A Table Driven ODS Macro

Diane E. Brown, eXponential Systems, Indianapolis, IN

ABSTRACT

Tired of coding ODS statements and SAS® output procedures for every report you write and having redundant or similar code everywhere? This paper will present a plan to reuse SAS output source code using a SAS table to drive the process and macros to execute the process. A complete review of the table design and content, and the source code will be provided. User controls available in the macro include: 1) variables to output, 2) the output styles; pdf, html, txt, SAS, excel, rtf, mdb, etc., 3) ODS style, 4) titles and footnotes, and 5) SAS output procedure. This routine also addresses the common problem situations: 1) Zero observations and no report output is generated, and 2) Too many observations and output is too large.

INTRODUCTION

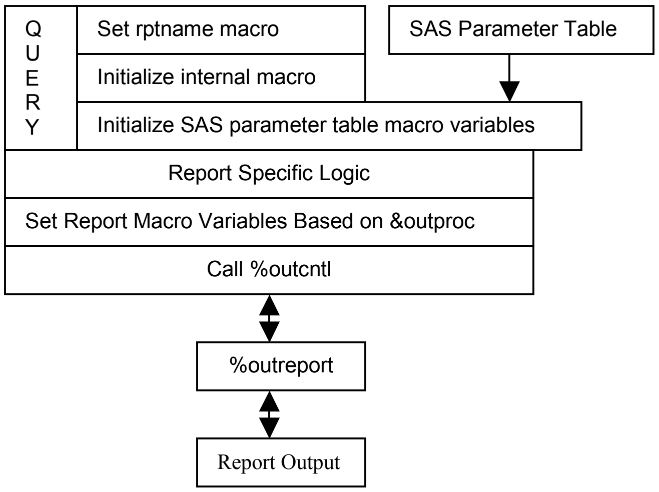
The table driven ODS macro approach was developed to support a SAS/AF application for healthcare reporting and analysis called eXpert Medical Analyst or EMA. EMA allows the user to generate a report in one or more output styles including pdf, html, txt, SAS, excel, rtf (Word), and mdb (Microsoft Access). EMA report parameters are stored in a SAS parameter table as default values. The user interface allows users to select from or override some of these default parameters. The EMA main menu displays the valid output style based on the SAS parameter table; the user then selects from this list. Figure 1 shows EMA based sample reports generated using the table driven ods macro.

For this paper, a scaled back version of the table driven ODS macro is presented without parameters from the SAS/AF interface. Parameter values stored in the SAS table are the control for report behavior and set the output styles desired. This version of the table driven ODS macro can be used outside an interface in batch mode or submitted in the SAS windows environment. This paper will present the design, the methodology, the SAS table definition, and the source code to allow you to implement this approach in your application or even used in an adhoc report environment.

PROGRAM FLOW

The table driven ODS macro consists of 4 components: 1) a SAS table containing report parameters, 2) macro source that is stored as compiled macros, 3) report source, and 4) report destination. The following diagram shows the program flow within the report source program. The Query on the left denotes the %query macro which performs the functions on the right. The %outcntl macro calls the %outreport macro which generates an output file for each output style requested.

Report Source Flow


Q U E R Y
Set rptname macro
Initialize internal macro
SAS Parameter Table
Initialize SAS parameter table macro variables
Report Specific Logic
Set Report Macro Variables Based on &outproc
Call %outcntl
%outreport
Report Output

DIRECTORY STRUCTURE

The following directory structure supports the table driven ODS macro. Directories shown are inside c:\ema_sugi. This structure is merely for demo purposes and can easily be altered to fit into your application.

Sub Directory	File	Notes
	Sugi_autoexec .sas	Autoexec
	Sugi_config_v 8.sas	Sets up bangs
\macro_source	Compile_sugi	Compiles all macros in \macro_source
\macro_source \build_input	Query.sas	Source to initialize macro variables and queries data
\macro_source \build_output	Frequency_out .sas	PROC FREQ source
	Outcntl.sas	Source called by reports to invoke report build
	Outreport.sas	Source to build reports
\macro_source \tools	Dsn_obs	Source to obtain number of obs in sas data set
\macros	Sasmacr.sas7 bcat	Compiled macros from \macro_source
\output	various	Report destination
\parms	Reports.sas7b dat	SAS table containing report parameters
\reports	various	Source for each report

SAS PARAMETER TABLE

The table component of the table driven ODS macro is a SAS data set with one row per report and variables that set report attributes. The following variables and their purpose are:

Variable	Purpose
Rptname	Report name in \reports
Datain	Libref.sasdatafile
Whereclause	Optional Where clause
Outfile	Fully qualified output file directory and file name without file extension
Outproc	SAS output PROC
Maxin	Optional maximum number of records from query
Maxout	Optional maximum number of records to output
Csv	1=csv output, 0=no csv output
Csvstyle	ODS style for csv output
Htm	1=html output, 0=no html output
Htmstyle	ODS style for html output
Pdf	1=pdf output, 0=nopdf output
Pdfstyle	ODS style for pdf output
Rtf	1=rtf output, 0=no rtf output
Rtfstyle	ODS style for rtf output
Txt	1=txt output, 0=no txt output
Mdb	1=mdb output, 0=no mdb output
Sas	1=sas output, 0=no sas output

The SAS parameter table used in this paper is shown in Figure 2.

SAS STARTUP

AUTOEXEC_SUGI.SAS

This autoexec or an autoexec with these statements added can be added to the command in the properties of the SAS icon to automatically be executed when SAS is brought up. The autoexec can also be included in the SAS program window after SAS is brought up.

```
options mstored sasmstore=macros missing = ' '
       mprint symbolgen;
 libname  datain   '!EMA_data/sasdw/current';
 libname  library   '!EMA_data/sasdw/formats';
 libname  rptcntl   '!EMA_sugi/parms';
 filename reports   '!EMA_sugi/reports';
 libname  macros  '!EMA_sugi/macros'
        access=readonly;
```

SUGI_ CONFIG_V8.CFG

Add the following to your existing SAS config file or create a new config file. If a new config, it must be added to the command properties of the SAS icon to automatically be executed when SAS is brought up. The config can not be executed after SAS is brought up. Setting the bangs is not necessary if you do not o use the bangs in the autoexec.

```
/* Set bangs */
-SET ema_sugi     C:\ema_sugi
-SET ema_data     C:\ema_data
```

MACRO SOURCE

The source code for all macros in c:\ema_sugi \macro_source is shown in this section. These are all stored as compiled macros in c:\ema_sugi\macros.

COMPILE_SUGI.SAS

This program merely compiles all macro source programs.

```
libname emamac  '!ema_sugi/macros';
options mstored sasmstore = emamac;
filename macinp   '!ema_sugi/macro_source/build_input';
filename macout   '!ema_sugi/macro_source/build_output';
filename mactools '!ema_sugi/macro_source/tools';
%include macinp(query);
%include macout(outcntl);
%include macout(outreport);
%include macout(frequency_out);
%include mactools(dsn_obs);
```

%QUERY

The query module accomplishes 3 tasks: 1) initializes internals used by the stored compiled macros, 2) initializes macro variables created from the SAS parameter table, and 3) queries the input data to be used for the report. In this paper, the input data is assumed to be a SAS data set, but again modifications can be made to support other databases.

```
%macro query (keepquery=) / store;

/* Initialize internal macros */
%global query_obs out_obs query_exceeded
           byvar bodyvar frqWithin frq1Var frq2Var
           frqOrder frqStyle frqWay  frqFreq  frqPct
           frqrPct frqcPct frqWeight frqcum
           ;

   /* job control */
   %let query_obs = 0;
   %let out_obs   = 0;
   %let query_exceeded = 0;

   /* Proc Print */
   %let byvar     = ;
   %let bodyvar   = ;

   /* Proc Frequency */
   %let frqWithin = ;
   %let frq1Var   = ;
   %let frq2Var   = ;
   %let frqOrder  = ;
   %let frqStyle  = ;
   %let frqWay    = 1;
   %let frqFreq   = ;
   %let frqPct    = ;
   %let frqrPct   = ;
   %let frqcPct   = ;
   %let frqWeight = ;
   %let frqcum    = ;

/* Initialize SAS table macros */
%global maxin maxout outproc whereclause csv csvstyle htm
          htmstyle pdf pdfstyle rtf rtfstyle txt mdb sas outfile datain;

   data _null_;
   set rptcntl.reports;
   where upcase(rptname) = upcase("&rptname");
   call symput('maxin',left(put(maxin,10.)));
   call symput('maxout',left(put(maxout,10.)));
   call symput('outproc',left(upcase(outproc)));
   call symput('whereclause',left(whereclause));
   call symput('csv',left(put(csv,1.)));
   call symput('csvstyle',left(csvstyle));
   call symput('htm',left(put(htm,1.)));
   call symput('htmstyle',left(htmstyle));
   call symput('pdf',left(put(pdf,1.)));
   call symput('pdfstyle',left(pdfstyle));
   call symput('rtf',left(put(rtf,1.)));
   call symput('rtfstyle',left(rtfstyle));
   call symput('txt',left(put(txt,1.)));
   call symput('mdb',left(put(mdb,1.)));
   call symput('sas',left(put(sas,1.)));
   call symput('outfile',trim(left(outfile)));
   call symput('datain',left(datain));
  run;

%mend query;
```

%FREQUENCY_OUT

This macro is used to generate output using PROC FREQ. PROC FREQ is used to generate output when the SAS parameter variable, &outproc, is set to FREQ.

```
%macro Frequency_out / store;

  %if &frqWithin ^= %str() %then %do;
    proc sort data=_emadata;
     by
       %if &frqWithin ^= %str() %then %do;
         &frqWithin
       %end;
         &frq1Var
       %if &frq2Var ^= %str() %then %do;
         &frq2Var
       %end;
        ;
    run;
  %end;

  proc freq data=_emadata notitle
   %if "%upcase(&frqOrder)" = "FREQ" %then %do;
       order=freq
   %end;
   %else %do;
       /*order=data*/
       order=internal
   %end;
   ;
   %if &frqWithin ^= %str( ) %then %do;
     by &frqWithin;
   %end;

   tables
    %if "&frqWay" = "1" %then %do;
        &frq1Var
    %end;
    %else %do;
        &frq1Var  * &frq2var
    %end;
    / missing
      %if "%upcase(&frqstyle)" = "LIST" %then %do;
         LIST
      %end;
      %if "&frqWay" = "1" %then %do;
         %if "&frqCum" = "No" %then %do;
            NOCUM
         %end;
      %end;
      %else %do;
        %if "%upcase(&frqStyle)" = "LIST" %then %do;
           %if "&frqCum" = "No" %then %do;
              NOCUM
           %end;
        %end;
        %if "&frqFreq"="No" %then %do;
           NOFREQ
        %end;
        %if "&frqRPct"="No" %then %do;
           NOROW
        %end;
        %if "&frqCPct"="No" %then %do;
           NOCOL
        %end;
      %end;
      %if "&frqPct"="No" %then %do;
         NOPERCENT
      %end;
       ;
      %if &frqWeight ^= %str() %then %do;
        WEIGHT &frqWeight;
      %end;
  run;

 %mend  Frequency_Out;;
```

%OUTCNTL

The %outcntl macro determines the number or observations in the output SAS file in order to insure that value does not exceed the &maxout macro variable. The &maxout macro variable is created from the SAS parameter table as the maximum number of observations allowed in the output report. Note that if &outproc is FREQ, the value of &maxout should be missing, as the number of observations going into PROC FREQ is not equal to the number of observations going out of PROC FREQ. Next, %outcntl calls %outreport for each output style set to 1 in the SAS parameter table (csv, htm, pdf, rtf, txt, mdb, and sas).

```
%macro outcntl / store;

/* Determine if _emadata has observations */
   %dsn_obs(_emadata,out_obs)

/* Call output styles */
   %if &pdf  = 1 %then %outreport(outtype=PDF);
   %if &csv  = 1 %then %outreport(outtype=CSV);
  %if &htm  = 1 %then %outreport(outtype=HTM);
   %if &rtf  = 1 %then %outreport(outtype=RTF);
   %if &txt  = 1 %then %outreport(outtype=TXT);
   %if &mdb  = 1 %then %outreport(outtype=MDB);
   %if &sas  = 1 %then %outreport(outtype=SAS);

  %mend outcntl;
```

%OUTREPORT

Here's the macro that is the heart of the table driven ODS macro, %outreport. The step 1 is to generate the beginning ODS options depending on the output style called the %outcntl macro. Notice the file= in one of the ods statements. It uses the &outfile macro variable created from the SAS parameter table. &outfile is the fully qualified output file name excluding the file extension. The file extension is added based on the output style. Additionally the style= option uses the style from the SAS parameter table based on the output style. The output styles, txt, mdb, and sas, do not use ODS, but are a part of the %outreport macro so that these types of output can also be generated. For txt, PROC PRINTTO is used to direct the output destination. For sas, setup code here parses the macro variable, &outfile, to separate the directory from the file name, creating &outlib and &sasdsn.

Program steps 2, 3, and 4 are very similar in code. Step 2 is executed when the output SAS data set contains no observations; step 3 is executed when the number of observations resulting from the query exceeded the &maxin macro variable; and step 4 is executed when the number of observations in the output SAS data set exceeded the &maxout macro variable.

Step 5 is executed when the number of observations has passed through steps 2, 3, and 4. Step 5 produces the actual output. First, Microsoft Access (mdb) and SAS output is handled, then logic is driven by the outproc macro variable. Microsoft Access source code requires SAS/ACCESS to PC File Formats®. SAS output logic uses PROC DATASETS to rename the SAS file and copy it to the location specified in the &outfile macro variable created from the SAS parameter table.

When &outproc is PRINT, the PROC PRINT source code is contained here in %outreport. Other values of& outproc, call a macro to build the output. Included in this presentation is %frequency_out only.

```
 %macro outreport (outtype=) / store;

/* 1 Set ODS options */
  %if %upcase(&outtype) = CSV %then %do;
     ods noresults;
     ods html file="&outfile..csv" style=&csvstyle;
     ods listing close;
  %end;
  %else %if %upcase(&outtype) = HTM %then %do;
     ods noresults;
     ods html file="&outfile..htm" style=&htmstyle;
     ods listing close;
  %end;
  %else %if %upcase(&outtype) = PDF %then %do;
      ods noresults;
      ods pdf file="&outfile..pdf" style=&pdfstyle notoc nopdfnote;
      ods listing close;
  %end;
  %else %if %upcase(&outtype) = RTF %then %do;
     ods noresults;
     ods rtf file="&outfile..rtf" style=&rtfstyle;
     ods listing close;
  %end;
  %else %if %upcase(&outtype) = TXT %then %do;
     ods listing;
     proc printto print = "&outfile..txt";
     run;
  %end;
  %else %if %upcase(&outtype) = SAS %then %do;
     %let fileleng = %length(&outfile);
     %do i=&fileleng %to 1 %by -1;
        %let val = %substr(&outfile,&i,1);
        %if &val = %str(\) or &val = %str(/) %then %do;
           %let saslib = %substr(&outfile,1,&i);
           %let sasdsn = %substr(&outfile,&i+1);
           %let i = 1;
        %end;
      %end;
      libname _sasout "&saslib";
  %end;

/*  Set pageno */
  options pageno=1;
  run;

/*  2 Produce report with message for 0 observations */
  %if &query_obs = 0 %then %do;
     data _emadata;
       length message $35;
       label message = 'Message';
       message = 'No data selected for WHERE clause';
       output;
     run;
     %if %upcase(&outtype) = MDB %then %do;
        proc export data= _emadata
          outtable= data
          dbms=ACCESS2000 replace;
          database="&outfile..mdb";
        run;
     %end;
     %else %if %upcase(&outtype) = SAS %then %do;
       proc datasets library = work nolist nodetails;
         delete &sasdsn;
         change _emadata = &sasdsn;
         copy in=work out=_sasout;
         select &sasdsn;
       run;
       quit;
     %end;
     %else %do;
       proc print data = _emadata noobs split='*';
         var message;
         label message = 'Message';
       run;
     %end;
  %end;

/* 3 Report with message for exceeded max obs for input */
  %else %if &query_exceeded = 1 %then %do;
    data _emadata;
      length message $55;
      label message = 'Message';
      message = 'Maximum number of rows for INPUT
                 exceeded.';
      output;
      message = "Maximum allowed is: %trim(&maxin)";
      output;
      message = "Query was halted";
      output;
    run;
     %if %upcase(&outtype) = MDB %then %do;
        proc export data= _emadata
          outtable= data
          dbms=ACCESS2000 replace;
          database="&outfile.mdb";
        run;
     %end;
     %else %if %upcase(&outtype) = SAS %then %do;
       proc datasets library = work nolist nodetails;
         delete &sasdsn;
         change _emadata = &sasdsn;
         copy in=work out=_sasout;
         select &sasdsn;
       run;
       quit;
     %end;
     %else %do;
       proc print data = _emadata noobs split='*';
          var message;
          label message = 'Message';
       run;
     %end;
  %end;
```

```
/*4  Report with message for exceeded max obs for output */
  %else %if &out_obs > &maxout and &maxout ^= %str( ) %then
  %do;
    data _emadata;
      length message $75;
      label message = 'Message';
      message = 'Maximum number of rows for OUTPUT
            exceeded.';
      output;
      message = "Maximum allowed is: %trim(&maxout)";
      output;
      message = "Actual is: %trim(&out_obs)";
      output;
    run;
    %if %upcase(&outtype) = MDB %then %do;
       proc export data= _emadata
          outtable= data
          dbms=ACCESS2000 replace;
          database="&outfile..mdb";
       run;
    %end;
    %else %if %upcase(&outtype) = SAS %then %do;
       proc datasets library = work nolist nodetails;
         delete &sasdsn;
         change _emadata = &sasdsn;
         copy in=work out=_sasout;
         select &sasdsn;
       run;
       quit;
    %end;
    %else %do;
       proc print data = _emadata noobs split='*';
         var message;
         label message = 'Message';
       run;
     %end;
  %end;

/* 5  Produce report for observations */
  %else %do;
    %if %upcase(&outtype) = MDB %then %do;
       proc export data= _emadata
          outtable= data
          dbms=ACCESS2000 replace;
          database="&outfile..mdb";
       run;

    %end;
    %else %if %upcase(&outtype) = SAS %then %do;
       proc datasets library = work nolist nodetails;
         delete &sasdsn;
         change _emadata = &sasdsn;
         copy in=work out=_sasout;
         select &sasdsn;
       run;
       quit;
    %end;
    %else %if %upcase(&outproc) = PRINT %then %do;
         proc print data = _emadata noobs label;
         %if &byvar ^= %str( ) %then %do;
            by &byvar notsorted;
         %end;
         %if &bodyvar ^= %str( ) %then %do;
            var &bodyvar;
         %end;
       run;
    %end;
     %else %if %upcase(&outproc) = REPORT %then %do;
         %Report_out
    %end;
    %else %if %upcase(&outproc) = FREQ %then %do;
         %Frequency_out
    %end;
    %else %if %upcase(&outproc) = MEANS %then %do;
         %Means_out
    %end;
    %else %if %upcase(&outproc) = MAP %then %do;
         %map_out
    %end;
    %else %if %upcase(&outproc) = PLOT %then %do;
         %plot_out
    %end;
    %else %if %upcase(&outproc) = BAR %then %do;
         %bar_out
    %end;
  %end;

/*  6 Close ODS */
  %if %upcase(&outtype) = PDF %then %do;
    ods pdf close;
    run;
  %end;
  %else %if %upcase(&outtype) = HTM  or %upcase(&outtype) =
     CSV %then %do;
    ods html close;
    run;
  %end;
  %else %if %upcase(&outtype) = RTF %then %do;
    ods rtf close;
    run;
  %end;
  %else %if %upcase(&outtype) = TXT %then %do;
    proc printto print = print;
    run;
    ods listing close;
    run;
  %end;
 %mend outreport;
```

%DSN_OBS

This is a utility macro that is part of the tool kit used by eXponential Systems. A data set is specified in the &dsn parameter, and a macro variable name is specified in the &obsvar parameter. This allows the programmer to name the macro variable to hold the number of observations in the SAS data set.

```
%macro dsn_obs(dsn,obsvar) / store;
   %if &obsvar = %str( ) %then %let obsmac = dsn_obs;
   %else %let obsmac = &obsvar;
   %global &obsmac;
   %let &obsmac = 0;
   data _null_;
     if 0 then set &dsn nobs = dsn_obs;
     call symput("&obsmac",left(put(dsn_obs,15.)));
     stop;
     run;
 %mend dsn_obs;
```

REPORT SOURCE

Report source code must start by setting the macro variable, &rptname equal to name the same value as the variable report name on the SAS parameter table. It is recommended that the source code file also be this same value. Next, the query macro is called. Query initializes internal macro variables and initializes macro variables created from the SAS parameter table by selecting the row where& rptname equals the &rptname macro variable. Query also queries the input SAS file in &datain macro variable with the &whereclause macro variable, both from the SAS parameter table. For computer performance, the report specific code is inside a %if that will execute only if the observation count resulting from the query is less than the &maxin macro variable. &maxin is the maximum number of observations resulting from the query allowed, as set in the SAS parameter table. For simple reports, this report specific logic may not exist at all. The last report source step sets the report titles and macro variables required for output variable definition. If &outproc from the SAS parameter table is PRINT or REPORT, macro variables, &byvar and &bodyvar are required. If &outproc is FREQ, several macro variables as shown in the initialization step of the Query source code, are required.

REPORT - HEART_PROC.SAS

```
%macro reportit;

/* Read claim data */
  %let rptname = HEART_PROC;
  %query(keepquery = proc subchg allowed_amt
         nbr_serv);

/* Continue if max input rows is not exceeded */
 %if &query_obs <= &maxin or &maxin = %str( )
    %then %do;
   /* Summarize data */
   proc summary data = _emadata nway;
       class proc;
       var subchg allowed_amt nbr_serv;
       output out = _emadata sum=;
     run;
     proc sort data = _emadata;
       by descending allowed_amt;
     run;
  %end;

/* Output */
  title1 'Congestive Heart Failure Procedures by
         Allowed Amount';
  %let byvar   = ;
  %let bodyvar = proc subchg allowed_amt
                 nbr_serv;
  %outcntl;
  run;

%mend reportit;
%reportit
```

REPORT - SPEC_FREQ.SAS

```
%macro reportit;

/*  Read claim data    */
  %let rptname = SPEC_FREQ;
  %query(keepquery = spec sex allowed_amt);

/*  Output  */

  title1 'Frequency of Non-Institutional Specialties';

  %let frqWithin = sex;
  %let frq1Var   = spec;
  %let frq2Var   = ;
  %let frqOrder  = freq;
  %let frqStyle  = ;
  %let frqWay    = 1;
  %let frqFreq   = ;
  %let frqPct    = ;
  %let frqrPct   = ;
  %let frqcPct   = ;
  %let frqWeight = allowed_amt;
  %outcntl;
  run;

%mend reportit;
%reportit
```

CONCLUSION

The table driven ODS macro saves code redundancy, changes in report control can be made easily by updating values in the SAS parameter table, and the macro source can be expanded as requirements change. This approach can be implemented in any application used in an adhoc environment. This allows the programmer to focus on the report itself, the data content, the data manipulation requires, and the business rules – and not on repeatedly writing similar soruce code.

TRADEMARK CITATION

SAS® and all other SAS Institute Inc product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

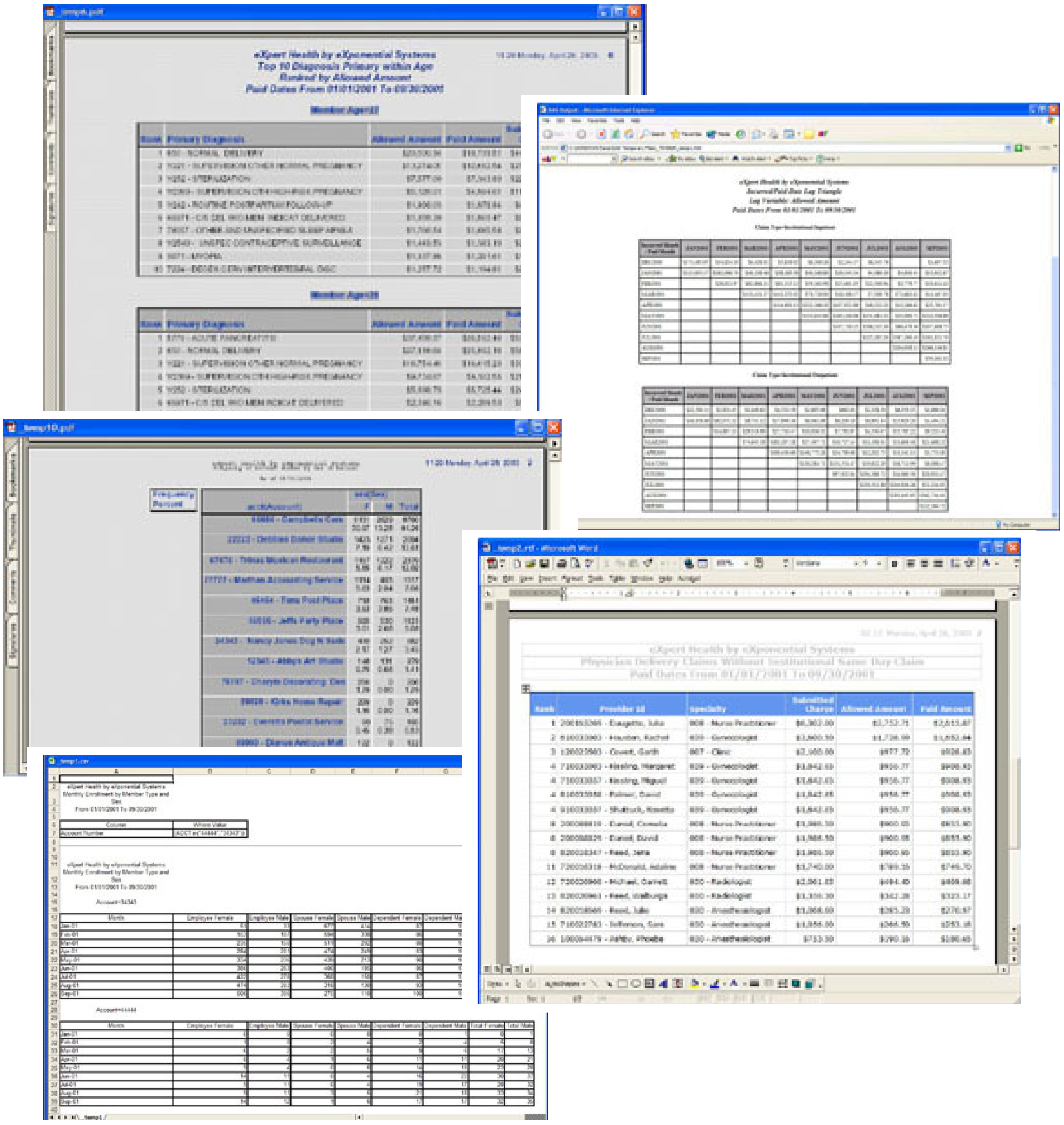
CONTACT INFORMATION

For more information, please contact:

Diane E Brown
7351 Shadeland Station, Suite 150
Indianapolis, IN 46256

(317) 823-0642
Diane.Brown@exponentialsystems.com
www.exponentialsystems.com

Figure 1: Sample Reports



eXpert Health by eXponential Systems
Top 10 Diagnosis Primary within Age
Ranked by Allowed Amount
Paid Dates From 01/01/2001 To 09/30/2001

eXpert Health by eXponential Systems
Incurred/Paid Date Lag Triangle
Lag Variable: Allowed Amount
Paid Dates From 01/01/2001 To 09/30/2001

eXpert Health by eXponential Systems
Monthly Enrollment by Member Type and Sex
From 01/01/2001 To 09/30/2001

eXpert Health by eXponential Systems
Physician Delivery Claims Without Institutional Same Day Claim
Paid Dates From 01/01/2001 To 09/30/2001

Rank	Provider Id	Specialty	Submitted Charge	Allowed Amount	Paid Amount
1	200153265 - Daugette, Julia	808 - Nurse Practitioner	$6,302.00	$2,752.71	$2,615.87
2	810033083 - Hausten, Rachel	839 - Gynecologist	$1,900.50	$1,738.00	$1,652.84
3	120023583 - Covert, Garth	807 - Clinic	$2,160.00	$977.72	$928.83
4	710033083 - Kiesling, Margaret	839 - Gynecologist	$1,842.65	$956.77	$908.93
4	710033057 - Kiesling, Miguel	839 - Gynecologist	$1,842.65	$956.77	$908.93
4	810033058 - Palmer, David	839 - Gynecologist	$1,842.65	$956.77	$908.93
4	910033057 - Shattuck, Rosetta	839 - Gynecologist	$1,842.65	$956.77	$908.93
8	200088810 - Daniel, Cornelia	808 - Nurse Practitioner	$1,965.50	$900.05	$851.90
8	200088820 - Daniel, David	808 - Nurse Practitioner	$1,968.50	$900.05	$851.90
8	820033347 - Reed, Jana	808 - Nurse Practitioner	$1,965.50	$900.00	$810.90
11	720036318 - McDonald, Adaline	808 - Nurse Practitioner	$1,740.00	$785.35	$746.70
12	720020960 - Michael, Darrell	830 - Radiologist	$2,061.83	$484.40	$459.08
13	820020963 - Reed, Walburga	830 - Radiologist	$1,356.30	$342.28	$323.17
14	820058068 - Reed, Julie	830 - Anesthesiologist	$1,068.00	$285.29	$270.97
15	710022783 - Jefferson, Sam	830 - Anesthesiologist	$1,056.00	$264.50	$251.18
16	100064179 - Ashby, Phoebe	820 - Anesthesiologist	$713.30	$200.16	$190.40

Figure 2: SAS Parameter Table

Obs	Rptname	Datain	whereclause
1	HEART_PROC	datain.ni	diag_primary in('40201' '40211' '40291' '40401' '40403' '40411' '40413' '40191' '40493' '4280')
2	SPEC_FREQ	datain.ni	
3	TOP40_HOSP	datain.ii	
4	ZERO_OBS	datain.ii	spec='034'
5	IN_EXCEEDED	datain.ii	
6	OUT_EXCEEDED	datain.ii	diag_primary = '4280'

Obs	Outfile	Outproc	Maxin	Maxout	Csv	Csvstyl	Htm	Htmstyle
1	c:\ema_sugi\output\heart_proc	PRINT	100000	2500	0	minimal	0	printer
2	c:\ema_sugi\output\spec_freq	FREQ	100000		0	minimal	0	printer
3	c:\ema_sugi\output\top40_hosp	PRINT	100000	2500	1	minimal	1	printer
4	c:\ema_sugi\output\zero_obs	PRINT	100000	2500	0	minimal	0	printer
5	c:\ema_sugi\output\in_exceeded	PRINT	1000	500	0	minimal	0	printer
6	c:\ema_sugi\output\out_exceeded	PRINT	100000	25	0	minimal	1	printer

Obs	Pdf	Pdfstyle	Rtf	rtfstyle	Tst	Mdb	sas
1	1	printer	0	sasweb	1	0	0
2	0	printer	1	sasweb	0	0	0
3	0	printer	0	sasweb	0	0	0
4	0	printer	0	sasweb	0	0	1
5	0	printer	0	sasweb	0	1	0
6	0	printer	0	sasweb	0	0	0