Paper 123-25

Lab*One*'s Production Reporting Environment - Pre-Historic to Pre-Web

Steven Beakley, Lab*One*, Inc., Lenexa, KS

ABSTRACT

LabOne has recently experienced unparalleled diversification and growth of its core businesses. This fact, along with significant changes in the diversity and magnitude of informational needs of LabOne Statistical Engineering clients, both internal and external, has created a paradigm shift in how we process, present, and distribute the requested information. We will discuss the evolution of our departmental goal to provide such value-added statistical and historical information, and the process changes we implemented to make our goal a manageable, auditable reality.

With the implementation of our corporate intranet came a change in our departmental strategy for delivering all this information. The following paper presents the strategy and the steps we took in meeting these objectives prior to our web enablement endeavors.

HISTORY

LabOne began as a laboratory providing health and lifestyle risk assessment testing to insurance underwriters, testing blood, urine or saliva specimens for the likelihood of nicotine addition, substance or alcohol abuse, or other chronic health problems. SAS® was first licensed at LabOne in 1990 as a means of producing statistical and historical reports for clients, and it quickly became apparent that the ability to provide such information to clients was going to be a key differentiator of LabOne over its competitors. Monthly and quarterly reports were implemented summarizing such information as test results as compared to national benchmarks, and laboratory turnaround time on specimen processing. As literally hundreds of clients would be receiving such reports, it became apparent that a robust production environment would be required so the generation of such reports would be efficient and timely.

ENTER STAGE LEFT: SAS/AF

The Statistical Engineering department quickly saw the need to turn over this ongoing reporting responsibility so that they could continue with the task of development to provide clients with other valuable information. The decision was made to turn this responsibility to the computer operations organization, whose primary function was to do regularly scheduled database moves and maintenance. But these computer operators had no knowledge of SAS. So we developed SAS CURE, a front end menu on our VMS operating system using SAS/AF, giving the operators access to the SAS programs required to run and produce these regularly scheduled reports.

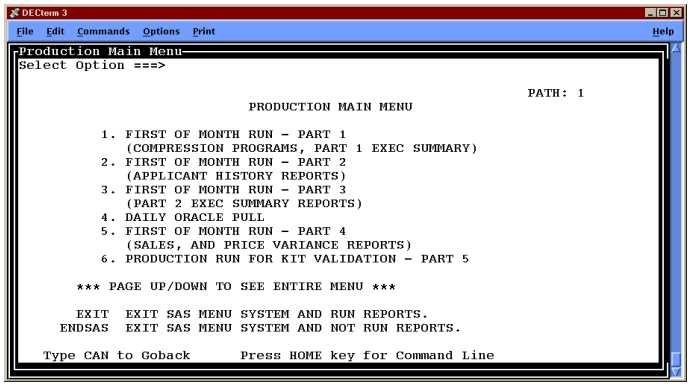

Figure I: Pre-Historic AF Menu Screen

The menu was rudimentary and simple in its use, but provided a great deal of functionality. Menu options were made available for daily processing, which populated a SAS data warehouse, along with weekly and monthly generation of summary datasets and reports; all from the execution of this single AF application. The underlying SCL in the application validated inputs (primarily dates), conditionally built a TEMP.SAS program file (including SAS programming statements and system print commands), and submitted this as a batch job. A sample SCL entry is included as Appendix I to this paper. (The submission of the batch job is omitted from this sample code.)

Some of the limitations of this application in its current state should become very apparent by reviewing the entry. As the number of reports generated grew, the SCL entry grew as well. Even this pared down example contains a significant amount of redundant coding. Maintenance of the application became very cumbersome, whether it be adding or deleting reports, or changing parameters such the number of copies generated or the output destination. Monitoring or querying to determine what reports were run, under what conditions, and delivered to whom was difficult to do. Also, the recipient or recipients of the report had to be manually updated on a static "packing list". With the number of reports being generated now in the hundreds if not thousands, clearly a better solution needed to be implemented.

A DATASET-DRIVEN SOLUTION

The first basic requirement of the solution was that it needed to be consistent with the current application. The number of computer operators was growing as the company grew, and employee turnover had become an issue. So we needed to keep it simple to operate, and ideally, the changes really needed to be transparent to the users. Also, the packing list that they used in facilitating distribution needed to be automated in order to keep it up to date without human intervention.

After several brainstorming sessions, we concluded that the most feasible solution was to modify the AF application to submit a batch job which was generated not from a series of SUBMIT blocks with static code in them as we had done in the past. Instead, we generated the batch job from code that was conditionally built by SAS/MACRO language, based on the values of parameters defined in a controlling dataset.

The SCL code was then stripped down to a basic SUBMIT block that executed a program called PRODSCL:

```
OPTIONS NOXWAIT;
%LET RUNID=TEST;
%LET PRINLOG=Y;
%INCLUDE 'SAS_PGMS:[PROGRAMS]PRODSCL.SAS';
RUN;
```

NOXWAIT returns control to the SAS session automatically after the specified command is executed. This is necessary because of the CALL SYSTEM commands that we process to handle things such as email, file transfer, and printing in our production environment.

The parameters (macro variables) that are required are RUNID, which identifies which set of reports to run (daily, weekly, monthly, post billing, and so forth). PRINLOG determines whether a packing list of the reports will be generated or not. PRODSCL, in turn will build and execute a large batch jobstream based upon the records it identifies in a master dataset called STATLIST, which contains a record or records for each report in production (Appendix II shows a contents of STATLIST. Appendix III of this paper replicates PRODSCL.SAS in a very small font,

due to Proceeding space constraints. A more readable version will be available at my presentation or can be requested from me via e-mail or fax. See contact information below.)

A comment about the use of the option NO$SYNTAXCHECK is in order here. By default, once SAS encounters an error, it will only perform syntax checks on subsequent statements. NO$SYNTAXCHECK allows us to complete subsequent jobs in the job stream even if a previous job errors out (remember, the entire process is a bunch of potentially unrelated SAS programs executing back to back). This way, when we do have a problem in one program in the jobstream, the others will process normally and not require re-running. This is a very handy, but potentially dangerous, feature. If you choose to use it, do so wisely.

While a complete discussion of the processing that PRODSCL does is beyond the scope of this introductory paper, the following outline describes the basic processing that it does:

1. Create current beginning and ending dates for all possible time periods (today, previous day; last week, month, quarter, or year; etc.).

2. Create a current master table of all possible client company ID's for all 3 lines of business along with their company name.

3. Generate a list of the reports that will be run based upon the entries in dataset STATLIST that match the current RUNID.

4. If it has been requested, print the physical packing list for the operators to reference in distributing the upcoming reports as they generate.

5. Count the number of report combinations to be run.

6. For each combination, execute a macro which will:

    6.1. Create macro variables from all the necessary dataset elements: report name, frequency, recipient, number of copies, output type and device, and so forth.

    6.2. String multiple company ID's, test ID's, and state ID's together into one large macro variable so that the report need only be run once for any number of companies, tests, or states.

    6.3. PROC PRINTTO to separate LIS and LOG files so that reports can be printed separately and to facilitate troubleshooting problems with specific reports later.

    6.4. Finally, execute the SAS program.

    6.5. Execute system command(s) to distribute output according to the output device and location specified.

    6.6. Clean up work datasets and macro variables.

    6.7. Continue with next report.

"IT'S A FEATURE, NOT A BUG"

The dataset STATLIST began as a basic control dataset identifying report names, users, and output destinations. As enhancements to the system have been suggested and implemented, the program has become more complex, and the functionality has increased significantly. A few of the kinds of functionality we have been able to introduce at the request or suggestion of our internal and external customers are:

- The addition of color graphics output.
- Forced prioritization of job sequence. This allows us to schedule long running jobs at the end, or to arrange prerequisites to particular jobs.
- Generation of spreadsheet and other supported file formats.
- Automatic generation of emails (both internal and SMTP) containing either textual information or attachments.
- Generation of email notifications updating production report status for IT personnel or recipients.
- Generation of faxed reports to predetermined sites (in conjunction with a "user exit" to an internal application).
- Support for "special day processing"; i.e., determination of day of month and specific program execution based upon that.

FUTURE ENHANCEMENTS

The majority of the enhancements we have planned for this system are web-based and consequently covered in another paper (Paper Number p240-25, in the Posters section). However, there are features which are not necessarily web related that we have considered and hope to implement in the near future. We have automated the process of reporting both text and graphics to paper, email, fax, and, now, web; sufficient output types and delivery methods seem to exist for the near term. Therefore, the scope of the enhancements we are pursuing are maintenance related.

It has become increasingly difficult to proactively monitor the ongoing integrity of each program and its underlying data. We hope to automate the process of identifying such problems as they run, rather than having the recipient of the information alert us to such problems. For example, we hope to produce automated emails that consolidate log error details from the numerous reports that are running. Other means of quickly identifying data inconsistencies are similarly in the works.

CONCLUSION

The intent of the preceding paper has been to describe how LabOne's Statistical Engineering Department has handled the growing need for information by its internal and external customers while keeping the administrative overhead involved to a minimum. Our basic methodology has been presented, along with some ideas for implementation and enhancement. While some code specific to our systems and processes exists, a similar approach could certainly be taken by other organizations trying to accomplish the same thing.

CONTACT INFORMATION

If you have any questions, comments or feedback, please do not hesitate to contact the author at:

Steven Beakley
LabOne,Inc.
10101 Renner Boulevard
Lenexa, Kansas 66219
Work Phone: (913) 577-1380
Fax: (913) 888-4160
Email: steve.beakley@labone.ccom
Web: http://www.labone.com

```
INIT:
*  CALCULATE DATES TO BE DISPLAYED ON INITIAL SCREEN;
   CUR=DATE();
   D=DAY(CUR);
   NEWCUR=CUR-D;
   Y=YEAR(NEWCUR);
   M=MONTH(NEWCUR);
   EDATE=PUT(Y,4.) || PUT(NEWCUR,MMDDYY4.);
   IF M <10 THEN DO;
      BDATE=TRIM(Y)||'0'||TRIM(LEFT(M))||'01';
   END; ELSE DO;
      BDATE=TRIM(Y)||TRIM(LEFT(M))||'01';
   END;
return;

MAIN:
* PERFORM SCREEN INPUT VALIDATION CHECK;
 IF BDATE > EDATE THEN DO;
    ERRORON BDATE EDATE;
    ALARM;
    CURSOR BDATE;
    _MSG_='BEGINNING DATE IS GREATER THAN ENDING DATE';
 END;
 ERROROFF BDATE EDATE;

return;

TERM:
 IF _STATUS_ = 'C' THEN RETURN ;

SUBMIT;

* MONTHLY KIT VALIDATION FOR U.S. INSURERS;

* ASSIGN VARIOUS MACRO VARIABLE INPUTS;
     %LET BDATE="&BDATE";
     %LET EDATE="&EDATE";
     %LET LEGAL=1;        * 1 = U.S.,    6 = CANADA;
     %LET RSORT=1;        * 1 = INSURER, 2 = EXAMINER;
     %LET FREQ="M";       * M = MONTHLY, Q = QUARTERLY;

* ASSIGN OUTPUT LISTING TEXT FILE NAME;
     PROC PRINTTO PRINT='AKITVMC1.LIS' NEW;

* %INCLUDE EXECUTABLE SAS PROGRAM;
 'SAS_PGMS:[PROGRAMS.HIST]AKITVAL.SAS'/SOURCE2;

* REASSIGN OUTPUT LISTING BACK TO THE DEFAULT;
        PROC PRINTTO;

* CLEAN UP WORK DIRECTORIES FOR THE MOST RECENT
EXECUTED PROGRAM;
        PROC DATASETS KILL; RUN;


* MONTHLY KIT VALIDATION FOR U.S. EXAMINERS;

* REASSIGN MACRO VARIABLE INPUTS THAT DIFFER FROM THE
PREVIOUS RUN;
     %LET RSORT=2;        * 1 = INSURER, 2 = EXAMINER;

* ASSIGN OUTPUT LISTING TEXT FILE NAME;
     PROC PRINTTO PRINT='AKITVME1.LIS' NEW;

* %INCLUDE EXECUTABLE SAS PROGRAM;
%INCLUDE 'SAS_PGMS:[PROGRAMS.HIST]AKITVAL.SAS'/SOURCE2;

* REASSIGN OUTPUT LISTING BACK TO THE DEFAULT;
        PROC PRINTTO;

* CLEAN UP WORK DIRECTORIES FOR THE MOST RECENT
EXECUTED PROGRAM;
        PROC DATASETS KILL; RUN;

* AND THE PROCESS WOULD CONTINUE FOR CANADIAN DIVISION
REPORTS...;

* ALL THE MONTHLY REPORTS ARE BEING PRINTED TOGETHER
  TO ELIMINATE OTHER REPORTS BEING PRINTED BETWEEN
THESE;

  DATA _NULL_;
     CALL SYSTEM ("PRINT
AKITVMC1.LIS/QUE=LCOMP_HP5000/FORM=HP_KITVAL");
     CALL SYSTEM ("PRINT
AKITVME1.LIS/QUE=LCOMP_HP5000/FORM=HP_KITVAL");
     CALL SYSTEM ("PRINT
AKITVMC6.LIS/QUE=LCOMP_HP5000/FORM=HP_KITVAL");
     CALL SYSTEM ("PRINT
AKITVME6.LIS/QUE=LCOMP_HP5000/FORM=HP_KITVAL");
  RUN;

 ENDSUBMIT;

* CALCULATE QUARTERLY DATES;

      DATE=MDY(INPUT(SUBSTR(BDATE,5,2),2.),
               INPUT(SUBSTR(BDATE,7,2),2.),
               INPUT(SUBSTR(BDATE,1,4),4.));
 IF MOD(MONTH(DATE),3)=0 THEN DO;
      BMTH=PUT(MONTH(DATE)-2,Z2.);
      EMTH=PUT(MONTH(DATE),Z2.);
      CALL SYMPUT
('NEWDATE',TRIM(SUBSTR(BDATE,1,4))||BMTH||
                              TRIM(SUBSTR(BDATE,7,2)));
      BDATE=SYMGET('NEWDATE');
      PUT _ALL_;
   SUBMIT;

* QUARTERLY KIT VALIDATION FOR U.S. CLIENTS (AFTER
REASSIGNING BEGIN AND END DATES);
     %LET EDATE="&EDATE";
     %LET BDATE="&BDATE";
     %LET LEGAL=1;
     %LET RSORT=1;
     %LET FREQ="Q";

* ASSIGN OUTPUT LISTING TEXT FILE NAME;
     PROC PRINTTO PRINT='AKITVQC1.LIS' NEW;

* %INCLUDE EXECUTABLE SAS PROGRAM;
%INCLUDE 'SAS_PGMS:[PROGRAMS.HIST]AKITVAL.SAS'/SOURCE2;

* REASSIGN OUTPUT LISTING BACK TO THE DEFAULT;
        PROC PRINTTO;

* CLEAN UP WORK DIRECTORIES FOR THE MOST RECENT
EXECUTED PROGRAM;
        PROC DATASETS KILL; RUN;


* QUARTERLY KIT VALIDATION FOR U.S. EXAMINERS;
     %LET RSORT=2;

* ASSIGN OUTPUT LISTING TEXT FILE NAME;
     PROC PRINTTO PRINT='AKITVQE1.LIS' NEW;

* %INCLUDE EXECUTABLE SAS PROGRAM;
%INCLUDE 'SAS_PGMS:[PROGRAMS.HIST]AKITVAL.SAS'/SOURCE2;

* REASSIGN OUTPUT LISTING BACK TO THE DEFAULT;
        PROC PRINTTO;

* CLEAN UP WORK DIRECTORIES FOR THE MOST RECENT
EXECUTED PROGRAM;
        PROC DATASETS KILL; RUN;

* AGAIN, THE PROCESS WOULD CONTINUE FOR CANADIAN
DIVISION REPORTS...;

* ALL THE QUARTERLY REPORTS ARE BEING PRINTED TOGETHER
  TO ELIMINATE OTHER REPORTS BEING PRINTED BETWEEN
THESE;

  DATA _NULL_;
     CALL SYSTEM ("PRINT
AKITVQC1.LIS/QUE=LCOMP_HP5000/FORM=HP_KITVAL");
     CALL SYSTEM ("PRINT
AKITVQE1.LIS/QUE=LCOMP_HP5000/FORM=HP_KITVAL");
     CALL SYSTEM ("PRINT
AKITVQC6.LIS/QUE=LCOMP_HP5000/FORM=HP_KITVAL");
     CALL SYSTEM ("PRINT
AKITVQE6.LIS/QUE=LCOMP_HP5000/FORM=HP_KITVAL");
  RUN;

 ENDSUBMIT;
END;
RETURN;
```

Data Set Name:	DATA.STATLIST	Observations:	2888
Member Type:	DATA	Variables:	32
Engine:	REMOTE6	Indexes:	0
Created:	15:46 Monday, January 10, 2000	Observation Length:	433
Last Modified:	9:01 Friday, January 21, 2000	Deleted Observations:	1
Protection:		Compressed:	NO
Data Set Type:		Sorted:	NO
Label:			

-----Engine/Host Dependent Information-----

Data Set Page Size:	32768
Number of Data Set Pages:	39
File Format:	607
First Data Page:	1
Max Obs per Page:	75
Obs in First Data Page:	65
Filename:	SAS$DRA2:[SAS_DATA]STATLIST.SASEB$DATA
Host Format:	AXP
Disk Blocks Allocated:	2502

-----Alphabetic List of Variables and Attributes-----

#	Variable	Type	Len	Pos	Label
1	BUSID	Char	3	0	BUSINESS SERVICE ID
15	COMPID	Char	4	365	COMPANY ID
18	COPIES	Num	8	371	NUMBER OF COPIES TO BE PRINTED
3	COPYTO	Char	100	19	EMAIL CARBON COPY
21	CPOPT	Char	1	383	COLOR PRINTER OPTION
9	DIR	Char	8	244	PROGRAM DIRECTORY
19	DSET	Char	2	379	DATASET INDICATOR
17	EXECREC	Char	1	370	EXECUTABLE RECORD INDICATOR
20	FACIL	Char	2	381	FACILITY
28	FILETYPE	Char	3	411	ATTACHMENT FILE TYPE
2	FORM	Char	16	3	PAPER FORM
16	FREQ	Char	1	369	REPORT FREQUENCY
4	GPHPRNTR	Char	20	119	GRAPHICS PRINTER
24	INHSE	Char	1	396	INHOUSE REPORT INDICATOR
5	MESSAGE1	Char	80	139	EMAIL MESSAGE LINE 1
29	NOTIFY	Char	1	414	EMAIL NOTIFICATION FLAG
23	PATH	Char	8	388	MENU PATH
12	PRINDEST	Char	20	277	OUTPUT PRINT DESTINATION
26	PRINTER	Char	8	400	OUTPUT PRINTER IDENTIFIER
27	PRIORITY	Char	3	408	PRIORITY INDICATOR
25	PROFILE	Char	3	397	PROFILE
13	RECIP	Char	60	297	REPORT RECIPIENT
8	REG	Char	2	242	REGION CODE
7	REM	Char	3	239	REMARK
10	REPNAME	Char	20	252	REPORT NAME
11	REPORTS	Char	5	272	REPORT PARAMETER
30	RUNDAY	Num	8	415	RUN DAY FOR SPECIAL PROCESSING
14	RUNID	Char	8	357	RUN IDENTIFIER
32	STATE	Char	2	431	STATES TO BE PROCESSED
22	TEST	Char	4	384	TESTS TO BE PROCESSED
6	TYPE	Char	20	219	TYPE PARAMETER

```
*********************************;
*                               *;
*    PROCSCL - BATCH JOB        *;
*    CREATOR FROM LABONE'S      *;
*    SAS_CURE AF MENU.          *;
*                               *;
*********************************;

OPTIONS  NO$SYNTAXCHECK PAGESIZE=59 LINESIZE=165
SORTSIZE=819200;

* A LIBNAME IS ASSIGNED FOR TEMPORARY DATASETS
THAT NEED TO BE MAINTAINED BETWEEN JOBS BUT DO NOT
NEED TO BE KEPT AFTER COMPLETION OF THE ENTIRE
JOB.  NORMAL WORK DATASETS ARE CLEANED UP BETWEEN
JOBS TO ENSURE THAT THE WORK DIRECTORY DOES NOT
BECOME FULL;
LIBNAME SCLTEMP 'SAS_misc:[SAS_TEMP]';
LIBNAME SCLDATA 'SAS_DATA:';

* CREATE A UNIQUE DATASET NAME BASED ON RUNID AND
TIME SO THAT OVERLAPPING PRODSCL RUNS WILL NOT
AFFECT ONE ANOTHER.;
DATA _NULL_;
 RETAIN ONE;
 ONE=' ';

TIME=left(trim(COMPRESS(PUT(TIME(),TIME5.),':')));
 CALL SYMPUT('RUNID2','PROD'||TIME);
 IF "&RUNID" ^IN ('POSTINS','POSTCLS','POSTSAT')
THEN ONE=SYMGET('BPSDATE');
 IF ONE=' ' THEN DO;  /* THIS IS DONE BECAUSE ON
MONTHLY PRODUCTION BPSDATE DOES NOT EXIST */
  CALL SYMPUT('WEEKDAY',WEEKDAY(TODAY()));
  CALL SYMPUT('DOM',DAY(TODAY()));
 END;
 PUT TIME;
RUN;

PROC FORMAT;

* THE FOLLOWING FORMAT IS USED TO GENERATE THE
BEGINNING AND ENDING DATES TO BE PROCESSED FOR
EACH REPORT.  THE VALUES OF THE DATES ARE STORED
IN AN ORACLE TABLE THAT IS UPDATED BY OPERATIONS;
  VALUE $FRQ
    'TODAY'        = 'D'
    'PREV_WEEK'    = 'W'
    'PREV_MONTH'   = 'M'
    'PREV_QUARTER' = 'Q'
    'PREV_YEAR'    = 'Y';

PROC SQL STIMER ;
 CONNECT TO ORACLE
  (USER=MMOORE ORAPW=EROOMM PATH='@SATCEN');

* SELECTS THE DATID AND CURRENT TIME SPAN FOR EACH
DATEID;
  CREATE TABLE DATE AS
  SELECT *
    FROM CONNECTION TO ORACLE
             (SELECT BATCH_PROC_DATE_ID,
                     BATCH_PROC_BEG_DATE,
BATCH_PROC_END_DATE
                FROM BATCH_PROC_DATE_DEF)
   AS D(DATEID,BDATE,EDATE);

QUIT;

PROC SQL STIMER ;
 CONNECT TO ORACLE
  (USER=MMOORE ORAPW=EROOMM PATH='@INSHST');

* SELECTS THE COMPIDS WITH ASSOCIATED COMPANY
NAMES *;
  CREATE TABLE COMPDEF AS
  SELECT *
    FROM CONNECTION TO ORACLE
             (SELECT COMP_ID,COMP_NAME
                FROM COMPANY_DEF)
   AS D(COMPID,CNAME);

QUIT;

* DEFINES THE ENDING DATES FOR THE MONTH AND THE
QUARTER SO AT RUN TIME IT WILL BE POSSIBLE TO SEE
IF MONTHLY OR MONTHLY AND QUARTERLY REPORTS SHOULD
BE RUN;
DATA _NULL_;
  SET DATE;
  IF PUT(DATEID,$FRQ.) IN ('M','Q');
  IF PUT(DATEID,$FRQ.) = 'M' THEN DO;
    CALL SYMPUT("SMDATE",EDATE);
    CALL SYMPUT("SMO",SUBSTR(EDATE,5,2));
    CALL SYMPUT("SYR",SUBSTR(EDATE,3,2));
  END;
  ELSE IF PUT(DATEID,$FRQ.) = 'Q' THEN CALL
SYMPUT("SQDATE",EDATE);
RUN;

* DEFINES COMPIDS AND COMPANY NAMES FOR CLINICAL
AND SAT;
DATA COMPDEF2 (KEEP= COMPID C_NAME
RENAME=(C_NAME=CNAME));
 SET SCLDATA.CCOMPDEF SCLDATA.SCOMPDEF;

* DEFINES A COMPID AND COMPANY NAME FOR AN ALL
COMPANY CODE;
DATA COMPDEF3;
  COMPID='ZZZZ';
  CNAME ='ALL COMPANY';

DATA COMPDEF;
  SET COMPDEF COMPDEF2 COMPDEF3;

* THE FOLLOWING SQL ASSOCIATES A COMPANY NAME WITH
EACH COMPID FOR EACH REPORT FOR THE SELECTED
RUNID;
PROC SQL;

  CREATE TABLE PROCLOG AS
  SELECT DISTINCT
CNAME,SL.COMPID,REPNAME,REG,REM,TYPE,REPORTS,COPIE
S,

FORM,RECIP,PRIORITY,RUNID,FACIL,PRINDEST,FREQ,RUND
AY,
                    NOTIFY,MESSAGE1,COPYTO,FILETYPE
    FROM SCLDATA.STATLIST SL,COMPDEF CD
   WHERE EXECREC ^= 'N'
     AND COPIES > 0
     AND SL.COMPID=CD.COMPID
     AND RUNID = "&RUNID"
%MACRO PB;
     %IF &RUNID=POSTBILL %THEN %DO;
     AND BUSID = "&BUSID"
     %END;
%MEND PB;
%PB;
 ;
QUIT;

* SELECTS THE REPORTS FOR THE REQUESTED RUNID.
ALSO DETERMINES AT RUNTIME IF MONTHLY OR MONTHLY
AND QUARTERLY REPORTS WILL BE RAN. PROCLOG IS THE
DATA USED TO BUILD THE PROCESS LOG AND
SCLTEMP.STATLIST IS THE PRODUCTION RUN DATA;
DATA PROCLOG;
  SET PROCLOG;
  IF COMPID='ZZZZ' AND REG ^= " " THEN CNAME="REG.
REPORT-"||PUT(REG,$REG.);
  IF COMPID='ZZZZ' AND FACIL ^= " " THEN
CNAME="FACILITY REPORT-"||PUT(FACIL,$FAC.);
  IF "&SMDATE" ^= "&SQDATE" AND FREQ='Q' OR RUNID
^="&RUNID" THEN DELETE;
  IF "&RUNID" = "SPECDAY" THEN DO;
   IF RUNDAY=&DOM THEN OUTPUT;
   ELSE IF &WEEKDAY=2 AND RUNDAY=(&DOM-1) THEN
OUTPUT;
   ELSE DELETE;
  END;
  ELSE IF "&RUNID" ^= "SPECDAY" THEN OUTPUT;

DATA SCLTEMP.&RUNID2;
  SET SCLDATA.STATLIST;
  IF COMPID = 'ZZZZ' THEN COMPLEV =
'ALL'||LEFT(TRIM(FACIL));
  ELSE COMPLEV = 'IND';
  IF PRIORITY = " " THEN PRIORITY = "5";
  IF "&SMDATE" ^= "&SQDATE" AND FREQ='Q' OR RUNID
^="&RUNID" THEN DELETE;
  IF "&RUNID" = "SPECDAY" THEN DO;
   IF RUNDAY=&DOM THEN OUTPUT;
   ELSE IF &WEEKDAY=2 AND RUNDAY=(&DOM-1) THEN
OUTPUT;
   ELSE DELETE;
  END;
  ELSE IF "&RUNID" ^= "SPECDAY" THEN OUTPUT;
RUN;

* THE FOLLOWING SQL SELECTS ALL THE DISTINCT
REPNAMES WITH SAME CRITERIA FOR THE SELECTED
RUNID;
PROC SQL;

  CREATE TABLE &RUNID  AS
  SELECT DISTINCT
REPNAME,FREQ,TYPE,DSET,DATEID,DIR,REG,

BDATE,EDATE,FACIL,COPIES,PATH,PRINTER,INHSE,REM,

REPORTS,PROFILE,PRIORITY,FORM,RECIP,PRINDEST,COMPI
EV,
       NOTIFY,MESSAGE1,COPYTO,FILETYPE,GPHPRNTR
    FROM SCLTEMP.&RUNID2,DATE
   WHERE EXECREC ^= 'N'
     AND RUNID = "&RUNID"
%MACRO PB;
     %IF &RUNID=POSTBILL %THEN %DO;
     AND BUSID = "&BUSID"
     %END;
%MEND PB;
%PB
     AND LEFT(TRIM(FREQ)) = PUT(DATEID,$FRQ.);

QUIT;

* DEFINES THE PRIORITY OF THE WAY THE REPORTS WILL
BE LISTED ON THE
  PROCESS LOG AND THE ORDER IN WHICH THE REPORTS
WILL RUN;
DATA &RUNID ;
  SET &RUNID ;
  IF PRIORITY = " " THEN PRIORITY = "5";
  IF PUT(DATEID,$FRQ.)='W' THEN DO;      * ADJUSTS
WEEKLY RUN DATES *;

BDATE=COMPRESS(PUT(INPUT(BDATE,YYMMDD8.)+1,YYMMDD1
0.),'-');

EDATE=COMPRESS(PUT(INPUT(EDATE,YYMMDD8.)+1,YYMMDD1
0.),'-');
  END;


DATA PROCLOG;
  SET PROCLOG;
  IF PRIORITY = " " THEN PRIORITY = "5";
DATA _NULL_;
  CALL SYMPUT("REPCNT","&SQLOBS");

* THE TWO FOLLOWING SORTS SYNCHRONIZES THE ORDER
THAT THE REPORTS WILL BE PRINTED OUT ON THE
PROCESS LOG AND THE ORDER IN WHICH THEY WILL RUN;
PROC SORT DATA=&RUNID  OUT=SCLTEMP.&RUNID ;
  BY PRIORITY RECIP REPNAME REPORTS TYPE COPIES
REM;

PROC SORT DATA=PROCLOG;
  BY PRIORITY RECIP REPNAME REPORTS TYPE COPIES
REM COMPID ;


* THE FOLLOWING MACRO DEFINES THE LOG AND LIS
FILES AND CREATES THE PROCESS LOG (A LISTING OF
THE RECIPIENTS AND REPORTS IN THE ORDER THAT THEY
WILL RUN IN). THE PROCESS LOG ALSO DEFINES IF THE
REPORT IS A FAX, EMAIL, TRANSMISSION. IF THE
REPORT IS TO PRINTED THE PROCESS LOG WILL LIST ON
WHICH PRINTER THE REPORT WILL PRINT;
%MACRO PRINLOG;
%IF &PRINLOG = Y %THEN %DO;

PROC PRINTTO PRINT="PROCLOG_&SMO&SYR..LIS"
                LOG="PROCLOG_&SMO&SYR..LOG" NEW;
%INCLUDE "SAS_PGMS:[PROGRAMS]PROCLOG.SAS";
PROC PRINTTO;

DATA _NULL_;
CALL SYSTEM ("PRINT
PROCLOG_%LEFT(%TRIM(&SMO))%LEFT(%TRIM(&SYR)).LIS/F
ORM=HP_WIDES/COPIES=1/QUE=SAS_PRINT");
RUN;
%END;
%MEND PRINLOG;
%PRINLOG;

OPTIONS LINESIZE=132;

*BEGIN THE LOOP DISTREPT WHICH WILL RUN EACH
REPORT;
 DISTREPT;
%DO SI = 1 %TO &REPCNT;

DATA _NULL_;
    SET SCLTEMP.&RUNID ;
    IF _N_ = &SI;
    CALL SYMPUT("REPNAME",TRIM(REPNAME));
    ...ETC...
* A NUMBER OF STATEMENTS OMITTED HERE, BUT ALL
SIMILAR TO THE ONE ABOVE.  CREATING MACRO
VARIABLES FROM THE STATLIST DATASET VALUES;
RUN;

* THE FOLLOWING SQL SELECTS THE DISTINCT COMPIDS
AND TESTS (IF APPLICABLE) FROM SCLTEMP.STATLIST
WHERE THE CRITERIA LISTED IN THE WHERE IS THE SAME
PROVIDING THE ABILITY TO BUILD ON LIST OF COMPIDS
(DEFINED AS CODE, OCODE AND PCODE LATER) AND A
LIST OF TESTS (DEFINE AS TEST LATER IF APPLICABLE)
SO THAT THE REPORTS CAN BE RUN FOR ALL DESIRED
COMPIDS AND TESTS WITH THE SAME CRITERIA AT ONE
TIME;
PROC SQL;
  CREATE TABLE REP AS
  SELECT DISTINCT SL.COMPID
    FROM SCLTEMP.&RUNID2 SL
   WHERE EXECREC ^= 'N'
     AND RECIP = "&RECIP"
     AND RUNID = "&RUNID"
%MACRO PB;
     %IF &RUNID=POSTBILL %THEN %DO;
     AND BUSID = "&BUSID"
     %END;
%MEND PB;
%PB
     AND PRIORITY="&PRIORITY"
     AND REM=&REM
     AND REPORTS=&REPORTS
     AND COPIES=&COPIES
     AND REPNAME = "&REPNAME"
     AND FREQ = "&FREQ"
     AND TYPE = "&TYPE"
   ORDER BY COMPID;

  CREATE TABLE TESTS AS
  SELECT DISTINCT TEST
    FROM SCLTEMP.&RUNID2
   WHERE EXECREC ^= 'N'
     AND RECIP = "&RECIP"
%MACRO PB;
     %IF &RUNID=POSTBILL %THEN %DO;
     AND BUSID = "&BUSID"
     %END;
%MEND PB;
%PB
     AND RUNID = "&RUNID"
     AND PRIORITY="&PRIORITY"
     AND REM=&REM
     AND REPORTS=&REPORTS
     AND COPIES=&COPIES
     AND REPNAME = "&REPNAME"
     AND FREQ = "&FREQ"
     AND TYPE = "&TYPE"    ;

 CREATE TABLE STATES AS
  SELECT DISTINCT STATE
    FROM SCLTEMP.&RUNID2
   WHERE EXECREC ^= 'N'
     AND RECIP = "&RECIP"
%MACRO PB;
```

```
     %IF &RUNID=POSTBILL %THEN %DO;
     AND BUSID = "&BUSID"
     %END;
%MEND PB;
%PB
     AND RUNID = "&RUNID"
     AND PRIORITY="&PRIORITY"
     AND REM=&REM
     AND REPORTS=&REPORTS
     AND COPIES=&COPIES
     AND REPNAME = "&REPNAME"
     AND FREQ = "&FREQ"
     AND TYPE = "&TYPE"   ;
QUIT;

* THE FUNCTION OF THE FOLLOWING DATASTEP IS AS
STATED:
  BUILDS THE MACRO CODE TO BE USED IN SAS
STATEMENTS (QUOTES AROUND EACH COMPID), BUILDS THE
MACRO OCODE FOR ORACLE STATEMENTS (NO QUOTES),
BULIDS THE MACRO PCODE USED ON THE BANNER PAGE;
DATA _NULL_;
  SET REP END=EOF;
  CALL SYMPUT('C'||LEFT(_N_),TRIM(COMPID));
  CALL
SYMPUT('OC'||LEFT(_N_),"'"||TRIM(COMPID)||"'");
  CALL
SYMPUT('CDE'||LEFT(_N_),'"'||TRIM(COMPID)||'"');
  IF EOF THEN CALL SYMPUT('MAX',LEFT(_N_));
RUN;

%DO SJ = 1 %TO &MAX;
  %IF &SJ = 1 %THEN %DO;
    %LET CODE = "&&C&SJ";
    %LET OCODE= &&OC&SJ;
    %LET GRPCODE= &&OC&SJ;

  %END;
  %ELSE %DO;
    %LET CODE = &CODE,"&&C&SJ";
    %LET OCODE = &OCODE,&&OC&SJ;
    %LET GRPCODE= &&OC&SJ;
  %END;
  %IF &SJ = 1 %THEN %LET PCODE = &&C&SJ;
  %ELSE %LET PCODE = &PCODE, &&C&SJ;
%END;
RUN;

* SETS UP THE MACRO VARIABLE TEST FOR THOSE
REPORTS THAT ARE RUN FOR MULTIPLE TESTS;
DATA _NULL_;
  SET TESTS END=EOF;
  CALL SYMPUT('T'||LEFT(_N_),TRIM(TEST));
  IF EOF THEN CALL SYMPUT('MAXT',LEFT(_N_));
RUN;

%DO SK = 1 %TO &MAXT;
  %IF &SK = 1 %THEN %LET TEST = &&T&SK;
  %ELSE %LET TEST = &TEST &&T&SK;
%END;
RUN;

* SETS UP THE MACRO VARIABLE STATE FOR THOSE
REPORTS THAT ARE RUN FOR MULTIPLE STATES;
DATA _NULL_;
  SET STATES END=EOF;
  CALL
SYMPUT('S'||LEFT(_N_),"'"||TRIM(STATE)||"'");
  IF EOF THEN CALL SYMPUT('MAXS',LEFT(_N_));
RUN;

%DO SL = 1 %TO &MAXS;
  %IF &SL = 1 %THEN %LET STATE = &&S&SL;
  %ELSE %LET STATE = &STATE, &&S&SL;
%END;
RUN;

* DEFINES THE DESTINATION FOR LOG AND LIS FILES BY
THE RECIPIENT AND REPORT NAME;
PROC PRINTTO
PRINT="%LEFT(%TRIM(&RECIP2))_%LEFT(%TRIM(&REPNAME)
).LIS"
%IF &COPIES > 0 %THEN %DO;

LOG="%LEFT(%TRIM(&RECIP2))_%LEFT(%TRIM(&REPNAME)).
LOG"
%END;
                NEW;

* DETERMINES WHICH DATABASE IS USED;
%GLOBAL TEST;
  %IF &DSET=C %THEN %DO;
    %LET SLABONE = INSURANCE;
  %END;
  %ELSE %IF &DSET=CL %THEN %DO;
    %LET SLABONE = C.L.S.;
  %END;
  %ELSE %IF &DSET=SA %THEN %DO;
    %LET SLABONE = S.A.T.;
  %END;
%MEND TEST;

%TEST RUN;

* IF THE REPORT IS TO GENERATE OUTPUT AND IS NOT A
FAX,TRANSMISSION, OR EMAIL THEN A HEADER PAGE IS
BUILT FOR THE REPORT;
%IF &COPIES>0 AND (("&TYPE" ^="AUTOFAX") AND
("&TYPE"^="AUTOTRANS")) %THEN %DO;
DATA _NULL_ ;
  FILE PRINT NOTITLES HEADER=H LINESLEFT=LL;
  R131 = REPEAT('-',130);
  PUT ////// @52 'THE VARIABLES PROCESSED'/
              @52 '-------------------------'////
        @40 "REPORT(S) RECIPIENT: &RECIP"   ///
        @40 "DATABASE: &SLABONE"    //
        @40 "COMPANY CODES:   &PCODE"    /
        @43 "(INDIVIDUAL COMPANY CODES OR 'ZZZZ'
FOR FACILITY OR ALL COMPANIES)"//
        @40 "FACILITY CHOICE:      " &FACIL/
        @43 "(PROVIDED ONLY IF SUMMARY BY FACILITY
REQUESTED)"/
        @43 'CHOICES: (10)-QUIVIRA   (12)-CANADA'/
        @43 '         (14)-S.A.T.    (16)-
CLINICAL'//
        @40 "BEGINNING DATE (YYYYMMDD): " &BDATE /
        @40 "ENDING DATE (YYYYMMDD):    " &EDATE
////
        @30 'YOU MAY HAVE NO RECORDS FOR PART OR
ALL OF THIS REPORT.'/
        @30 'PLEASE VERIFY THE ABOVE PARAMETERS
BEFORE CONTACTING SAS ON-CALL.';
RETURN;
H:  PAGENO + 1;
* THESE VARIABLES ARE DEFINED FOR HEADERS IN THE
NULL
NONE SHOULD NEED CHANGING AS TITLE LINE IS READ
FROM MACRO VARIABLE;
    HEAD1  = 'LabOne, Inc.';
    HEAD3  = "SAS STATISTICS NOTIFICATION FOR
REPORT = &RPTNAME";

   PUT @1 HEAD1 $131.-C;

   PUT  @1 HEAD3 $131.-C
        @118 'PAGE:    ' PAGENO 6.;
   PUT  @1 "SAS RPT:  &RPTNAME"
        @118 "DATE:  &SYSDATE";
   PUT  @1   'PATH:   '
        @118 "TIME:    &SYSTIME" ;
           PUT @1 R131 ;

RETURN;
RUN;
;

*EXECUTE THE EXTERNAL SAS PROGRAM;

"SAS_PGMS:[PROGRAMS.%LEFT(%TRIM(&DIR))]%LEFT(%TRIM
(&REPNAME)).SAS";

* DETERMINES THE NUMBER OF OBSERVATIONS IN THE
LAST DATA SET CREATED BY THE JOB;
PROC SQL;
 RESET NOPRINT;
 SELECT COUNT(*) INTO:NUMOBS
 FROM _LAST_;

* IF THE REPORT IS TO BE FAXED AND THERE WHERE NO
OBSERVATIONS SENDS A FAX STATING THAT THERE WAS NO
DATA GENERATED FOR THE TIME FRAME THE TIME PERIOD
IN WHICH THE REPORT WAS RUN;
DATA _NULL_;
 IF &NUMOBS=0 AND "&TYPE"="AUTOFAX" THEN DO;
  %INCLUDE 'SAS_FORMATS:NULLFAX.SAS';
 END;
RUN;

* ENDS THE DESTINATION FOR THE LIS AND LOG FILES;
PROC PRINTTO FAX;
  %IF (("&TYPE" ="AUTOFAX") OR
("&TYPE"="AUTOTRANS")) %THEN %DO;
  FILENAME OLD
"%LEFT(%TRIM(&RECIP))_%LEFT(%TRIM(&REPNAME)).LIS"
;
  FILENAME FAX
"%LEFT(%TRIM(&RECIP))_%LEFT(%TRIM(&REPNAME)).FAX"
;
  DATA ONE ;
   INFILE OLD RECFM=V LENGTH=LG;
   INPUT FAXREC $VARYING132. LG;
   FORMFEED='0C'X;
   LGT=LG;

  DATA _NULL_;
  SET ONE;
  FILE FAX NOPRINT;
  IF FAXREC = FORMFEED THEN DELETE;
  ELSE DO;
  LEN=LENGTH (FAXREC);
  PUT FAXREC $VARYING132. LEN;
  END;
  RUN;
%END;
%MEND FAX;
%FAX;

* CREATES A FILE OF THE REPORTS WHERE THE NUMBER
OF OBS IN THE LAST DATA SET CREATED WAS 0 AND THE
COPIES => 1;
%MACRO EMPTY;
 DATA _NULL_;
  %INCLUDE 'SAS_FORMATS:EMPTY.SAS';
 RUN;
%MEND EMPTY;

%IF &COPIES =>1 AND &NUMOBS=0 AND (("&FILETYPE" ="
") OR (&FILETYPE="LIS")) %THEN %DO;
 %EMPTY;

* CREATES THE .FTM FILE WITH AN ATTACHMENT THAT
WILL BE USED FOR FTP MAIL.  USE IS TO SEND AN
EMAIL WITH AN ATTACHMENT (.XLS .DBF .HTM ETC...);
%GLOBAL REPNAME2;
%MACRO EMAIL;
  %LET
REPNAME2=%LEFT(%TRIM(&RECIP2))_%LEFT(%TRIM(&REPNAM
E));
 DATA _NULL_;
  %INCLUDE 'SAS_FORMATS:EMAIL.SAS';
 RUN;
%MEND EMAIL;

%IF "&TYPE"="EMAIL" %THEN %DO;
 %EMAIL;

* CREATES THE .FTM FILE WITHOUT AN ATTACHMENT THAT
WILL BE USED FOR FTP MAIL USE IS TO SEND A
NOFICATION THEIR JOB HAS COMPLETED AND WHERE THE
OUTPUT IS;
%MACRO NOTIFY;
 DATA _NULL_;
  %INCLUDE 'SAS_FORMATS:NOTIFY.SAS';
 RUN;
%MEND NOTIFY;

 "&NOTIFY"="Y" %THEN %DO;
 %NOTIFY RUN;
;

* CLEAN UP ALL WORK DATASET FOR THIS REPORT;
PROC DATASETS KILL;
RUN;

* THE FOLLOWING DATA STEP IS USED TO CALL SYSTEMS
TO EITHER PRINT, FAX EMAIL, OR TRANSMIT THE
REPORTS;
DATA _NULL_;
IF "&NOTIFY"="Y" THEN DO;
  CALL SYSTEM
 ("FTP_MAIL
%LEFT(%TRIM(&RECIP2))_%LEFT(%TRIM(&REPNAME)).FTM")
;
END;
IF "&GPHPRNTR" IN
("SALES_COLOR","SALES_COLOR1","SAS_COLOR","CSALES_
COLOR") THEN DO;
   CALL SYSTEM
 ("PRINT
%LEFT(%TRIM(&RECIP2))_%LEFT(%TRIM(&REPNAME)).GPH/Q
UE=&GPHPRNTR/PASSALL/NOFEED/FORM=FULL/COPIES=%LEFT
(%TRIM(&COPIES))");
END;
IF &COPIES => 1 THEN DO;
 IF "&PRINDEST" IN ("SAS_PRINT","SAS_PRINT1") AND
INDEX("&FORM",'3')>0 THEN DO;
   CALL SYSTEM
 ("PRINT
%LEFT(%TRIM(&RECIP2))_%LEFT(%TRIM(&REPNAME)).LIS/Q
UE=&PRINDEST/FORM=&FORM/COPIES=%LEFT(%TRIM(&COPIES
))/HOLD");
 END;
 ELSE IF "&TYPE" ^IN("AUTOFAX","AUTOTRANS") THEN
DO;
   CALL SYSTEM
 ("PRINT
%LEFT(%TRIM(&RECIP2))_%LEFT(%TRIM(&REPNAME)).LIS/Q
UE=&PRINDEST/FORM=&FORM/COPIES=%LEFT(%TRIM(&COPIES
))&PRINTER");
 END;
 IF "&TYPE"="AUTOFAX" THEN DO;
   CALL SYSTEM
 ("COPY
%LEFT(%TRIM(&RECIP))_%LEFT(%TRIM(&REPNAME)).FAX

BPS::RECV_TEMP:SAS_XMT_%LEFT(%TRIM(&RECIP))_%LEFT(
%TRIM(&REPNAME))_&SYSDATE..BPS");
 END;
 ELSE IF "&TYPE"="AUTOTRANS" THEN DO;
   CALL SYSTEM
 ("COPY
%LEFT(%TRIM(&RECIP))_%LEFT(%TRIM(&REPNAME)).DAT

BPS::RECV_TEMP:SAS_XMT_%LEFT(%TRIM(&RECIP))_%LEFT(
%TRIM(&REPNAME))_&SYSDATE..BPS");
 END;
 ELSE IF "&TYPE"="EMAIL" THEN DO;
 IF "%LEFT(%TRIM(&FILETYPE))" IN
("TXT","DAT","CSV","LIS") THEN CALL SYSTEM
 ("FTP_MAIL
%LEFT(%TRIM(&RECIP2))_%LEFT(%TRIM(&REPNAME)).FTM

%LEFT(%TRIM(&REPNAME2)).%LEFT(%TRIM(&FILETYPE))");
  ELSE CALL SYSTEM
 ("FTP_MAIL
%LEFT(%TRIM(&RECIP2))_%LEFT(%TRIM(&REPNAME)).FTM

%LEFT(%TRIM(&REPNAME)).%LEFT(%TRIM(&FILETYPE))
IMAGE");

 END;
END;

%END;
%MEND DISTREPT;
%DISTREPT;
RUN;

* LOOP DISTREPT COMPLETE, ALL REPORTS HAVE RUN.
CLEAN UP TEMPORARY DATASET THAT WERE CREATED FOR
THIS RUN;
PROC DATASETS LIB=SCLTEMP;
  DELETE &RUNID &RUNID2;
QUIT;

RUN;
```

Appendix III: PRODSCL Code