Paper 2

IMPROVING SAS® DATA ACCESS - AN EVOLUTIONARY EXPERIENCE

(From Input Statements to Data Forms)

Bernd E. Imken, Patented Medicine Prices Review Board, Government of Canada, Ottawa, Canada

Abstract:

During a recent SAS® Users Group presentation in Ottawa the audience was asked to identify how they input/edit/view data. 50% said they use the Data step and Proc Print, another 45% said they used Proc FSEDIT, FSVIEW or VIEWTABLE, 5% said they developed custom AF screens. Out of this 5%, only one user said that they have experimented with Data Forms and Data Tables.

The purpose of this paper is to identify the strengths and weaknesses of each approach and in particular to show why organizations such as the Patented Medicine Prices Review Board (PMPRB) benefit so significantly by using the more recently developed SAS facilities for user interaction. This paper describes how the Board evolved over the last 10 years. You may find that your organization is currently at some point on this same development time line.

Introduction

The PMPRB was established as a judicial agency by the Canadian Government to ensure that prices of patented medicines are not excessive.

Nearly $4 Billion of patented medicines are sold each year in Canada. There are, approximately 70 manufactures marketing nearly 1,000 patented medicines within Canada.

The PMPRB SAS Data Base

Legislation in Canada requires all manufactures to report the following to the PMPRB:

- **Product Data:**
  - generic name, brand name
  - date product introduced on the market
  - patent number, patent dates, patent holders
  - manufacture name, address
  - other data such as, usage, ingredients etc.

- **Canadian Sales Data:**
  - details must include package size, price per package, and quantity sold
  - sales broken down by geography (e.g. Province)
  - customers, including, hospitals, wholesalers and pharmacies

- **International Price Data**
  Ex-factory prices for Germany, Sweden, UK, Italy, France, Switzerland and the USA
  - details must include prices for hospitals, wholesales and pharmacies and the associated price currency.

- **Other Data**
  - Includes exchange rates on a monthly basis for all countries being monitored.
  - Consumer Price Index data including forecasted and actual values.

The data base currently contains over 8 million observations. Each data set is fully indexed to improve performance. You may access any part of the data base either with a screen or a report in less than 2 seconds.

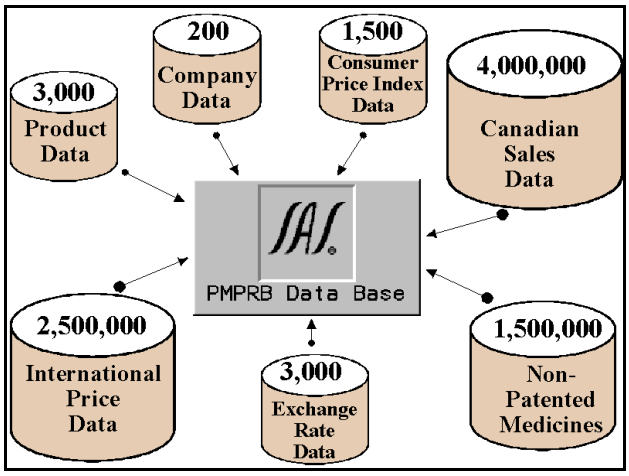

Figure 1 Logical View of the PMPRB Data Base

The PMPRB Network Configuration

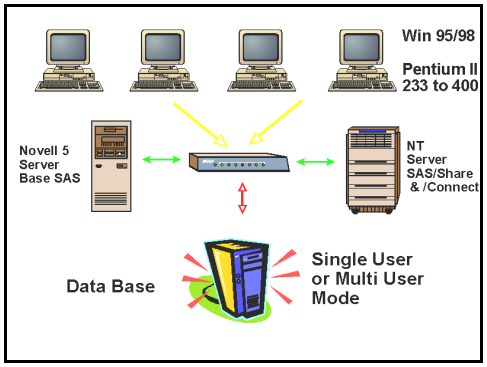

Figure 2 Simplified View of the PMPRB Network

Figure 2 displays the PMPRB network configuration. The Board runs both Novell and NT servers. All SAS processing is on the NT server (Pentium Pro 200). This server runs SAS 6.12 (Server Version) and provides the users with both SAS/Share and SAS /Connect Software. Each user has a Pentium 233 with 64 Meg Ram.

The Original PMPRB Approach (1987)

The PMPRB originally accessed data by programming simple data steps and reports. An example is displayed below.

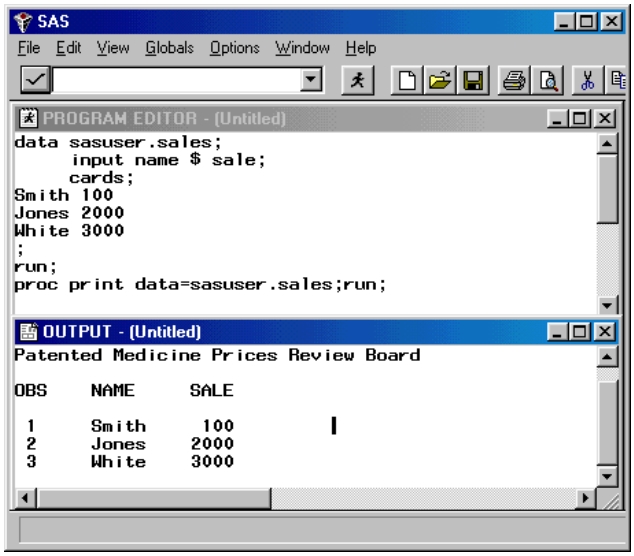

Figure 3 The use of the Input Statement and Proc Print

Twenty percent of the data received at the PMPRB was in the form of Lotus\Excel spreadsheets. The PMPRB quickly adopted the INFILE statement, to import ASCII files, and, adopted Proc DBF to read Lotus files. During the first two years of operations these techniques worked well. Data input personnel reported directly to the SAS programers, since direct interaction was frequently required. The approach proved to be an expensive use programming resources.

Other advantages and disadvantages also came to light:

Advantages:

✔ Just BASE SAS required
✔ Inexpensive

Disadvantages

✗ Use of delimiters (e.g. INFILE testin delimiter=';')
✗ No data verification
✗ Spacing important
✗ Multiple steps
✗ Not user friendly, must know SAS

Using this approach, the PMPRB found that there were many inherent problems which arose resulting in incorrect keypunching, losing data records, training input personnel etc. The search was quickly begun for an alternate solution.

The Middle Era (1991-1997)

During this period of time the quantity of data was rapidly increasing. The old approach became increasingly slower. The lack of indexing (version 6.04 and before) caused such poor performance that a search for other data base solutions was commenced, in 1992, using an external consulting agency. A competition was held between SAS and other vendors such as Paradox, DBASE, Access etc.

Late in 1992, the programmers devised a method of creating indexed data sets containing the KEY and only the observation number. These files could be searched easily and unique observation numbers could be retrieved quickly, which could then be passed on to the data set with the POINTER= option. The performance problem was resolved.

In 1993, SAS was officially declared as the winner of the competition and the development team was requested to continue development using SAS as the tool for it's mission critical systems.

Next, to improve user interface programmers suggested the acquisition of SAS/FSP to facilitate the design of screens and menus.

If the data set already existed then:

PROC FSEDIT data=sasuser.sales
screen=sasuser.salesscr ;
RUN;

If the data set did not exist and the developer wanted to create a new one, then the "DATA=" was replaced with "NEW=". The process could not be much simpler. SAS not only produced defaults screens, but also allowed you to do ADDs, FINDs, WHERE, NEXT SCREEN etc. by simply making clickable selections from the PMENU bar. At the time facilities as simple as this did not exist in the other data base products such as DBASE or Paradox. The programmer soon developed many kinds of customized screens - it was simply a matter of typing MOD on Command line to access the facilities to customize a default screen.

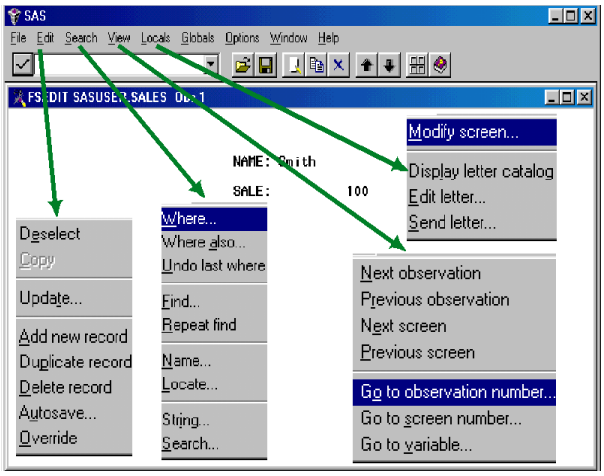

Figure 4 Using the Default FSEDIT Screen

Screen modifications allowed the developers to “paint” the screen with field descriptions, titles etc., making it much easier for users to adopt. Now users only had to hit TAB or ENTER to navigate from one field to another, without having to worry about columns, delimiters or the other restrictions they had with the INPUT statement used previously.

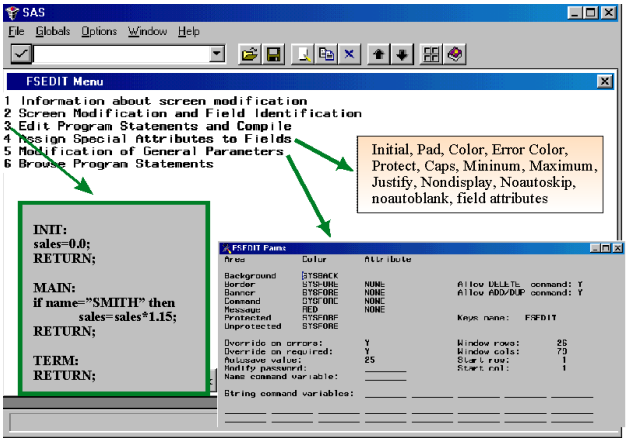

Figure 5 Capabilities of MOD

The procedure also allowed the developers to specify many options for each input field, including field color, protected fields, non-display fields, etc. This proved to be useful to the developers, reducing field validation requirements significantly. A great deal of the initial checking code could now be easily eliminated.

In addition, customized screen processing could easily be added to each screen such that calculations could be performed in front of the users each time the enter button was pressed. Also background colors could be set, options to ADD or DUPLICATE could be set, and search/string parameters could be identified.
Next, the programmers began using customized MENU selections. Proc PMENU proved to be powerful and yet easy to use, giving the developers far more control within the application.

In 1994, SAS/AF was acquired allowing the programmers to develop custom MENU screens, as well as screens for HELP and LIST. PROGRAM screens produced in SAS/AF proved to be even more powerful initially . The system could now be designed for an end user who had virtually no knowledge of SAS or SAS coding. The input personnel were now moved to within the operational units, much closer to the end users. This allowed time to be freed up for the development team.

One other significant breakthrough of this time was discovering that the screen developed in FSEDIT could also be used to display information from more than one data set using a number of simple SCL statements.

You can add variables from other data sets to the screen by creating a new screen variable

trade = getvarc(dsid,varnum(dsid,'trade'));

(Note: The data set must be opened first as DSID, this can be done using the SCL OPEN function, corresponding observations may then be jumped to using either WHERE or FETCHOBS, or any number of other SCL functions)

Large quantities of data required developing a dual entry system to improve the accuracy. A FSVIEW screen allowed each data entry person the ability of entering one observation per row, just like in a normal spreadsheet. Once the “A” data set was entered, the corresponding “B” set was entered by another individual. When both were done an automatic record by record comparison was performed identifying inconsistencies even at a field level. Having such stringent quality control is a necessity since the data could in the future be used as evidence in a judiciary ruling.

The use of FSEDIT/FSVIEW especially in a SAS/AF environment proved to have many more advantages, but, also, we became aware of certain disadvantages.

Advantages:

✔ Powerful
✔ Reasonable error checking, min max etc.
✔ Ease of use for data input
✔ Good color control on screen
✔ Works well in SHARE mode

Disadvantages

✗ Character based screen does not permit graphics
✗ Images not allowed
✗ No push buttons, radio buttons etc.
✗ Not up to date with competition (ACCESS etc)
✗ Little control over record locking
✗ Cannot easily blend full screen look with FSView (spreadsheet appearance)

Below can be seen an actual FSEDIT application develop using FSEDIT at the Board. The first screen shows some of the basic data collected on each medicine.

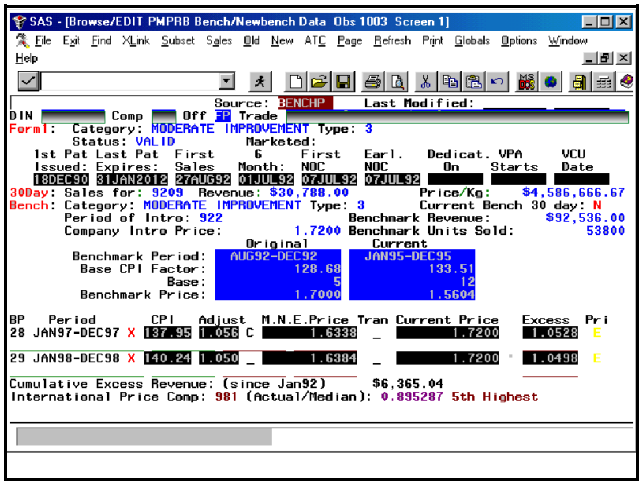

Figure 6 One of the major PMPRB FSEDIT Screens

This screen shows only data for Jan97 to Dec98. Subsequent screens show, in increasing detail, data going

back to 1987. In total there are 6 screens containing nearly 3000 SAS variables. Note the users can navigate through the screens using the customized MENU. This screen had nearly 4,000 lines of SCL code allowing online update. A full REFRESH could take place in less than 2 seconds.
Although the users were very pleased with this design, the development team found it onerous to constantly have to add variables to the screen to accommodate data base growth. This was caused by historical records having to be mapped to SAS ARRAYS for presentation on the screen. All of this was required because scroll down facilities did not exist for FSEDIT screens - nor could one easily populate a FSVIEW screen over an FSEDIT screen.

As a result the screen became more and more difficult to maintain. An alternate solution was required.

The Intermediate Stage

Beginning with SAS Version 6.10 and continuing with SAS Version 6.11 the PMPRB noticed changes which were happening indicating a move to a graphically based environment.

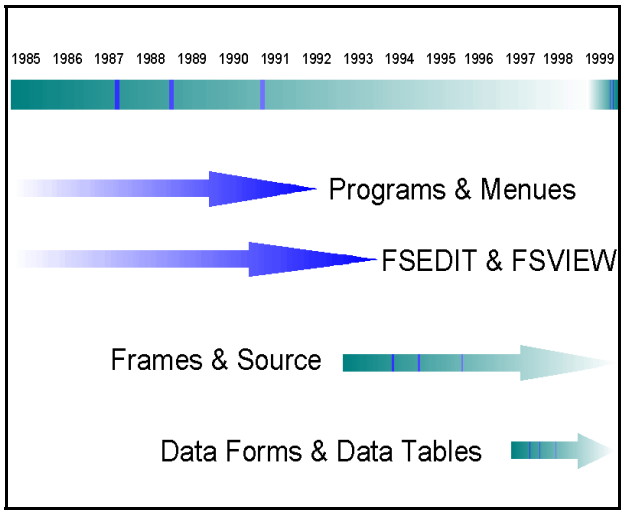

Figure 7 Evolutionary changes

The first sign was the introduction of FRAMES in SAS/AF. This quickly took over from PROGRAMS and MENUS. It was now possible to develop radio boxes, list boxes etc.. It was also the first step towards OBJECT ORIENTED programming.

Evidence appeared in other areas in SAS development which demonstrated that their was a concerted effort by SAS to conform more to the new graphical techniques. One of the first pieces of evidence was the move to VIEWTABLE.

Immediately one could see the use of buttons, supporting multi-line labels rather than the character based profile of FSVIEW.

Also when one clicked to view the data one observation at a time they saw the first Data Forms being presented.

O

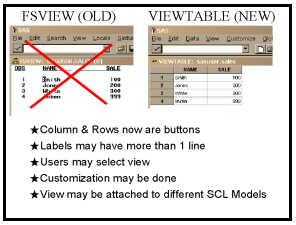

Figure 8 Moving to Viewtable

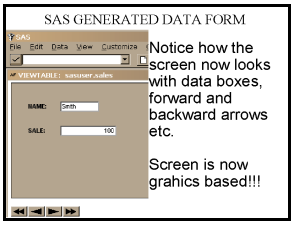

Figure 9 Default DataForm

bviously SAS was working extensively to allow graphical user interfaces which were competitive to Microsoft Access and other relational data base products current on the market at the time.

The Current Period (1998-1999)

The PMPRB started developing graphical object starting in 1997 with SAS Version 6.10. First the development team learned that they could develop menus far better than those allowed by PROGRAM entries by using SAS/AF FRAMES. This technique proved to be very easy to implement and soon all MENUs had disappeared in favor of icons etc. Next the team discovered that the graphical interface provided by a FRAME could also be applied to PROGRAM entries. The users much preferred to use radio selection boxes, list boxes, input objects etc.

Control over multi-sized fonts allowed the developer to bring emphasis to various sections of the screen. It was a please to develop FRAMES using CONTROL BOXES, IMAGE TEXT etc. - the variety at that time seemed almost endless.

Within a short period of time most PROGRAM entries had also been replaced with FRAME entries, however the FSEDIT screens remained. The development team discovered that developing FSEDIT screens was still faster than having to develop a new FRAME were every object had to be identified separately. AF FRAMES had to be linked with a data set using more complex SCL code - each screen field was treated as a separate object. One could not navigate up or down through a data set without having to program SCL code. AF frames were a great substitute for PROGRAMS but could not easily replace the speed, ease of development and default characteristics of FSEDIT.

Finally, with the introduction of SAS Version 6.10 this problem appeared to be starting to be resolved with Data Forms (Experimental in 6.10) and Data Tables as object within the SAS/AF environment.

The development team eagerly started to convert it's most elaborate FSEDIT screen to DATA FORM late in 1997. Early development proved to be difficult especially for a product still in the Experimental stage. Documentation was scarce at the time and the Board found itself relying heavily on SAS Institute in Cary for additional support.

Some of these problems were gradually resolved as SAS Version 6.12 was introduced. Finally now Data Tables can be displayed on the same screen as a Data Form and linked accordingly.

The screen displayed at the top of the next page, shows the multi-page Data Form which was developed to service the application. (Figure10)

The DATA FORM shown has four pages, selectable by clicking on the fold at the top of the page. The DATA FORM itself is shown positioned on a TAB OBJECT newly available within SAS/AF. Other pages on the data form use radio buttons and check boxes to improve user accessibility.

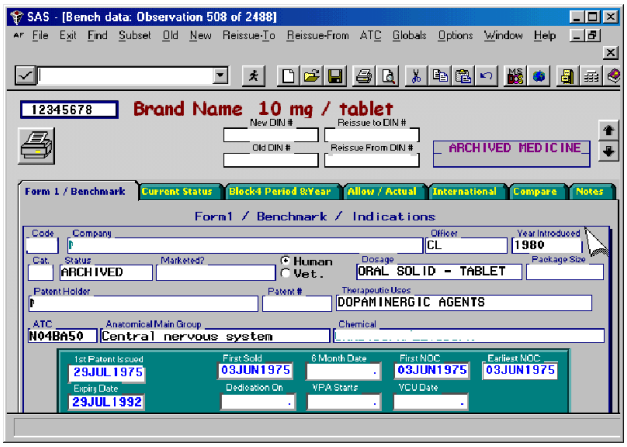

Figure 10 The new DataForm appraoch being used at the PMPRB

Above the TAB object you will also notice other fields which are redundant to the data forms and tables located on the TAB object.

Notice the use of:

- ❑ different font sizes and styles
- ❑ labels associated directly with a data region
- ❑ icons to print out results for the entire screen on one page
- ❑ icons to allow the user to navigate with the click of the mouse.

When the user presses the third TAB a screen is displayed showing two Data Tables which correspond or are linked to the Data Form which was on the first TAB.

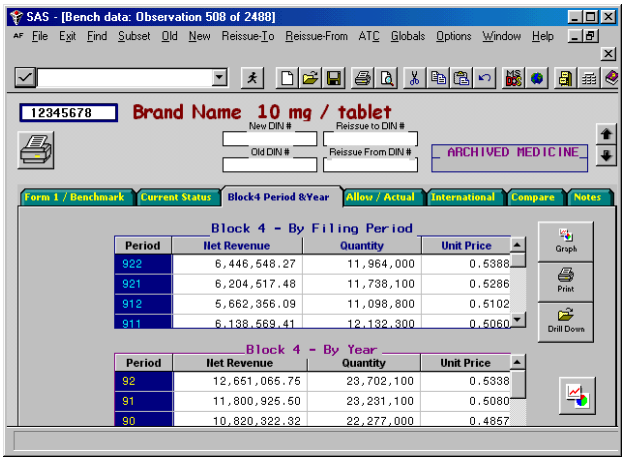

Figure 11 View of period & Yearly sales on another Tab

Both Data tables are independently scrollable with their own vertical scrollbars. The icons on the right permit the user to obtain a printout, a graphic presentation, and in addition permits further drill down, since the data represented in the Data Tables is of a summary nature.

What have been the major improvements for the PMPRB?

- ❑ Easier to navigate for users
  - Scroll bars allow users improved access
- ❑ Scrollable objects reduce maintenance
  - In the previous FSEDIT version of the screens, the developers would have to add new screen variables for new yearly periods
- ❑ Improved drill-down capability - can use buttons and icons for representations

At this point in time the PMPRB is still makes both the FSEDIT and Data Form technologies available. Each has it's own merits.

Conclusion

Each of these interfaces discussed in this paper has it's own place in the SAS world - it depends totally on what the user is trying to accomplish, and how much experience and time is available to the user.

BUT WHY GO THROUGH ALL THIS EXTRA WORK!!!

- ❑ Easier for users to operate
- ❑ Reduced SAS end user training required
- ❑ Improved control over what the user sees and does
- ❑ Better control over development
- ❑ Keep up with tools like Oracle Access etc. but everything still in one environment
- ❑ Excellent performance results

In the graphic below I have tried to summarize the relative aspects of using one approach versus the other. A small poll was conducted of the various SAS developers and users to see how they positioned each technique. The common comments which were rendered with the poll are included below the charts.

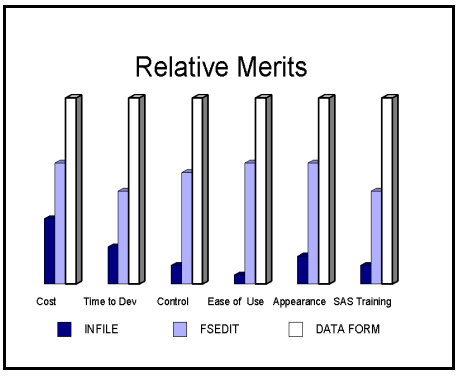

Figure 12

- ❑ COST: Most users did not know about SAS software costs. Developers recognized that only

Base SAS was required for the INFILE approach. Using FSEDIT required the additional purchase of SAS/FSP. Using DATA FORM required purchasing SAS/AF, as well as modules such as SAS/GRAPH if graphics were to be used.

- TIME TO DEVELOP: The INFILE statement is frequently one of the first things learned. FSEDIT required the developer a short time to become familiar but reasonably sophisticated screens could be developed in 1-2 hours. Data Form developers indicated they had to learn far more about SCL, object programming etc. This was not recommended as an approach unless the developer had a strong background in AF and SCL.

- CONTROL: End users recognized that the real time feedback given to verification and control was valuable especially using FSEDIT and DATA FORM. Developers tended to give DATA FORMs more weight as a control mechanism because they could use object such as check boxes and radio buttons etc.

- EASE OF USE: Most comments here were received from end users. Data Form performed best - easier to use because of the graphic interface. Users like differing font sizes, and graphical elements such as buttons, boxes etc. They also indicated it was easier to navigate around - “like having a FSEDIT screen and a SPREADSHEET all on the screen at one time”.

- APPEARANCE: Both end users and developers liked like the look of the DATA FORM screens “It looks better. IT looks more like what we are used to in Windows 95".

- DATA FORM was veiwed as being more intuitive. The developers however had very strong opinions on the amount of SAS training required for development. Introductory developers could handle INFILE easily. FSEDIT was used by either advanced introductory programmers or intermediate one, DATA FORMs were only used by the experienced programmers, frequently also with advanced training in AF, SCL etc.

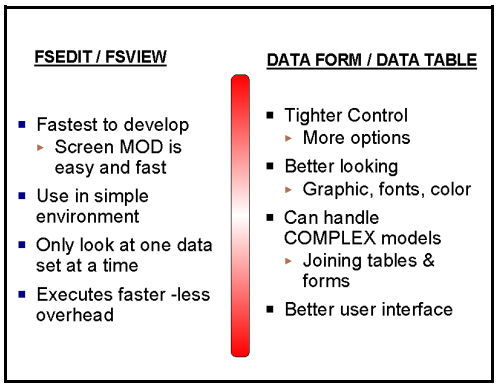

Figure 13

For those of you who have not experimented with FSEDIT and DATA FORMS I strongly advise you to give them a try. Once management and the end users see systems developed/customized in this way their level of satisfaction increases substantially.

References

- **SAS on the WEB**
  - Many articles exist on such things as controlling tables, multi-line labels etc.
- **Object-Orientated Programming with the SAS System**
  - 4" thick manual publish last year by SAS
  - Useful for all FRAME development work
  - For experience SCL users - very powerful
- **23 Annual SUGI Proceeding**
  - Marge Scerbo and Alan Wilson, *Tables and Views and Forms: Oh My!*
  - Derek Morgan and Mike Province, *A Bag of FSEDIT Tricks*
- **Solution@Work CD Rom from SAS**

Acknowledgement

I would like to give special thanks to:

Ann Carpenter Rockett **Lynn P. Leone**

from SAS Institute, Cary, North Carolina,for the assistance and advice they have given me in developing Data Forms & Tables for the PMPRB.

Contact Information

Bernd E. Imken
Chief, Information Systems Division
Patented Medicine Prices Review Board
333 Laurier Avenue West
Ottawa, Ontario K1P 1C1
Phone 613 952-3312 Fax 613 952-7626
E-mail bimken@pmprb-cepmb.gc.ca