Paper 50-28

GoodsHound - Building Multi-Functional Web-Based applications with SAS/IntrNet and JavaScript

Blake R. Sanders and Mikhail Gruzdev, U.S. Census Bureau, Washington, DC

ABSTRACT

The advantages of SAS/MDDBs can be built into applications without needing the large SAS/MDDB files. The trick is to know what the user needs.

INTRODUCTION

SAS/MDDB files provide an incredible amount of flexibility for data users. The price of that flexibility, however, can be steep if the original data sets are large enough. With the right planning and research, it is possible to lessen that burden by using SAS data sets and including enough functionality in the application itself.

HISTORY

The U.S. Census Bureau's Foreign Trade Division (FTD) has used SAS/MDDBs for data dissemination since their inception. Their speed and flexibility have proven a boon to our staff. Previously, they used dBase programs to create multiple text files that were assembled into spreadsheets. The spreadsheets were formatted and the output was sent out. Now, with our EIS interface, a request that once took hours can be completed in minutes using SAS/MDDBs.

DRAWBACKS

Our only complaint about SAS/MDDBs was on the production end. That problem, of course, is of our own doing.

SAS/MDDBs take a SAS data set and include all of the possible permutations of the data. This way, when a person asks a specific question, the answers are already computed. However, if a large data set is processed to create the SAS/MDDB, it takes a large amount of resources to produce those numbers: sorts, totals, sub-totals, etc. They have to come from somewhere. By design, production is when that happens.

Many of our data sets, unfortunately, are so large that the computers nearly choked while creating the SAS/MDDB files. Either that or they took a LONG TIME to complete.

OUR WORK-AROUND

As we wrote in our paper THE SAS/MDDB SHELL GAME: OPTIONS THEY DON'T SHOW IN SAS/HELP (SUGI 25, Paper 35-25), our solution was to create a SAS/AF application that subset SAS datasets and created a SAS/MDDB file directly on the user's machine. In doing this, we were able to store as much data as we needed in SAS data sets and did not have the production burden associated with our mammoth SAS/MDDB files.

This new application, called "The Flying MDDB" gave us access to more data than we'd been able to fit into any of our SAS/MDDB applications while still providing all of the advantages of SAS/MDDB applications. It also solved our production problems.

SOLUTIONS HAVE DRAWBACKS

"The Flying MDDB" gave our users access and flexibility that was unheard of. We even managed to win a few fans in the group database programmers who were tired of writing the same code week after week. For all "The Flying MDDB"'s theoretical advantages, our implementation had some practical disadvantages:

1.) The application's SAS/SCL code limited the number of items that could be included in a request.
2.) Our hardware and local area network (LAN) would occasionally deny people access.

HOW DO WE IMPROVE THE SITUATION?

The first "problem", limiting the scope of the request, was partly a function of the application being implemented in SAS v 6.12. So, while we could correct that by porting the program to v 8.2, our LAN problems drove us to consider alternatives.

Once people got to "The Flying MDDB", they had no problem accessing the data. Past the access problem, the main complaint involved the program's interface.

So, we decided to reconstruct the interface of the application and not the entire program and its underlying data structure.

GOODSHOUND

Since we'd had success in creating Internet applications using SAS/IntrNet, we decided to leverage that experience in reworking "The Flying MDDB" into an application that later became known as "GoodsHound".

Why did we choose a web-based route?

1.) With our experience, development time would be short.
2.) It could fit on top of the existing data structure for the "Flying MDDB".
3.) Access to the application was not a problem if you had permission… whereas our LAN made access to "The Flying MDDB" "spotty" regardless of your permissions.
4.) Web server processing is faster than the processing on many people's desktops.
5.) Take advantage of SAS v 8.2

REFINING THE INTERFACE

As with any data application, the user's access to data depends on how you present it. With our applications, we need to know what kind of trade data the user needed.

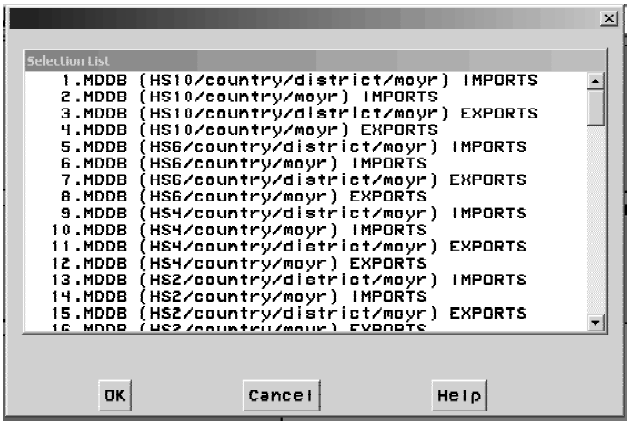


Figure 1 Report selection menu of "The Flying MDDB"

This selection list from "The Flying MDDB" that helps define a request (Figure 1) tells the application:

1.) The commodity classification system desired (out of five possible choices)
2.) If countries should be included
3.) If districts should be included
4.) Whether the user wants imports or exports

It's a simple and concise way to operate. Just select from the list. The complaint was "**There are over 90 different options!**" For the new user, it's intimidating.

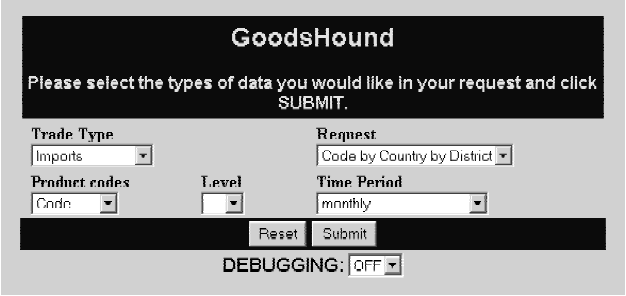


Figure 2 Report selection menu of "GoodsHound"

The same screen in GoodsHound (Figure 2) has five menus:

1.) Trade type: imports, exports or balance
2.) Request: The basic type of request (e.g. "Code by Country by District", "Code by Country", etc.)
3.) Product codes: The commodity classification system
4.) Level: Works in tandem with "Product codes" to define the detail of the data needed.
5.) Time period: monthly, annually or monthly comparison

Up to five selections in GoodsHound as opposed to selecting from over ninety options in "The Flying MDDB".

"Much better," said our customers.

The rest of the application followed suit.

INITIAL PHASE

The initial phase of GoodsHound helps to generally define the request.

1.) What is the timeframe of the request? Monthly data? Annual data?
2.) Are you looking for imports, exports or trade balance?
3.) What type of data are you looking for? Products? Countries? Some combination?
4.) If you are looking for product information, what kind are you looking for and how detailed should it be?

For the most part, this screen is just a collection of HTML pull-down menus. "Product codes" and "Level", however, use JavaScript to add flexibility. Select "HS" from the "Product codes" menu, "Level" will offer options such as "10", "6", "4" and "2". However, if "SITC" is selected under "Product codes", the options under "Level" change to "5", "4", "3", "2" and "1".

SUBSET PHASE

Now that the application knows generally what the user wants, it needs more specifics.

"You want 10-digit HS product information? Great. Which codes?"

Based on selections in the initial phase, the application pulls pieces of pre-defined HTML and displays them to the screen. It works much the same way as a server-side-include might when coupled with a scripting language. For example, if a user selected "Imports", "Code by Country by District", "HS", "10" and "Monthly" during the initial phase of the application, the server would pull HTML that included a selection lists for all available months (Figure 3), all valid 10-digit HS product codes (Figure 4), all countries (Figure 5) and all districts. From the user's perspective, it's transparent.

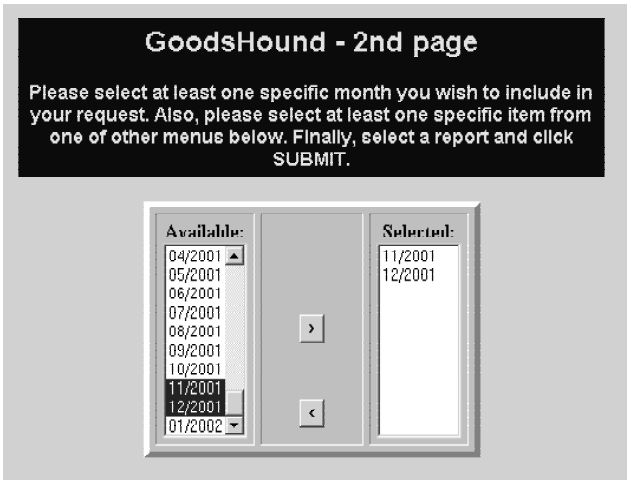


Figure 3 GoodsHound (subset phase): available months

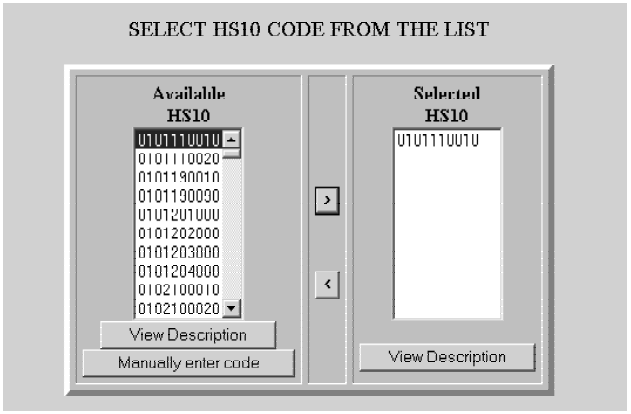


Figure 4 GoodsHound (subset phase): available product codes

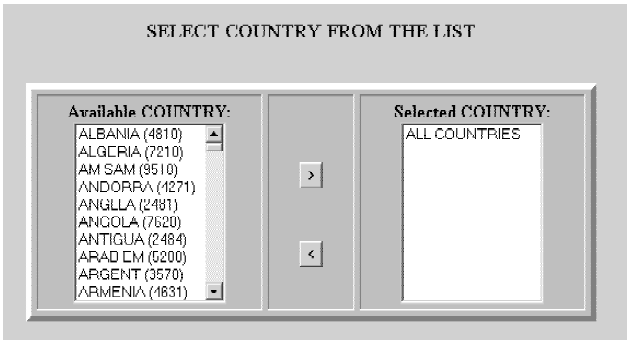


Figure 5 GoodsHound (subset phase): available countries

Like most SAS applications, this application gives the user a list of AVAILABLE items from which to select (on the left) and lists those selected items under the SELECTED menu (on the right). This is possible through the use of JavaScript. The label and value of the items selected on the AVAILABLE menu are copied to the next available space on the SELECTED menu when the "right arrow" button is pressed.

Additional JavaScript functionality is built into the list of product codes. Not everyone wants to scroll through and select from a list of over 15,000 options. So, if they click the button labeled "Manually enter code", they can type in multiple codes separated by spaces (Figure 6). Once the user clicks on the "OK" button, those items are added to the SELECTED list (Figure 7).

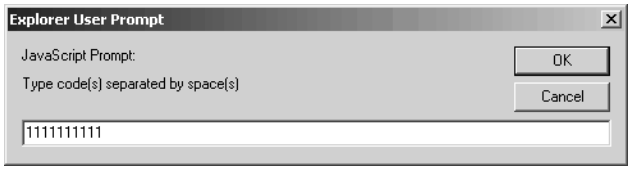


Figure 6 Manually entering codes

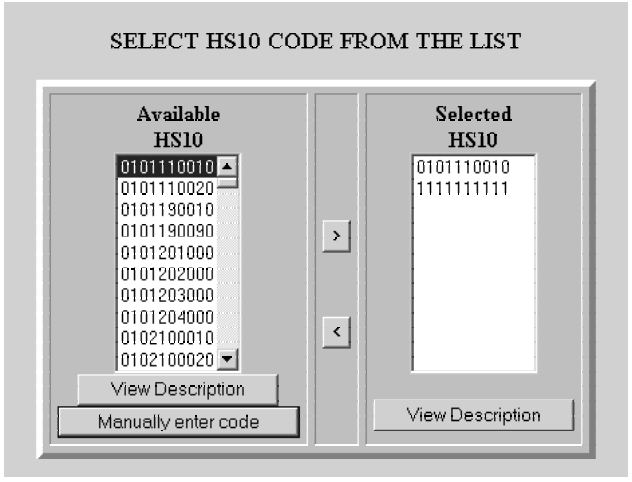


Figure 7 Manually entered code should now be on the SELECTED list

Also on the list of product codes is a button labeled "View Description". Because of the volume of product codes, it's easier to place a request base don the code that to try to identify it by it's description. However, sometimes the users need to make sure they're selecting the right code by viewing the description. If they select a code and press the "View Description" button, they will (Figure 8).

Figure 8 View description

Code

To send multiple selections to the next program, we took a different approach. Initially, we tried to process all of the selected items in one menu into a formatted string that could fit neatly into a WHERE clause (i.e. "where (country in ('1010','1220','1610'))". This didn't work as well as expected. Our work-around was to forward all selected items from one menu as a string and parse it once it got to the SAS program. The HTML would include the following JavaScript.

```
<script language="Javascript">
function makeStringFromSelect(selectCtrl) {
        var i;
        var j = 0;
        var outlist = "";
        for (i = 0; i <
selectCtrl.options.length; i++) {
                if (j > 0) {
                        outlist = outlist + '';
                }
                outlist = outlist +
selectCtrl.options[i].value;
                j++;
        }
        return outlist;
}
```

Along with that JavaScript, several menus would be available. One of which would likely be the COUNTRY menu.

```
<INPUT TYPE="HIDDEN" NAME="SelCtry">
<big> SELECT COUNTRY FROM THE LIST</big>
Available country:<BR>
<SELECT  NAME="AvailItemsCtry" SIZE="10">
<option value='1010'>1010 (GREENLD)</option>
<option value='1220'>1220 (CANADA)</option>
<option value='1610'>1610 (SP MQEL)</option>
<option value='2010'>2010 (MEXICO)</option>
<option value='9610'>9610 (N MAR IS)</option>
<option value='9800'>9800 (US O IS)</option>
</SELECT>
<INPUT TYPE="BUTTON" NAME="AddBtnCtry" VALUE="
ADD  " OnClick="Add_onChange_Ctry(this)">
<INPUT TYPE="BUTTON" NAME="RemoveBtnCtry"
VALUE="REMOVE"
OnClick="Remove_onChange_Ctry(this)">
Selected country:<BR>
<SELECT MULTIPLE NAME="SelItemsCtry" size=10>
<OPTION VALUE="00">ALL COUNTRIES</OPTION>
</SELECT>
<INPUT TYPE="BUTTON" VALUE="Submit"
OnClick="this.form.SelCtry.value =
makeStringFromSelect(this.form.SelItemsCtry);
this.form.submit(); this.form.reset();">
<input type='hidden' name='_program'
value='mig.mig_5_pgm.sas'>
<input type='hidden' name='_service'
value='default'>
<input type='hidden' name='_debug' value='131'>
```

Items are moved from the "Available" menu to the "Selected" menu (see included screen shots). When SUBMIT is clicked, the JavaScript makes a long string of all the items selected from the indicated menu.

```
%macro co;
     %do i = 4 %to %length(&selCtry) %by 4;
          "%substr(&selCtry,%eval(&i-3),4)"
          %if &i ne %length(&selCtry) %then %do;
          ,
          %end;
     %end;
%mend co;
```

if for instance VALUE of &selCtry = 101012201610
macro %co will transform this value to "1010","1220","1610"
which we can use in WHERE clause

```
data one;
     set in.hsil0999(where=(country in(%co)));
run;
```

is equal

```
data one;
     set in.hsil0999(where=(country
```

```
      in("1010","1220","1610")));
run;
```

REPORTING PHASE

Output options for GoodsHound are:

1.) A list of predefined reports (availability depends on definition in the initial request)
2.) Standard report: Essentially a dump of the requested data. If the user wanted "Code by Country" data, whatever they selected in the subset phase is displayed "Code by Country". There is no flexibility.
3.) Advanced report: If the user has select lots of detailed data but only wants to look at a portion (e.g. select "Code by country by district" and only wants to view "Code"), this report offers the flexibility.

PREDEFINED REPORTS

There are some legacy reports that this application needs to support. Their availability depends on the request, but they are always the same. Same format. Same content.

Previously, however, they were just text files with little to no formatting except for spaces.

```
Report EM545              U.S. Exports of Domestic and Foreign Merchandise
                                By all Methods of transportation
October - 1998                Current month and Cumulative, January to Date
                                      Current Month                                Cummulative, Ja
Country    District      Count  Quantity 1       Quantity 2            Value      Count  Quantity 1

1207200010   COTTON SEEDS FOR SOWING
MEXICO     LAREDO          0            0 KG              0                0         14       2398198 KG
MEXICO     EL PASO         0            0 KG              0                0          8        188063 KG
MEXICO     SAN DGO         0            0 KG              0                0        210       4626109 KG
MEXICO     NOGALES         0            0 KG              0                0         12        244944 KG
MEXICO     TOTAL           0            0 KG              0                0        244       7457314 KG

SALVADR    MOBILE          0            0 KG              0                0          1         18144 KG
SALVADR    TOTAL           0            0 KG              0                0          1         18144 KG
```

Figure 9 IM145 - Plain text

GoodsHound offers HTML and PDF (Figure 10) versions through the use of the Output Delivery System (ODS).

IM145 - U.S. General Imports by All Methods of Transportation, Current Month and Cumulative

MONTH/YEAR=NOV2001 Commodity=0101110010 HORSES, LIVE, PUREBRED BREEDING, MALE

					Current month					Cumulative year to date				
Country	District	CSC	Un1	Un2	Count	Qty-1	Qty-2	Custom Val	C.I.F. Val	Count	Qty-1	Qty-2	Custom Val	C.I.F. Val
CANADA	ST ALBN	0	NO		0	0	0	0	0	6	55	0	40,986	41,545
	OGDENSB	0	NO		2	2	0	5,500	5,650	15	44	0	196,664	198,310
	BUFFALO	0	NO		2	17	0	66,481	67,624	9	28	0	125,301	127,470
	SEATTLE	0	NO		0	0	0	0	0	1	2	0	7,000	7,180

Figure 10 IM145 - PDF version

STANDARD REPORT

The standard report is a "you asked for it, you got it" option. If the user said they want "Code by Country by District" data, that's what they get. No flexibility. It's a good option for those that are just checking data or don't need slick output.

Imports, Commodity x Country x District, Monthly, HS 10-digit
Statistical Month and Cumulative

COMMODITY: 0101110010 HORSES, LIVE, PUREBRED BREEDING, MALE (NO)

MONTH: November YEAR: 2001

		November			JANUARY to November		
Country	District	QTY1	CUSTOM VAL	C.I.F. VAL	QTY1	CUSTOM VAL	C.I.F. VAL
CANADA	ST ALBN	0	0	0	55	40,986	41,545
CANADA	OGDENSB	2	5,500	5,650	44	196,664	198,310
CANADA	BUFFALO	17	66,481	67,624	28	125,301	127,470
CANADA	SEATTLE	0	0	0	2	7,000	7,180
CANADA	ALASKA	0	0	0	1	7,500	7,800
CANADA	GTFALLS	0	0	0	5	9,342	9,483
CANADA	PEMBINA	2	55,000	55,225	26	163,695	165,720
CANADA	DULUTH	2	3,799	3,849	0	39,159	39,699
CANADA	DETROIT	61	70,741	72,725	101	150,674	155,979
* Subtotal from CANADA		86	201,521	205,073	270	740,321	753,186
MEXICO	LAREDO	0	0	0	2	2,684	2,684

Figure 11 Standard report

ADVANCED REPORT

The advanced report has all of the flexibility that the other options lack. The user selects what they want to see. Which variable should be going down? Across? Which analysis variables should be in the columns?

Much like in the subset phase, depending on the definition of the request, the application will assemble pre-defined sections of HTML so the user can describe what they want their advanced report to look like.

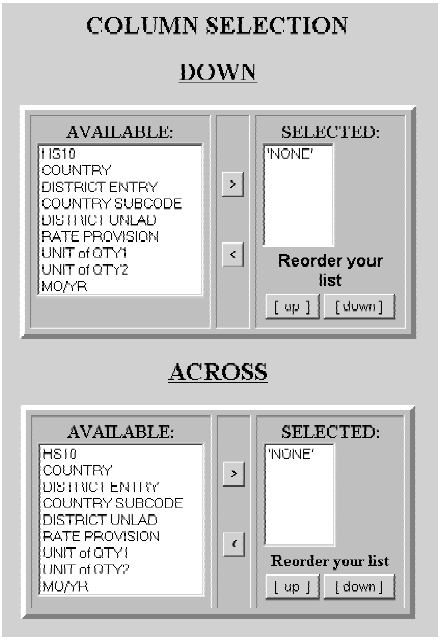


Figure 12 Advanced report - Down and across

Also, like the subset phase, JavaScript is used to move items from the AVAILABLE menu to the SELECTED menu (Figure 12). This page, however, has an added feature.

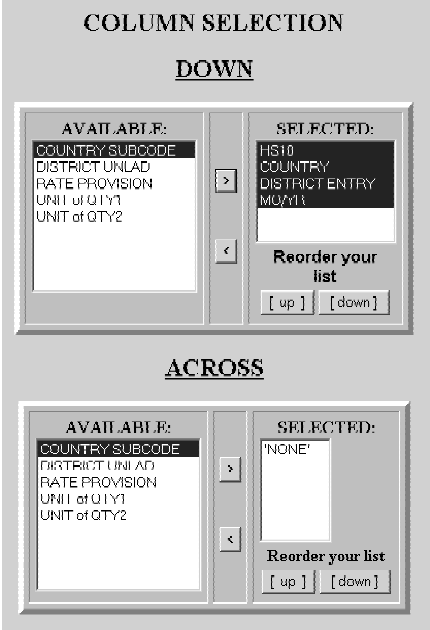


Figure 13 Advanced report - Added feature

When an item is moved from AVAIALBLE to SELECTED under the DOWN section, it's removed from the AVAIALBLE menu under the ACROSS section (Figure 13).

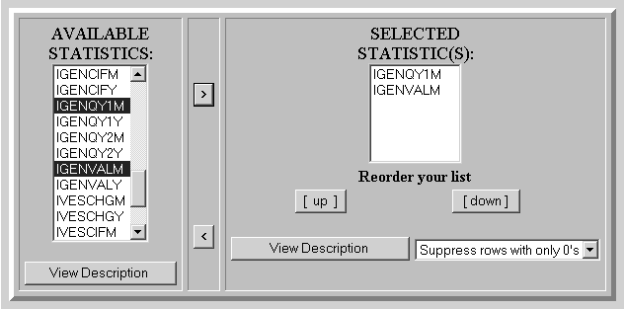
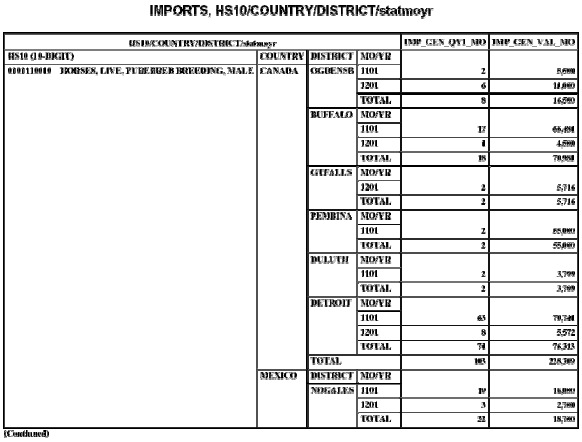


Figure 14 Selected analysis variables

After selecting the analysis variables for the report (Figure 14), the advanced report funnels all of the parameters that have been selected and cerates an HTML or PDF report (Figure 15) using PROC TABULATE and ODS.



IMPORTS, HS10/COUNTRY/DISTRICT/statmoyr

HS10/COUNTRY/DISTRICT/statmoyr				IMP_GEN_QY1_MO	IMP_GEN_VAL_MO
HS10 (10-DIGIT)	COUNTRY	DISTRICT	MO/YR		
0101110010 HORSES, LIVE, PUREBRED BREEDING, MALE	CANADA	OGDENSB	1101	2	5,000
			1201	6	11,000
			TOTAL	8	16,000
		BUFFALO	MO/YR		
			1101	17	68,491
			1201	1	4,500
			TOTAL	18	72,991
		GRFALLS	MO/YR		
			1201	2	5,716
			TOTAL	2	5,716
		PEMBINA	MO/YR		
			1101	2	55,000
			TOTAL	2	55,000
		DULUTH	MO/YR		
			1101	2	3,709
			TOTAL	2	3,709
		DETROIT	MO/YR		
			1101	63	70,744
			1201	8	5,572
			TOTAL	71	76,313
		TOTAL		103	228,389
	MEXICO	DISTRICT	MO/YR		
		NOGALES	1101	19	16,000
			1201	3	2,700
			TOTAL	22	18,700

(Continued)

Figure 15 PDF report produced through ADVANCED REPORT

An "advanced report" based feature that creates Excel compatible files is also available.

THE BEST CHOICE (CONCLUSION)

Solutions like GoodsHound are not the only choice. However, when using something that's not an "out of the box" solution like SAS/MDDB files, there is a lot of work that needs to be done: planning, maintenance, troubleshooting, etc. Building an application like GoodsHound has given us a great tool that shows lots of promise, but SAS/MDDB-based applications still offer the least amount of work for the developer.

REFERENCES

Blake Sanders and Mikhail Gruzdev, *The SAS/MDDB Shell Game: Options They Don't Show in SAS/HELP* , SUGI 25 (2000), Paper 35-25

CONTACT INFORMATION

1.) Blake R. Sanders
U.S. Census Bureau
Foreign Trade Division
FB 3, Rm 2158
Washington, D.C. 20233

Phone: 301-763-2234
E-mail: blake.r.sanders@census.gov

2.) Mikhail Gruzdev
U.S. Census Bureau
Foreign Trade Division
FB 3, Rm 2158
Washington, D.C. 20233

Phone: 301-763-2234
E-mail: mikhail.g.gruzdev@census.gov