18	2	15
sdr_2023_responses_prog	A4S_response_data	Z_COHORT	2	255
sdr_2023_responses_prog	A4S_response_data	ADDR1	2	1024
sdr_2023_responses_prog	A4S_response_data	ADDR2	2	1024
sdr_2023_responses_prog	A4S_response_data	CITY	2	1024
sdr_2023_responses_quex	survey_response_data	PHDFIELDV	2	100
sdr_2023_responses_quex	survey_response_data	BAYRCHK1	1	8
sdr_2023_responses_quex	survey_response_data	ADDR1_LAST	2	100
sdr_2023_responses_quex	survey_response_data	SALARY_DIGITSMAX	1	8
sdr_2023_responses_quex	survey_response_data	APT_LAST	2	16
sdr_2023_responses_quex	survey_response_data	CITY_LAST	2	100
sdr_2023_responses_quex	survey_response_data	STATE_LAST	2	2
sdr_2023_responses_vcc	vcc_response_data	ZIP_LAST	2	10
sdr_2023_responses_vcc	vcc_response_data	STATPROV_LAST	2	100
sdr_2023_responses_vcc	vcc_response_data	PSTCDE_LAST	2	12
sdr_2023_responses_vcc	vcc_response_data	CELL_DIALING_1	1	8
sdr_2023_responses_vcc	vcc_response_data	NOCONTA1	1	8
sdr_2023_responses_vcc	vcc_response_data	LANGLN2	1	8
sdr_2023_responses_vcc	vcc_response_data	STATPROV_LAST	2	255

Manage and Maintain your Repository

- Data-driven approach to load multiple files

```
%local vars;
%let vars='';
%let nvars=;

proc sql ;
    select count(NAME)
    into :nvars
    from master
    where cdr_output="&source." and note ^='derived' and upcase(name) ^= 'SU_ID';

    select strip(NAME)
    into :vars separated by ","
    from master
    where cdr_output="&source." and note ^='derived' and upcase(name) ^= 'SU_ID';

    select strip(NAME)
    into :keepvars separated by " "
    from master
    where cdr_output="&source." and note ^='derived';

quit;

data VIEW_TEMP&SYSDATE. /view=VIEW_TEMP&SYSDATE.;
    set &libin..&datain.(keep=SU_ID &keepvars.);
    if cmiss(&vars.)=&nvars. then delete;
run;
```

Manage and Maintain your Repository

- Inspect the data structure of source and target and check for differences
  - Data type, new variables, deleted variables

```
data qc;
 merge input_vars(in=_sas)
       sql_vars(in=_sql);
 by name;
 in_sas=_sas;
 in_sql=_sql;
 run;
```

```
data _null_;
 set qc;
 if type ^= sql_dbtype or in_sas=0 or in_sql=0 then do;
     put "WARNING: Changes in Sources.%str(;)";
     call symputx("droptbl",1);
 end;
 run;
```

Flag structural changes

Manage and Maintain your Repository

- In a DEV environment, you may drop and recreate an object dynamically

```
%if &droptbl.=1 %then %do;
    %put %str(NOTE: Dropping Table &dataout....);
    proc sql _method feedback exitcode;
        connect to ODBC
        (CONNECTION=GLOBAL NOPROMPT="server=&reposerv.;
            DRIVER=ODBC Driver 17 for SQL Server;Trusted_Connection=yes; database=&dbname."
        );
        execute (   drop table &schema..&dataout. ) by ODBC;
           %put &=sqlxrc;
           %put &=sqlxmsg;
        disconnect from ODBC;
    quit;
%end;
%else %do;
    %put %str(NOTE: Truncating Table &dataout....);
        proc sql _method feedback exitcode;
        connect to ODBC
        (CONNECTION=GLOBAL NOPROMPT="server=&reposerv.;
            DRIVER=ODBC Driver 17 for SQL Server;Trusted_Connection=yes; database=&dbname."
        );
        execute (   truncate table &schema..&dataout. ) by ODBC;
           %put &=sqlxrc;
           %put &=sqlxmsg;
        disconnect from ODBC;
    quit;
%end;
```

Truncating is more efficient than deleting

Example

- Load data into a database by creating a SAS view and using PROC APPEND
  - If target table does not exist in the database, PROC APPEND will create it

Source data

Target database

```
libname icode postgres server='myserver' port=9500
user='myuser' password="&dbpass."
database='mydatabase';

libname mainrepo odbc noprompt="server=&reposerv.;
DRIVER=ODBC Driver 17 for SQL Server;Trusted_Connection=yes;database=&dbname."
schema=&schema. ;


%PUT %LEFT(NOTE:) Creating staging table for loading &table....%str(;) ;
proc sql _method feedback exitcode;
    create view __temp as
    select * from icode.&table.;
quit;

%PUT %LEFT(NOTE:) Loading Table &table. ...%str(;) ;
proc append data=__temp base=&stagelib..&table. force;
run;

proc datasets lib = work nolist nowarn memtype = (data view);
    delete __temp;
quit;
run;
```


Example

- Create a SQL View in the repository for reporting purposes

```
proc sql;
 connect to ODBC
 (CONNECTION=GLOBAL NOPROMPT="server=&reposerv.;
 DRIVER=ODBC Driver 17 for SQL Server;Trusted_Connection=yes;database=&dbname.");
 execute
 (
     CREATE VIEW [nsuite].[vw_address_person] AS
     SELECT
      [address].[address_key]
     ,[address].[origin_date]
     ,[address].[active]
     ,[address].[addr1]
     ,[address].[addr2]
     ,[address].[unit]
     ,[address].[city]
     ,[state].[state]
     ,[address].[zip]
     ,[address].[zip4]
     ,[address].[territory]
     ,[address].[postal_code]
     ,[address].[country]
     ,[person_address].[person_key]
     ,[person_address].[address_usage]
     ,[person_address].[address_status]
     ,[person_address].[address_result]
     FROM       [address]
     LEFT JOIN [person_address] ON [address].[address_key]=[person_address].[address_key]
     LEFT JOIN [state]          ON [state].state_id = [address].state

 ) by ODBC;
 disconnect from ODBC;
 quit;
```


Example

- Read-in a SQL View definition and load it to target database

vw_ALL_CP_EMAIL.sql

: > WORK_TEMP > vw_ALL_CP_EMAIL.sql

```
1   CREATE VIEW [NSPROD].[vw_ALL_CP_EMAIL] AS
2   SELECT
3    [NSPROD].[EMAIL].su_id
4   ,emailsrc.sample_unit_key
5   ,[NSPROD].[EMAIL].person_key
6   ,[NSPROD].[EMAIL].person_roster
7   ,[NSPROD].[EMAIL].person_type
8   ,[NSPROD].[EMAIL].person_type_label
9   ,[NSPROD].[EMAIL].email_key
10  ,[NSPROD].[EMAIL].email_address
11  ,[NSPROD].[EMAIL].active
12  ,[NSPROD].[EMAIL].link_rank
13  ,[NSPROD].[EMAIL].origin_date
14  ,emailsrc.original_source
15  ,[emailsrc].most_recent_source
16  ,[emailsrc].most_recent_source_date
17  FROM      [NSPROD].EMAIL
18  LEFT JOIN [NSPROD].[vw_min_max_email_src] as emailsrc
19  ON NSPROD.EMAIL.SU_ID=emailsrc.SU_ID
20  AND NSPROD.EMAIL.EMAIL_KEY=emailsrc.RECORD_KEY
21  WHERE SUBSTRING(NSPROD.EMAIL.SU_ID,1,1)='1'
22  AND PERSON_TYPE=2;
```

Example Cont.

```
%macro table_syntax(tbl=);
    filename part1 temp;
    filename part2 "&sqlviews.\&table..sql";
    filename part3 temp;

    data _null_;
        file part1 ;
        put 'proc sql _method feedback; connect to ODBC ( &sqlinit. ); execute (';
        file part3 ;
        put ') by ODBC; disconnect from ODBC; quit;';
        stop;
    run;
%mend;
```

SQL file with view definition

```
*Drop View if Exists;
proc sql _method feedback;
connect to ODBC
(
CONNECTION=GLOBAL NOPROMPT="server=&reposerv.;
DRIVER=ODBC Driver 17 for SQL Server;Trusted_Connection=yes;database=&dbname."
);
execute
(
    IF (SELECT 1 FROM SYS.VIEWS WHERE NAME=%str(%')&table.%str(%') AND TYPE='V')=1
        BEGIN DROP VIEW [&schema.].[&table.] END
) by ODBC;
disconnect from ODBC;
quit;

** ### THIS IS A SQL STATEMENT TO CREATE A SQL VIEW ###### ;
%PUT %LEFT(NOTE:) Creating staging table for loading &table....%str(;);
%include part1 part2 part3 /source2;
```

APPLY OBJECT PERMISSIONS

- Controlling access to repo objects
  - Apply database permissions when a new table or view is created
  - You can also revoke permission to existing database objects

```
%macro grant_select(schema=,table=);
    proc sql _method feedback exitcode;
    connect to ODBC
    (CONNECTION=GLOBAL
        NOPROMPT="server=&reposerv.;
        DRIVER=ODBC Driver 17 for SQL Server;
        Trusted_Connection=yes;database=&dbname.");
    execute
    (
        GRANT SELECT ON &schema..&table. TO CDR_READER;
    ) by ODBC;
       %put &=sqlxrc;
       %put &=sqlxmsg;
    disconnect from ODBC;
    quit;
%mend;
```

APPLY OBJECT PERMISSIONS

- Macro approach to alter and grant permissions

```
%macro grant_select(schema=,table=);
    proc sql _method feedback exitcode;
    connect to ODBC
    (CONNECTION=GLOBAL NOPROMPT="server=&reposerv.;
        DRIVER=ODBC Driver 17 for SQL Server;Trusted_Connection=yes;database=&dbname.");
    execute
    (
        GRANT SELECT ON &schema..&table. TO CDR_READER;
    ) by ODBC;
       %put &=sqlxrc;
       %put &=sqlxmsg;
    disconnect from ODBC;
    quit;
%mend;

%macro add_timestamp(schema=,table=);
    proc sql _method feedback;
    connect to ODBC
    (CONNECTION=GLOBAL NOPROMPT="server=&reposerv.;
        DRIVER=ODBC Driver 17 for SQL Server;Trusted_Connection=yes;database=&dbname.");
    execute
    (
    IF (SELECT count(*) FROM syscolumns where id=OBJECT_ID(%str(')&schema..&table.%str(')) and name
        BEGIN ALTER TABLE &schema..&table. ADD staged_on DATETIME NOT NULL DEFAULT (getdate()) END
    ) by ODBC;
    disconnect from ODBC;
    quit;
%mend;
```

Thank You !

- José Centeno
- NORC at the University of Chicago
- centeno-jose@norc.org

Trademark Citation