

Getting Started With the SAS/MDDB

Ben Cochran, The Bedford Group, Raleigh, NC

Abstract

With Release 6.12, the SAS System contained a new feature which is an extension of its data warehousing strategy, the SAS multidimensional database (MDDB). This paper describes an MDDB, tells why you would want to use one, and shows how to build one. Finally this paper will illustrate the use of an MDDB through the SAS/EIS Multidimensional Report Object.

Introduction

To accomplish its objectives, this paper is divided into two parts. The first section focuses on describing and building the MDDB as well as data registration issues. The second section looks at putting the MDDB to work as well as performance issues.

What is an MDDB? Basically the MDDB is a type of SAS file (it has a filetype of MDDB). It is a read-only file that provides the SAS System with a way to store large amounts of pre-summarized data (similar to, but different from PROC SUMMARY output).

It contains at least one summary structure known as an NWAY table. Additionally, it usually contains one sub-table created from the NWAY table. While there are no limits to the number of sub-tables, too many unnecessary ones can result in storage space problems. It can be thought of as a 'cube' with multiple dimensions that provide fast and flexible access to the data.

Just like in the SUMMARY procedure, analysis and classification variables are specified. There is no limit to the number of either type of variable. Also, like the summary procedure, one can specify the statistics you wish to have. The MDDB can store up eight statistics and create an additional thirteen at run-time.

The SAS/MDDB has incremental update capability and can be created either in batch programs or on-line applications.

Before Building an MDDB, you should have a good understanding of your data and how it will be used. One of the biggest issues is finding a good balance between response time and storage space. You need to know which sub-tables to build so that you do not use disk space needlessly, and at the same time, have the best performance. In order to achieve the best response time, a good rule to remember is to have a sub-table that contains exactly the same categorical variables as the drill-down report.

Having a good understanding of user expectations will also enable you to address the time vs. space issue. What reports do the end-users want, what drill-down hierarchy is needed, etc.

Building an MDDB can be accomplished in more than one way. This paper will focus on the PROC MDDB approach. The general syntax is:

```
proc mddb data = dsname
  out = libref.mddb ;
  class categorical variable list ;
  hierarchy class variable list /
    name= xxx display = yes | no ;
  var analysis variables list / statistics ;
run ;
```

On the `proc` statement, the `data =` option names the SAS data set to be summarized. The `out =` option names the MDDB that will be created. Ex.

```
proc mddb data = sas.yr98 out = new.mddb ;
```

The `class` statement names the variables to be used as classification variables in the same way that a `class` statement is used in the summary procedure. Ex.

```
class country city company ;
```

The default order of the variables' values on the class statement is ascending. This can be changed by using any of the following order options: descending, ascformatted, desformatted, disorder. For example, a data set has countries in order by continent and that order must be retained in a report. To accomplish this, the above class statement needs to be modified as follows:

```
class country / disorder ;
class city company ;
```

The **hierarchy** statement creates sub-tables in the MDDB. The sub-table can be named with the **name =** option (this is highly recommended). By default, these named sub-tables are not recognized by SAS/EIS during the during the metabase registration process. To change this default action, specify *yes* as the value for the **display =** option. This will eliminate the need to create hierarchies while going through the SAS/EIS registration process. Ex.

```
hierarchy country city company /
name='geo' display = yes ;
```

Note: If the hierarchy statement is NOT used, then only the NWAY table is produced.

The **var** statement identifies the analysis variables and the statistics to be stored in the MDDB. Ex.

```
var sales96 sales97 / sum n ;
```

More than one var statement may be used, however a given variable may appear in only one var statement. While the sum is the default statistic, other statistics that can be used on the var statement are: n nmiss min max uss sumwgt and uwsum.

The SAS dataset that will be used in this paper to illustrate the building of an MDDB is **SASHELP.MDV**. Since this dataset ships with the SAS System, it will be fairly easy for the reader to use this paper as a guide to building an MDDB.

The complete proc mddb step for building example an MDDB is:

```
proc mddb in = sashelp.mdv
out= sasuser.mymddb ;
class country city company ;
var sales93 sales94 sales95 _4cast96 /
n sum ;
hierarchy country city company /
name = 'geo' display = yes ;
run ;
```

The above code builds the MDDB, but before it can be used in a SAS/EIS application, the MDDB needs to be registered.

Registering the MDDB: To register the MDDB so that it can be used in drill-down applications, the user needs to enter the SAS/EIS environment. Do this by typing **eis** on the command line, or go to the PMENU facility and select **Globals**, then **Develop**, and then **EIS application builder**. This makes the main menu for SAS/EIS appear. Select the Metabase icon to open the *Metabase* window.

For the purposes of this paper, we will use the default metabase of **SASUSER.MBUSER**. Make sure the path points to the **SASUSER** library and the metabase is set to **MBUSER**. If this is the first time this metabase has been used, it will be empty. Select the **Add** pushbutton under the **Tables** listbox. This opens the *Select Table* window. Choose the path for the **SASUSER** library, then scroll to find the **MYMDB mddb** and select it. Choose **OK** to return to the *Metabase* window. The **SASUSER.MYMDB** file should be in the **Tables** listbox. Notice the default attributes that appear in the **Attributes** listbox. Notice that the MDDB is designated as an **EXTERNAL MDDB**. All this means is that it was created outside of SAS/EIS with **PROC MBBDB**. Double click on the **LABEL** attribute. Change the label to **MDDB for User Group DEMO**. Next, select **OK** to close the *Enter Label* window. Now, double click on the **HIERARCH** attribute to open the **Table Hierarchies** window. Verify that the **Geo Hierarchy** exists as defined in the **PROC Mddb** step. Select **OK**, then **Close** to return to the SAS/EIS main menu. The MDDB is now registered to be used in an eis object.

Building the Object: The next step is to

build the Multidimensional Report Object that will use the MDDB. From the main menu in SAS/EIS, select the Build EIS icon to make the *Build* window appear. Use the default application database, SASUSER.SASAPPL to build this object.

To begin, choose the *Add* pushbutton at the bottom of the *Build EIS* window. Select Business Reports from the Object Database listbox. Then select Multidimensional Reports from the *Objects* listbox. Next, select the *Build* pushbutton to open the build environment. Supply the name and description of your choice. Click on the arrow beside Table: and select the the SASUSER.MYMDDB from the *Available* listbox. Select OK. Click on the arrow beside Columns: . For the Down dimension, select Geo (built from the proc mddb step). Next, select Analysis and choose all the variables listed. We are finished setting up the columns and rows of the report, so click on OK to close the *Column Selection* window. Next, select the Statistics arrow, and choose the SUM as the statistic. The report is ready to be viewed, so click on the *Test* pushbutton.

Viewing the report, run-time customization, and MDDB performance issues will be demoed at the presentation of this paper.

Conclusion

The SAS System has greatly improved its ability to access data quickly with the offerings of Release 6.12 which includes the multidimension-al database. It is ideal for drill-down reports on datasets commonly found in data warehouses. Therefore, it greatly enhances the capability of SAS/EIS to be deployed as part of the front-end to data marts as well as data warehouses.

The author can be reached at the following location:

Ben Cochran
The Bedford Group
3216 Bedford Avenue
Raleigh, NC 27607