Paper PS08_05

MXG DB2 – A Cookbook Approach

Juliana Hughes
Federated Systems Group, Inc. Mainframe Capacity Planning

FSG (Federated Systems Group, Inc.) is a subsidiary of Federated Department Stores, Inc. (FDS, NYSE FD). FSG processes data for both Macy's and Bloomingdale's. On February 28, 2005 an announcement was made concerning the merger of Federated and May Company that will be accomplished in 3Qtr2005. As a result of the merger announcement, it is probable that the Federated mainframe complex will double in size when consolidation is complete. At the present time [May 2005], the Federated mainframe production complex consists of five logical partitions and a test complex containing two logical partitions. There are four IBM z900 model 2064 mainframe computers consisting of a total of 6733 MIPS. There are a two IBM S/390 model 9672's which are used as Coupling Facilities. Currently, the Performance Data Base (MXG PDB) resides on the z/OS sysplex and is built with SAS/MXG[1].

Federated DB2 usage has been increasing over the last several years, but we had not plowed under-the-hood because of other projects. I attended Barry Merrill's[1] MXG class just before the CMG 2003 conference. After returning from the conference, I proposed an MXG DB2 collection solution to Federated management and the proposal was accepted. Our committee for implementation included our DBA group, our performance analysis guru, our IGS partners and the FSG mainframe capacity planner (me). The implementation was not without 'trials-and-tribulations', but with perseverance, the deed has been done. We now are benefiting from a stable nightly data collection for understanding DB2 batch usage. Next, we will focus on collecting DB2 information from CICS data. When that is accomplished, the picture of what Federated DB2 is doing will be complete.

Because of the volume of SMF Type 100, 101, and 102 accounting records, the production jobs are submitted in parallel with each logical partition log dataset as input. The DB2 log switch occurs at midnight, but the jobs execute at 3:30AM. The average wall clock time for daily processing averages a sum of 6.5 hours. These jobs process only the DB2 data. We use MICS to process all other performance and capacity data.

For the project itself, I first began with a one LPAR input data set and used it as a pilot to see if the MXG code really worked. (OK – we had to start somewhere.) I was able to get the JCL and data together enough to produce some reasonable output. The DBA staff reviewed the output and we decided to gather more data from other LPARS. Using the theory of start small and grow big, we ran a full-complex day of Sunday data. We were ready to check our results. IBM Global Services (IGS) evaluated the usage of the system and determined how much CPU and DASD would be necessary to process the whole complex on a regular basis. The first-use code was turned over to IGS and the nightly collection process was begun. The dataset names were changed for standards usage and the jobs were input into the scheduler. Our IGS partners suggested some code in the middle of the many space abends that provided for keeping only the variables which we would need for reporting. Many jobs had to be rerun because of out-of-space problems. (Our IGS partner suggests that if, you too, have space problems because of the volume of data, consider putting the

data to tape). Note that the intermediate datasets are output to tape as quickly as possible and are needed in order to merge the data. By reducing the number of variables, we have been able to minimize DASD requirements and satisfy reporting requirements. Notice, that the SAS compression step is performed in the final SAS step. A previous CMG paper, written in 1995, points to the most efficient usage of SAS compression when using MXG [2]. In addition, our IGS code shark helped with MXG macro changes.

The DB2 processing requires creating both the **accounting** data and **accounting-package** data into a SAS PDB, and merging the two files. The **accounting** data has the DB2 TCB time, and the **accounting-package** data has the package id. Business units are identified as part of our DB2 package names. The output **merge** data contains both the TCB time and package id. The basic merge criteria used is the timestamp. In the paper, I refer to the creation of both the accounting data and the accounting-package data as the creation of intermediate datasets. I consider the creation of the merge data as the final step. The MXG documentation contains detailed descriptions of the variables.

The initial production jobs had abends due to volume (as noted before). Therefore, we limited the variables collected to the ones in the KEEP statement in Figure 1

For the **Account-P (Package)** data, refer to code in Figure 2.

The merge was originally coded by Chuck Hopf and is documented in the MXG sourclib.
The FSG modification added a few variables that we needed for reporting. See Figure 3.

The first report that was coded using the merge data was the 'DB2 Pig' report. It is a report of a day's worth of DB2 accounting merge data sorted by cpu time to identify the top high users. (By the way, I love the name 'pig' because it catches the attention of others. We affectionately refer to our various piglist reports which have been created.) The code for the report is in figure 4.

Note: Figure 4 report is not included in MXG and is specific to FSG use. In particular, we have naming standards for applications and packages which associate to different businesses and we use the accounting codes. Your mileage of similar SAS code may vary.

Our current plan is to output the DB2 data into a Microsoft Access database and use the trending functions for that product to show patterns. The current output is a daily comma-separated-value (CSV) file for input into Access, and two daily package reports.

I would like to encourage non-mainframers to give this methodology a 'look-see' because both SAS and MXG can be run on both midrange and PC computers. If you are concerned about the language, the code is the almost the same. SAS has a 'Learning-Edition' version which can be purchased for a nominal amount and is an inexpensive way to start; however, the learning-edition only allows for one thousand observations (or records). It contains many functions but limits the use by volume input.

Many thanks to Barry Merrill (Mr. MXG) and Chuck Hopf (Mr. DB2). Also, thanks to Rosalind Howe, IBM Global Services (IGS) and MXG code shark, for implementing the daily collection and to Stuart Pennington, IGS, for his work with the coordination of the implementation. In addition, thanks to Patti Perry, my manager, for editing and obtaining corporate legal publishing permissions. Much appreciation to

Rick Ralston, CMG Director, for his encouragement to write a paper and for his contribution called “My PDB” for the CMG MeasureIt publication.

If you have any questions, my phone number is 678-474-2708 and my email address at work is Juliana.hughes@fds.com.

The following code creates the accounting data.

```
---------- ACCT ----------
%LET PDB2ACC=PDB;
%LET ZDATE = TODAY() -1;
%LET MACKEEP=
  %QUOTE(
     MACRO _VDB2ACC /* PDB.DB2ACCT */
            KEEP=ACCOUNT1-ACCOUNT3
            ZDATE
      JOB DB2TCBTM QMDACNAM
      NETSNAME UOWTIME SYSTEM SMFTIME
      QWHSSSID QWHCOPID
      QPACALCT QPACALOG QPACANAR QPACARNA QPACARNC
QPACARND
      QPACARNE QPACARNG QPACARNH QPACARNJ QPACARNK
QPACARNL
      QPACARNM QPACARNN QPACARNO QPACARNQ QPACARNR
QPACARNS
      QPACARNW QPACAWAR QPACAWCL QPACAWDR QPACAWTE
QPACAWTG
      QPACAWTI QPACAWTJ QPACAWTK QPACAWTL QPACAWTM
QPACCANM
      QPACAWTO QPACAWTP QPACAWTQ QPACAWTR QPACAWTW
QPACUDNU
      QPACUDST QPACRECN QWHCCN   QWHCCV   QWHCAID  QWHCOPID
      QWHCPLAN NETSNAME UOWTIME  QWHSSTCK QPACPKID
    MACRO _BDB2ACC
     SYSTEM   QWHSSSID QWHCPLAN QWHCAID  QWHSLOCN QWHCCV
     QLACLOCN QWHCCN   QWACBSC  QWHSSTCK
    %
    MACRO _SDB2ACC
      PROC SORT NODUP %%VMXGFOR DATA= _WDB2ACC OUT= _LDB2ACC;
      BY _BDB2ACC ;
    %

   _NDB2
   MACRO _WDB2ACC DB2ACCT %
   MACRO _SDB2 _SDB2ACC %

   )  /* END OF QUOTE() */
   ;
 %INCLUDE SOURCLIB(TYPEDB2);RUN;
------------ END ACCT --------------------
```

Figure 1

The following code creates the accounting package data.

```
---------- ACCTP ----------
 %LET PDB2ACCP=PDB;
 %LET ZDATE = TODAY()-1;
 %LET MACKEEP=
  %QUOTE(
     MACRO _VDB2ACP /* PDB.DB2ACCP */
           KEEP=ACCOUNT1-ACCOUNT3
           ZDATE
      QPACALCT QPACALOG QPACANAR QPACARNA QPACARNC
QPACARND
      QPACARNE QPACARNG QPACARNH QPACARNJ QPACARNK
QPACARNL
      QPACARNM QPACARNN QPACARNO QPACARNQ QPACARNR
QPACARNS
      QPACARNW QPACAWAR QPACAWCL QPACAWDR QPACAWTE
QPACAWTG
      QPACAWTI QPACAWTJ QPACAWTK QPACAWTL QPACAWTM
QPACCANM
      QPACAWTO QPACAWTP QPACAWTQ QPACAWTR QPACAWTW
QPACUDNU
      QPACUDST QPACRECN QWHCCN   QWHCCV   QWHCAID
QWHCOPID
      QWHCPLAN NETSNAME UOWTIME  QWHSSTCK QPACPKID
      UOWTIME QPACREC QWHCATYP
      SYSTEM SMFTIME

        MACRO _BDB2ACP
     QWHSSSID QPACLOCN QPACCOLN QPACPKID QWHSSTCK
   %
   MACRO _SDB2ACP
     PROC SORT NODUP  %%VMXGFOR DATA= _WDB2ACP OUT= _LDB2ACP;
     BY _BDB2ACP ;
   %
   _NDB2
  MACRO _WDB2ACP DB2ACCTP %
  MACRO _SDB2 _SDB2ACP %

  )  /* END OF QUOTE() */
  ;
 %INCLUDE SOURCLIB(TYPEDB2);RUN;

---------- END ACCTP ----------
```

Figure 2

The following code creates the merged data.

```
---------- MERGE ----------
 PROC SORT DATA=DB2ACCT.DB2ACCT OUT=WORK1.SORTACCT;
    BY QWHCCN QWHCCV QWHCAID QWHCOPID QWHCPLAN NETSNAME
UOWTIME
       DESCENDING QWHSSTCK;
    RUN;
PROC SORT DATA=DB2ACCTP.DB2ACCTP OUT=WORK2.SORTPKGS;
   BY QWHCCN QWHCCV QWHCAID QWHCOPID QWHCPLAN NETSNAME
UOWTIME
      DESCENDING QWHSSTCK;
   RUN;
 DATA ACCT PKGS/VIEW=PKGS;
    SET WORK1.SORTACCT(IN=INACCT) WORK2.SORTPKGS(IN=INPKGS);
    BY QWHCCN QWHCCV QWHCAID QWHCOPID QWHCPLAN NETSNAME
UOWTIME
      DESCENDING QWHSSTCK;
 IF INACCT THEN GROUP+1;
 IF GROUP=. THEN GROUP=1;            /* INIT IF FIRST NOT ACCT */
 IF INACCT THEN OUTPUT ACCT;
 IF INPKGS THEN OUTPUT PKGS;
 RUN;
PROC MEANS DATA=PKGS NOPRINT;
   BY QWHCCN QWHCCV QWHCAID QWHCOPID QWHCPLAN NETSNAME
UOWTIME
     GROUP;
   ID ACCOUNT1 ACCOUNT2 ACCOUNT3;
   OUTPUT OUT=SUMSPKGS(DROP=X _TYPE_ _FREQ_)
   SUM=X QPACALCT QPACALOG QPACANAR QPACARNA QPACARNC
QPACARND
       QPACARNE QPACARNG QPACARNH QPACARNJ QPACARNK
QPACARNL
       QPACARNM QPACARNN QPACARNO QPACARNQ QPACARNR
QPACARNS
       QPACARNW QPACAWAR QPACAWCL QPACAWDR QPACAWTE
QPACAWTG
       QPACAWTI QPACAWTJ QPACAWTK QPACAWTL QPACAWTM
QPACAWTN
       QPACAWTO QPACAWTP QPACAWTQ QPACAWTR QPACAWTW
QPACCANM
       QPACCAST QPACSCT  QPACSPNS QPACSQLC QPACTJST QPACUDNU
       QPACUDST
    MAX=QPACRECN;
    VARIABLES
       QPACRECN
```

```
        QPACALCT QPACALOG QPACANAR QPACARNA QPACARNC
QPACARND
        QPACARNE QPACARNG QPACARNH QPACARNJ QPACARNK
QPACARNL
        QPACARNM QPACARNN QPACARNO QPACARNQ QPACARNR
QPACARNS
        QPACARNW QPACAWAR QPACAWCL QPACAWDR QPACAWTE
QPACAWTG
        QPACAWTI QPACAWTJ QPACAWTK QPACAWTL QPACAWTM
QPACCANM
        QPACAWTO QPACAWTP QPACAWTQ QPACAWTR QPACAWTW
QPACUDNU
        QPACUDST;
PROC DELETE DATA=WORK1.SORTACCT;
RUN;
 PROC DELETE DATA=WORK2.SORTPKGS;
 RUN;

 DATA DB2ACCTK INPKGONL;
   MERGE ACCT(IN=INACCT) SUMSPKGS(IN=INPKGS);
   BY QWHCCN QWHCCV QWHCAID QWHCOPID QWHCPLAN NETSNAME
UOWTIME
     GROUP;
 IF INACCT THEN OUTPUT DB2ACCTK;
 ELSE IF INPKGS THEN OUTPUT INPKGONL;
 LABEL QPACRECN="MAXIMUM*PACKAGES*EXECUTED";
 RUN;
OPTIONS COMPRESS=YES;      /* compression is optional – another FSG
modification */
 PROC COPY IN=WORK OUT=PDB
    SELECT DB2ACCTK;
    RUN;
---------- END MERGE ----------
```

Figure 3

The following code creates the reporting data and reports.

```
---------- REPORT -------------------------
//*         ---EXTRACT---
//STEPBR10 EXEC PGM=IEFBR14
//SASLIST  DD DSN=xxxxxx.SASLIST.DB2PIGS.EXTRACT,
//            LRECL=133,RECFM=FB,
//
DISP=(MOD,DELETE,DELETE),SPACE=(CYL,(9,9),RLSE)
//SASLOG   DD DSN=xxxxxx.SASLOG.DB2PIGS.EXTRACT,
//            LRECL=133,RECFM=FB,
//
```

```
DISP=(MOD,DELETE,DELETE),SPACE=(CYL,(9,9),RLSE)
//OUT      DD
DSN=xxxxxx.DB2PIGS.SAS,DISP=(MOD,DELETE,DELETE),
//            RECFM=FS,LRECL=27648,SPACE=(CYL,(1100,1100))
//*------use micsproc because of format statement ------
//STEP0020 EXEC MIC6SHRA,COND=(4,LT),OPTIONS='ERRORABEND'
//WORK     DD
DCB=BLKSIZE=27648,SPACE=(CYL,(1000,1000),RLSE)
//SOURCLIB DD DISP=SHR,DSN=xxxxxx.MXG.V2202.SOURCLIB
//LIBRARY  DD DISP=SHR,DSN=xxxxxx.MXG.V2202.FORMATS
//PDBC101  DD
DISP=SHR,DSN=xxxxxx.MXG.DB2C101.MERG.PDBLIB(0)
//PDBC102  DD
DISP=SHR,DSN=xxxxxx.MXG.DB2C102.MERG.PDBLIB(0)
//PDBC104  DD
DISP=SHR,DSN=xxxxxx.MXG.DB2C104.MERG.PDBLIB(0)
//PDBC106  DD
DISP=SHR,DSN=xxxxxx.MXG.DB2C106.MERG.PDBLIB(0)
//PDBC107  DD
DISP=SHR,DSN=xxxxxx.MXG.DB2C107.MERG.PDBLIB(0)
//SASLIST  DD DSN=xxxxxx.SASLIST.DB2PIGS.EXTRACT,
//            LRECL=133,RECFM=FB,
//       DISP=(MOD,CATLG,CATLG),SPACE=(133,(100,100),RLSE)
//SASLOG   DD DSN=xxxxxx.SASLOG.DB2PIGS.EXTRACT,
//            LRECL=133,RECFM=FB,
//       DISP=(MOD,CATLG,CATLG),SPACE=(133,(100,100),RLSE)
//OUT      DD
DSN=xxxxxx.DB2PIGS.SAS,DISP=(NEW,CATLG,DELETE),
//        RECFM=FS,LRECL=27648,SPACE=(CYL,(1100,1100),RLSE)
//SYSIN    DD *
options source source2 nocenter ps=32767;
%macro db2pigs(lpar);   /* macro start -------------- */
data data&lpar(drop=system qwhsstck rate db2tcbtm);
set pdb&lpar..db2acctk(keep=system qwhsstck db2tcbtm
qpacpkid  job qwhsssid account1-account3 qwhccv);
    attrib xtran  format=$4.    length=$4.
label="Corr*Tran";
    attrib fob    format=$16.   length=$14. label="FOB";
    attrib xdate  format=date9.             label="Date";
    attrib xhour  format=2.                 label="Hr";
 proc printto print=saslog;
 proc datasets library=pdbc101;
 proc contents data=pdbc101._all_;
 proc printto;
%mend skip;   /* macro end ------------------------ */

%db2pigs(C101);
```

```
run;
%db2pigs(C102);
run;
%db2pigs(C104);
run;
%db2pigs(C106);
run;
%db2pigs(C107);
run;
    /* put datasets together */
data data;
    set datac101 datac102 datac104 datac106 datac107;
run;
proc copy in=work out=out;
run;
//*        ---REPORT---
//STEPBR30 EXEC PGM=IEFBR14
//SASLIST  DD DSN=xxxxxx.NM.SASLIST.DB2PIGS.REPORT,
//            LRECL=133,RECFM=FB,
//            DISP=(MOD,DELETE,DELETE),SPACE=(CYL,(9,9))
//SASLOG   DD DSN=xxxxxx.NM.SASLOG.DB2PIGS.REPORT,
//            LRECL=133,RECFM=FB,
//            LRECL=133,RECFM=FB,
//            DISP=(MOD,DELETE,DELETE),SPACE=(CYL,(9,9))
//CSV      DD DSN=xxxxxx.NM.DB2PIGS.OUTPUT,
//            DISP=(MOD,DELETE,DELETE),
//            LRECL=80,RECFM=FB,SPACE=(CYL,(25,25),RLSE)
//*  ------------------------------------------------
//STEP0020 EXEC SAS,COND=(4,LT),OPTIONS='ERRORABEND'
//IN       DD DISP=SHR,DSN=xxxxxx.NM.DB2PIGS.SAS
//LIBRARY  DD DISP=SHR,DSN=xxxxxx.NM.MXG.V2202.FORMATS
//SASLIST  DD DSN=xxxxxx.NM.SASLIST.DB2PIGS.REPORT,
//            LRECL=133,RECFM=FB,
//            DISP=(NEW,CATLG,CATLG),SPACE=(133,(100,100))
//SASLOG   DD DSN=B0TCAP.NM.SASLOG.DB2PIGS.REPORT,
//            LRECL=133,RECFM=FB,
//            DISP=(NEW,CATLG,CATLG),SPACE=(133,(100,100))
//CSV      DD DSN=B0TCAP.NM.DB2PIGS.OUTPUT(DB2PIGS),
DISP=OLD
//SYSIN    DD *
options source source2 nocenter ps=32767;
proc summary data=in.data missing noprint nway;
    class xdate fob qpacpkid xtran job;
    id account1-account3;
    var ncputm;
    output out=data(drop=_type_) sum=;
    run;
```

```
    /* print to saslog */
%macro skip;               /* define macro ----------- */
    proc printto print=saslog;
    proc contents data=_last_;
    title "Output from summary(obs=10)";
    proc print data=_last_(obs=10) width=minimum;
       attrib xdate format=date9.;
    run;
    proc printto;
    %mend skip;        /* end macro -------------- */
proc sort data=data;
    by xdate fob descending ncputm;
    run;
    /* report */
proc printto print=saslist;
title1 "FSG   <DB2PIGS>";
proc report data=_last_(where=(ncputm>="0:10:00"t))
                       headline split='*' missing
out=data;
cols xdate fob qpacpkid xtran job ncputm _freq_ avg
account1-account3;
define xdate    / group order format=date9. "Date";
define fob      / group order format=$16.   "Fob";
define qpacpkid /             format=$18.   "Package";
define xtran    /             format=$4.    "Corr*Tran";
define job      /             format=$8.    "Job";
define ncputm   / order descending format=time. "Ncputm";
define _freq_   /                             "Freq";
define avg      / computed    format=time.  "Avg";
define account1 /             format=$3.    "Ac1";
define account2 /             format=$3.    "Ac2";
define account3 /             format=$3.    "Ac3";
compute avg;
    avg=_c6_/_c7_;
endcomp;
    run;
------------------ END REPORT --------------------
```

Figure 4

References:

(1)Merrill, Barry "Merrill's Expanded Guide to Computer Performance Evaluation" www.mxg.com

(2)Franklin, John, CMG Proceedings, 95AU5024 "The Performing Analyst, A Tutorial for Users of SAS and MXG"