CHAPTER 4

Using MDDBs

Using Version 6, Version 7, and Version 8 MDDBs

If you are using MDDBs created with different versions of SAS/MDDB Server, you should be aware of the following:

- Version 8 of SAS/MDDB Server reads Version 6 MDDBs without conversion but not Version 7 MDDBs. If you have MDDBs created in Version 7, you should re-create them with Version 8.
- MDDBs created in a given version will not be read by a previous version of SAS/MDDB Server. For example, SAS/MDDB Server Version 6 will not read MDDBs created in Version 7 or Version 8.

Using an MDDB with SAS/EIS Software

To exploit your MDDBs, you can use the MDDB-enabled objects in SAS/EIS software for a large variety of multidimensional reports. SAS/EIS software includes a wide range of business reporting objects that have been specifically optimized for use with MDDBs. These reporting objects, which provide a choice of styles for dynamic display and interactive analysis, include the following:

- 3D Business Charts
- Bubble Charts
- Graphical Variance Reports
- Maps
- Multidimensional Business Trends
- Multidimensional Charts
- Multidimensional Pie Charts

- □ Multidimensional Reports
- □ Organizational Charts.

The multidimensional report objects have multiple levels with built-in drill-down capabilities for fast navigation through data. Using these objects and customizable viewers, users can

- □ navigate through the data
- □ perform analyses by adding calculations at run time
- □ dynamically reach through to detail data anywhere on the network.

For details on using an MDDB with SAS/EIS objects, refer to the SAS/EIS online Help.

Using an MDDB with SAS/AF Software

You can access MDDBs from applications that you create with SAS/AF software by using the MDDB_M class. The following example shows how to use the MDDB_M class to provide data to the SAS/AF Table Editor viewer. For more information on the MDDB_M class, refer to the SAS/MDDB Server online Help.

This SCL code attaches the table editor to an instance of the MDDB_M class and reads in the MDDB that was created in “Example 1: Building an MDDB Using the MDDB Procedure” on page 21.

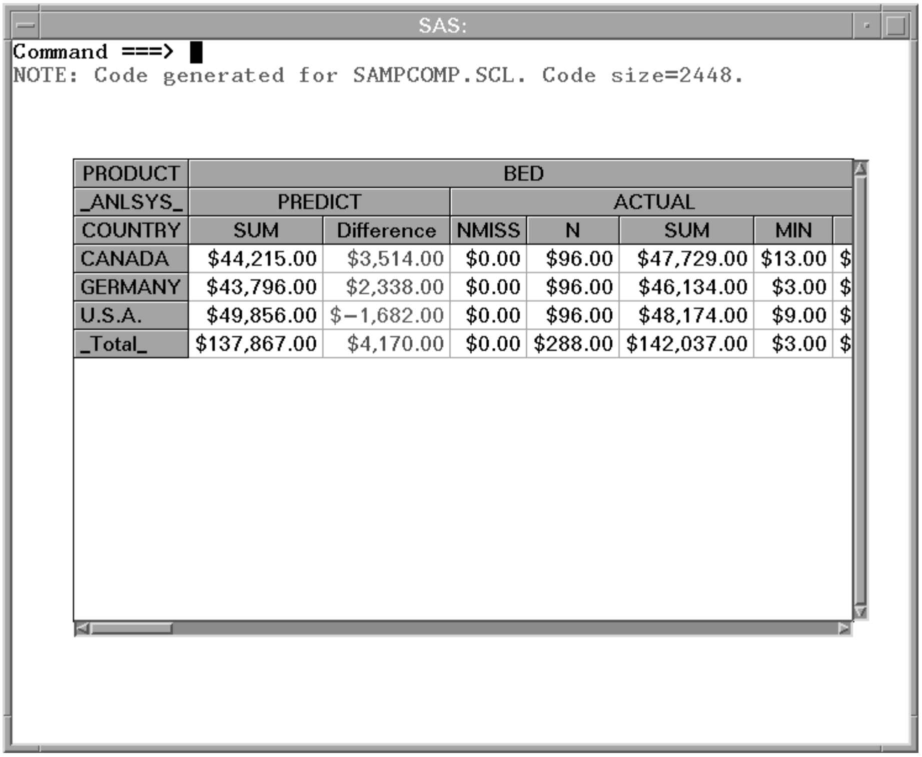
```
/*-- load the MDDB_M class to read the mddb data --*/
dcl object tabid;
dcl object mddbid=_new_ sashelp.mddb.mddb_m();

init:
   /*-- get the table editor id --*/
   _frame_._getWidget('obj1', tabid);

   /*-- attach the table editor to this instance of mddb_m --*/
   tabid._attach(mddbid);

   /*-- read in the MDDB you created --*/
   mddbid._setMddbTable('SASUSER.MDDB');


   classlist=makelist();
   /*-- create the class list for your table --*/
   rc = insertc(classlist, 'COUNTRY', -1);
   rc = insertc(classlist, 'PRODUCT', -1);

   /*-- make your table --*/
   mddbid._makeTable(classlist);
   mddbid._addComputedValue('_STATS_', 'Difference');

   /*-- set the SCL that will be run to calculate Difference --*/
   mddbid._setSource('SAMPCOMP.SCL', 'Y');
   rc=rc;
return;
```

The following SCL code is the SAMPCOMP.SCL program called from the previous code. It calculates the computed values for this example:

```
_stats_='';;
_anlsys_='';;

init:
   cellist=makelist();
   rc=rc;
   /*-- find out the address of the cell we are computing --*/
   _self_._getCompCellAddress(cellist);

   /*-- if we are computing the difference of ACTUAL/PREDICT sales --*/
   if ( _stats_="Difference" ) then
   do;
      getlist=makelist();
      rc=copylist(cellist, 'y', getlist);

      /*-- setup the address to request SUM values --*/
      rc = setnitemc(getlist, 'SUM', '_STATS_');

      if (_anlsys_='ACTUAL') then
      do;
         _self_._getValues(getlist, aval);
         rc=setnitemc(getlist, 'PREDICT', '_ANLSYS_');
         _self_._getValues(getlist, pval);
      end;
      else
      do;
         _self_._getValues(getlist, pval);
         rc=setnitemc(getlist, 'ACTUAL', '_ANLSYS_');
         _self_._getValues(getlist, aval);
      end;

      /*-- set up the value of the computed cell to contain --*/
      /*-- the difference of predicted sales from actual sales --*/
      value=aval - pval;

      /*-- if we predicted more than we actually sold,   --*/
      /*-- turn that cell to RED so the value stands out --*/
      if (value < 0) then
      /*-- if the difference is negative it will be RED --*/
      _self_._setCellColor(cellist, 'red');
      else
      /*-- if the difference is positive it will be GREEN --*/
      _self_._setCellColor(cellist, 'green');
      rc=dellist(getlist,'Y');
   end;
   rc=dellist(cellist,'Y');
return;
```

The following display shows the results of running a FRAME entry with the SAS/AF Table Editor Viewer object and gives the results of the above programs:

Display 4.1 Table Editor Viewer Frame



```
Command ===>
NOTE: Code generated for SAMPCOMP.SCL. Code size=2448.
```

PRODUCT	BED						
ANLSYS	PREDICT		ACTUAL				
COUNTRY	SUM	Difference	NMISS	N	SUM	MIN	
CANADA	$44,215.00	$3,514.00	$0.00	$96.00	$47,729.00	$13.00	$
GERMANY	$43,796.00	$2,338.00	$0.00	$96.00	$46,134.00	$3.00	$
U.S.A.	$49,856.00	$−1,682.00	$0.00	$96.00	$48,174.00	$9.00	$
Total	$137,867.00	$4,170.00	$0.00	$288.00	$142,037.00	$3.00	$

Using an MDDB with the MDDB Report Viewer

If you want to access MDDBs without creating and deploying your own application, you can use the MDDB Report Viewer application, which is available as a component of SAS/IntrNet Application Dispatcher software.

The MDDB Report Viewer enables users to generate and view reports and graphs of data that are stored in MDDBs without running a SAS session. This capability eliminates the need to have SAS software running on all users' machines and provides access to the MDDB reports and graphs in a Web environment.

The MDDB Report Viewer contains three pages in which the user enters information to view an MDDB report. In general, the user selects an MDDB to view; specifies the down, across, and analysis variables and statistics to be included in the report; and sets some report options. Users can also specify whether to display a graph with the report. After the report appears, users can drill down to additional values in a report; rotate the down and across dimensions of the report; modify the report dimensions, analysis variables, and statistics; generate a two-dimensional or three-dimensional graph of the report data; download the report to a spreadsheet; and print the report.

For complete instructions on how to use the MDDB Report Viewer to view MDDBs, refer to the online SAS/IntrNet Application Dispatcher documentation.

Transporting MDDBs Across Operating Environments

In some business computing environments, it is necessary for MDDBs that are created in one operating environment to be used on a different, incompatible operating environment. In this case, the possible solutions are to

- □ copy the MDDB to the other operating environment
- □ access the MDDB from the other operating environment over a network.

The latter option is addressed in “Accessing MDDBs Remotely” on page 32. This section discusses copying (“transporting”) MDDBs from one operating environment to another.

The CPORT and CIMPORT procedures, which are used to transport SAS files between operating environments, do not support the transport of MDDBs. Instead, the following methods can be used:

- □ the COPY procedure
- □ the UPLOAD and DOWNLOAD procedures
- □ the Cross Environment Data Access (CEDA) facility.

The first method requires either SAS/SHARE software or SAS/CONNECT software. The last two methods require SAS/CONNECT software.

Transporting MDDBs with the COPY Procedure

To use the COPY procedure to transport MDDBs, you must have SAS/SHARE or SAS/CONNECT software licensed.

1 Establish a connection between the local host and the SAS/SHARE server or remote host. See *SAS/SHARE User's Guide* or *SAS/CONNECT User's Guide* for details.

2 Use the LIBNAME statement to declare a library on the SHARE server or remote host:

- □ For SAS/SHARE sessions

```
libname rmtlib server=rmthost.share1;
```

This example informs the local SAS session that the libref RMTLIB references the RMTLIB library on the SAS/SHARE server share1, which is located on a host called RMTHOST.

- □ For SAS/CONNECT sessions

```
signon rmtpc;
libname rmtlib remote 'd:\sasmddb' server=rmtpc;
```

This example establishes a connection with the remote host RMTPC and informs the local SAS session that the libref RMTLIB references the SAS data library d:\sasmddb on the RMTPC host.

3 Use the COPY procedure to copy an MDDB between the local and remote hosts:

```
proc copy out=work in=rmtlib memtype=mddb;
```

This example copies all MDDBs from the RMTLIB library on the remote host to the temporary library on the local host. The COPY procedure can also be used to copy MDDBs from the local host to the remote host.

Transporting MDDBs with the UPLOAD and DOWNLOAD Procedures

To use the UPLOAD and DOWNLOAD procedures to transport MDDBs, you must have SAS/CONNECT software licensed.

1 Establish a connection to the remote host. (See the *SAS/CONNECT User's Guide* for details.)

2 Use the UPLOAD procedure to transport the MDDB to the remote host or the DOWNLOAD procedure to transport the MDDB from the remote host:

```
proc upload data=school.grades out=grades_copy;
```

This example copies the MDDB GRADES from the local SCHOOL library to the WORK library on the remote host as GRADES_COPY.

For complete details on the UPLOAD and DOWNLOAD procedures, see the *SAS/CONNECT User's Guide.*

Accessing MDDBs Remotely

Remote access of MDDBs is preferable to copying them if the MDDBs are frequently updated or your data center rules prohibit multiple copies of data. Depending on which method you use, you can specify read, write, and update access to your MDDBs from remote clients on a single-client or multiple-client basis.

Accessing MDDBs Via Remote Library Services

To access MDDBs via Remote Library Services (RLS), you establish a SAS/SHARE or SAS/CONNECT session and declare libraries on the remote hosts as shown in “Transporting MDDBs with the COPY Procedure” on page 31. MDDBs that are stored on the remote hosts are then available for reading.

Accessing MDDBs Via CEDA

To access MDDBs with the Cross Environment Data Access (CEDA) facility, you must have SAS/CONNECT software licensed on the host that will be accessing the file.

CEDA is a SAS Version 8 facility that provides read-only access of supported member types to all Version 8 operating environments except CMS. For complete details on CEDA, see *SAS/CONNECT User's Guide* and *Moving and Accessing SAS Files across Operating Environments.*

To make an MDDB available for reading via CEDA, use Network File Services (NFS) to mount the file on a network for host access. Ensure that the file has the appropriate filename extension (for example, **.sas8bmdb** for Version 8). See the documentation for NFS and your operating environment for details. When the file is mounted, any Version 8 host (except CMS) with access to the NFS can then read the MDDB.

The correct bibliographic citation for this manual is as follows: SAS Institute Inc., *SAS/MDDB® Server Administrator's Guide, Version* 8 , Cary, NC: SAS Institute Inc., 1999.

SAS/MDDB® Server Administrator's Guide, Version 8

ISBN 1–58025–504–3

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513.

1st printing, October 1999

The Institute is a private company devoted to the support and further development of its software and related services.