When the question of customizing training for SAS software users is raised, a common response is to take existing generic instructional materials and combine, re-structure, subset and augment them to assemble a hybrid curriculum that satisfies the needs of the students. Frequently, however, one major element is missing from this approach — the immediate relevance of the examples shown in the materials to the students' daily on-the-job tasks.

The sample data and applications included in most vendor-produced training products are not associated with the students' daily on-the-job tasks. Data to be integrated into "off-the-shelf" training materials.

For industry-based consortiums, a set of training manuals specific to that industry's applications can be created, i.e., materials particularly relevant to pharmaceutical firms, insurance companies, health care organizations, utilities, etc.

Our customization technique will be illustrated using excerpts from one-day seminars on PROC TABULATE and PROC SQL composed by the staff of CSC Professional Services Group.

DEFINING THE CURRICULUM

The traditional methods of producing customized training fall into two main categories: (1) writing the course materials "from scratch" and (2) combining selected modules from one or more existing courses into a single course. The first approach, composing a custom course, either in-house or through an independent consultant or training company, is expensive, so many organizations have chosen the second strategy. Among the companies which have chosen to create a hybrid instructor-based curriculum are American Cyanamid (Murto, 1991) and Texas Instruments (Scott, 1992).

In-house staff at American Cyanamid were capable of teaching a course, but they had limited time for course development. They chose to purchase the SAS Trainer's Kit for the SAS Fundamentals (Version 6) course. Some sections of this course were eliminated since they dealt with SAS Institute products that were not licensed at the site. Sections that were kept received increased emphasis at the students' request. Sections that discussed site-specific topics (file structures, naming conventions, the host editor) were written and inserted.

The approach used at Texas Instruments (TI) also resulted in a hybrid curriculum after consultation with the SAS Institute training staff. The equivalent of 14 days of training in base SAS, SAS/GRAPH®, SAS/STAT® and SAS/QC® software was pared down into a four-day course taught at a TI site by a SAS Institute instructor.

The hybrid curriculum concept is part of the approach implemented the Seminars in SAS Programming series offered by CSC Professional Services Group. Although we have formulated standard 1-day, 2-day, and 3-day seminars, we have been asked to re-work the topics by several customers, substituting topics where desired, reducing emphasis of others, and eliminating unneeded subjects altogether.

WHY IS DATA CUSTOMIZATION DESIRABLE?

After an appropriate curriculum is established, making certain that the training materials are valuable to the students, both during and after the class, is a key consideration. Live demonstrations and hands-on workshop sessions during the formal classroom training help keep students more attentive and interested in learning the material. Unfortunately, using data unrelated to the students' day-to-day activities can dampen their enthusiasm. When I have taught classes using generic data, some students have commented about this, saying things like "I wish I could use my own data instead of this fake stuff."

While a quality instructor can assist students with their "real-life" applications during breaks and after class, we felt that integrating the customer's own data into the formal class would improve the students' overall comprehension of the material. Some government agencies already stipulate that course exercises, examples and problems be tailored to the students' environment, employing the agency's terminology. By actively using data that were somewhat familiar to them, the students would be higher on their learning curve following the class than if they used irrelevant data.

One additional benefit to customizing materials for the student audience is the ability of the instructor to speak their "language." It makes sense to students from a clinical trials background in terms of "patients" and "visits" or those working in the banking industry in terms of "branches" and "accounts," and so forth. Our instructors often have had experience in the students' industry through one of our consulting or contract programming projects. Even instructors without this experience, however, learn about
their students’ applications simply by going through the customization process.

**COURSE MATERIAL DESIGN**

Modularity is a key requirement for course materials that are to be customized for many different customers. In designing its own seminars to teach the concepts and applications of the SAS System, we have concentrated on constructing instructional modules that can focus on new concepts while simultaneously building upon old ones.

To create a truly custom curriculum, each module must be, to a large degree, independent from other modules. However, great care is required to prevent the individual modules from becoming too dissimilar, since the smooth flow of a class is critical to its success. Therefore, the examples and workshop exercises in each module should be simple enough to stand alone by focusing on key points while fitting into the scheme of a large course. Adding "clutter" to examples or using unfamiliar or complex techniques unrelated to the principal concepts can confuse or distract even the most eager students.

The order of presentation of various topics must also be considered. A consistent appearance is especially necessary when multiple courses will be taught to the same students. One example of this concept is the automatic inclusion of an introductory section that describes the sample data, usually with descriptive output from PROC CONTENTS and PROC PRINT. The actual LIBNAME statement required for the students to access the data is also provided in this section, along with the FILENAME statements that point to any raw data files used in the course.

Each module of training text must be formatted consistently so that the final course looks uniform after assembly. WordPerfect® was the word processor of choice. We used headers, footers and styles liberally to produce the most attractive final product possible. Another mandatory feature is a very detailed Table of Contents. Some courses also have an Index at the back of the training manual, although the detailed Table of Contents often serves the same purpose.

Since the examples and workshop exercises provide the most valuable benefits of customization, those portions of the course must be of extremely high quality. Unfortunately, we found that composing examples and exercises that can be customized, both easily and appropriately, is quite difficult. This occurs primarily because the code should be interesting yet not too ornate, a rare combination in many SAS applications.

Some of the examples require structured or repetitive SAS code that could be generated by a macro program. For these situations, the SAS code that is physically customized by our automated procedures does not appear in the training manual — those programs write to separate files which are integrated into the materials in much the same way as other non-code portions, e.g., procedure output, are inserted.

**COURSE DATABASE DESIGN**

One of our goals in producing customized training materials was the capability to integrate the same data into multiple courses for the same customer. For instance, a single data set is sufficient for a 1-day seminar on PROC TABULATE, but it is clearly inadequate for a course module covering the MERGE statement.

This goal was one of our greatest challenges. A breakthrough in designing the customizable course materials was the establishment of guidelines for sample databases. Driven by the requirements of course materials, the standardized database structure had to be general enough to suit all types of government and commercial organizations. A simple hierarchical structure was established that contains a demographic "header" file, multiple "detail" files of very similar design, and numerous "lookup" tables.

Among some of the basic requirements of our sample database are:

- persons' names that can illustrate functions that manipulate character strings — some with middle names, some without; some with hyphenated last names
- state postal codes and ZIP codes to illustrate related DATA step functions
- SAS dates, preferably more than one date to permit calculations using date arithmetic
- coded fields — some numeric, some character; most with a limited number of values to minimize printed output; some missing values
- secondary sort variables allowing for FIRST/LAST. examples
- a series of repeated measurements to permit array processing

**IDENTIFYING RELEVANT APPLICATIONS**

Once the subject matter is determined, we consult with the customer to find an application that is relevant to many students and also can meet the overall guidelines of our database. The application must also be simple enough for the instructor to learn prior to presenting the seminar, usually no more than three weeks after customization.

This is the point where our database analysts enter the picture. While the customer probably has data stored in SAS data set form already, the current structure may not be suitable for the customization of course materials. It is the
The responsibility of our customization analyst to determine how each customer's data can be "massaged" to meet the requirements of the training materials.

The sample database guidelines are loose enough to permit the customer's data to be fit into our structure without destroying the relevance of the data to the students' everyday work environment. In fact, there have been occasions when our analysis of the sample data highlighted shortcomings in the customer's database design. By showing the revised structure of the sample database, we have assisted the students in redesigning their existing databases, even when database design is not part of the formal curriculum.

AUTOMATING THE CUSTOMIZATION PROCESS

Up to this stage, the creation of a training seminar is similar to a consulting project. We meet with training officers to decide on the best curriculum for the students enrolled. We also meet with some of the application staff to construct an appropriate sample database. Now, we can use the customer's input to produce training materials specific to that customer.

Not unlike development of training materials, customization can be a labor-intensive activity. We searched for a way to automate the process. Our plan was to build a fill-in-the-blank system that could prompt for information to be substituted into the course materials. This system would also substitute the same parameters into the SAS code used in the examples. The modified SAS code would be executed and the resulting output inserted into the training manual.

Using base SAS and SAS/AF® software on the PC (SAS release 6.04), our staff dedicated hundreds of hours devising a system that could read our training manual text documents (WordPerfect files) in a SAS DATA step, identify certain "flagged" items that represented substitution parameters, prompt for a value for each item, and plug the values back into the WP document. During this operation, the SAS code nested inside the WP document also would be modified, then executed through SUBMIT blocks in the SCL of the SAS/AF application. The results (.LOG and/or .LST files) would be inserted into the WP document at a specified position. This system represented the first generation of our Course Adapter software.

While the overall strategy was appropriate, it became apparent after several months of use that the initial implementation of Course Adapter had several drawbacks. Since entire WordPerfect documents were being scanned for a limited number of flagged items, the process took longer than we had hoped. We were also dependent on the internal structure of the WordPerfect files since WP [COMMENT] codes were used to signal a flagged item. It was decided that an alternate implementation should be investigated.

The original system, initially developed under release 6.04 and converted to release 6.06/6.07 under VMS®, was transferred to OS/2® for modification. The updated version contained a great deal of DATA step and macro language programming, using SAS/FS® procedures for data entry and review. The corresponding WordPerfect documents were made more general, moving most of the substitution elements out of the text portions and into the SAS program examples. The SAS code itself was moved out of the WP files into separate PC files, one file per example. The integration of SAS code and LOG and LISTING output with the WordPerfect files was accomplished with WP subdocuments. The new system is known internally as "Course Adapter II".

CUSTOMIZATION AT THE INDUSTRY LEVEL

There are some circumstances where integrating actual data into course materials is not practical. These situations typically involve confidential or proprietary data which cannot be disclosed to the public. Occasionally, a customer does not want to show "real" data to the class instructor or the person producing the customized training manuals. In other cases, the customer is actually a consortium of companies involved in related businesses; obviously, none of these companies wants to broadcast their data to their competitors.

When issues of confidentiality arise, our database analysts and our application experts for the industry involved are called upon to establish a model equivalent of the real data. As long as the nature of the sample data looks familiar to the students and the proper levels of validity and believability are maintained, the customization process still produces superior training materials for the intended audience.

Since the process of producing materials for a training seminar is still quite labor-intensive, even with the assistance of the Course Adapter software, eliminating the customization phase of the process would lower the cost of training significantly. Instead of courses customized for individual organizations, a series of courses customized for specific industries could be produced. That set of courses could be taught repeatedly at many different sites. The only primary costs would be the class instructor and the reproduction of training manuals, the same as for generic training.

A series of training seminars customized for the pharmaceutical industry could be developed based on clinical trials applications. Another series could use claims processing in the insurance industry as its basis. Yet another series could deal with consumption patterns for customers of electric, natural gas, and water utilities. The list of possibilities is endless.

As mentioned earlier, this type of focused, customized training has one noticeable advantage — the cost savings. However, the potential savings affects different organizations in different ways. Larger companies that
already can afford to conduct classes on-site will save on overall training costs. Individual companies with many SAS users and plenty of equipment but limited training budgets may be able to afford industry-based customized courses.

The benefits for smaller companies are even greater. Training that used to be beyond their reach due to cost considerations becomes more affordable. If several companies each have only a few people requiring training, they can form a consortium to obtain the training — customized for their industry.

SAMPLES OF CUSTOMIZED MATERIALS

On the following pages are samples of customized training materials extracted from the Seminars in SAS Programming series. The samples from the PROC SQL seminar are customized for utilities, mortgage servicing, and pharmaceutical applications. The samples from the PROC TABULATE seminar utilize the same utilities and mortgage servicing sample databases.

During a class, if a student wants to run a particular example using variables other than those shown in the training manual, the student can use his own experiences with the data to try different variations. The connection between the programs and results shown in the training manual and the student's day-to-day work becomes more concrete because the sample data are so similar to the customer's own data.

ABOUT THE AUTHOR

Warren Repole is a Senior Technical Consultant for CSC Professional Services Group in Rockville, MD. He has used SAS software since 1979. He may be contacted at:

CSC Professional Services Group
1301 Piccard Drive, 2nd floor
Rockville, MD 20850
voice: 301/258-5300
fax: 301/258-6897

REFERENCES


ACKNOWLEDGEMENTS

This paper was previously presented at the Sixth Annual Northeast SAS Users Group Conference, November 7-9, 1993.

Course Adapter is a proprietary software program of CSC Professional Services Group. The concept for Course Adapter software was developed by Robert Hall. Original program design and implementation by Derek Drummond, Steve Feder, Robert Hall and Jeff Phillips. Technical assistance provided by Michael Harris. Maintenance and enhancements by Ahmed Al-Hujazi, Derek Drummond, Anish Sharma and Warren Repole.

Getting the Most Out of PROC TABULATE was originally written by Veronica Walsamotte and Lisa Shutta. Revisions by Derek Drummond, Ahmed Al-Hujazi, Anish Sharma, and Warren Repole.

Introduction to PROC SQL was written by Robert Hall, Judith Mopsik, Jeff Phillips and Meg Seldin. Revisions by Ahmed Al-Hujazi and Warren Repole.

SAS, SAS/AF, SAS/FSP, SAS/GRAPH, SAS/QC, and SAS/STAT are registered trademarks of SAS Institute Inc., Cary, NC, USA.

OS/2 is a registered trademark of International Business Machines Corporation.

VMS is a trademark of Digital Equipment Corporation.

WordPerfect is a registered trademark of WordPerfect Corporation.
Subqueries
(from Seminars in SAS Programming — Introduction to PROC SQL)

A query expression can also be used as part of the FROM clause. The result of the query is treated as a table or data set.

Example:

```sql
PROC SQL;
TITLE "Visit date(s) with the most observations in PHARM.VISIT";
SELECT VISITDAT, FREQ LABEL='Observations'
FROM (SELECT VISITDAT, COUNT(*) AS FREQ
      FROM PHARM.VISIT
      GROUP BY VISITDAT
      HAVING FREQ = MAX(FREQ))
      AS FREQ_vis;
QUIT;
```

Visit date(s) with the most observations in PHARM.VISIT

<table>
<thead>
<tr>
<th>Date of Visit</th>
<th>Observations</th>
</tr>
</thead>
<tbody>
<tr>
<td>12MAY92</td>
<td>4</td>
</tr>
</tbody>
</table>

```sql
PROC SQL;
TITLE "State/Loan Type Combination(s) with the most observations in MORTGAGE LOANS";
SELECT STATE, LOANTYPE, COUNT(*) AS FREQ
FROM (SELECT STATE, LOANTYPE, COUNT(*) AS FREQ
      FROM MORTGAGE LOANS
      GROUP BY STATE, LOANTYPE
      HAVING FREQ = MAX(FREQ))
      AS FREQ_stl;
QUIT;
```

State/Loan Type Combination(s) with the most observations in MORTGAGE LOANS

<table>
<thead>
<tr>
<th>State</th>
<th>Loan Type</th>
<th>Observations</th>
</tr>
</thead>
<tbody>
<tr>
<td>DC</td>
<td>ADJUSTABLE</td>
<td>4</td>
</tr>
<tr>
<td>ID</td>
<td>30-yr FIXED</td>
<td>4</td>
</tr>
<tr>
<td>NO</td>
<td>ADJUSTABLE</td>
<td>4</td>
</tr>
<tr>
<td>TN</td>
<td>30-yr FIXED</td>
<td>4</td>
</tr>
</tbody>
</table>

```sql
PROC SQL;
TITLE "Jurisdiction(s) with the most observations in UTILITY CUSTOMER";
SELECT AREA, COUNT(*) AS FREQ
FROM (SELECT AREA, COUNT(*) AS FREQ
      FROM UTILITY CUSTOMER
      GROUP BY AREA
      HAVING FREQ = MAX(FREQ))
      AS FREQ_jur;
QUIT;
```

Jurisdiction(s) with the most observations in UTILITY CUSTOMER

<table>
<thead>
<tr>
<th>Jurisdiction</th>
<th>Observations</th>
</tr>
</thead>
<tbody>
<tr>
<td>MARYLAND</td>
<td>6</td>
</tr>
<tr>
<td>D.C.</td>
<td>6</td>
</tr>
</tbody>
</table>
Using Label Modifiers in the TABLE Statement
(from Seminars in SAS Programming — Getting the Most Out of PROC TABULATE)

Label modifiers can be used in the TABLE statement to alter the label displayed for a variable. To specify a label modifier, follow the variable name in the table specification with an equal sign (=) and a string enclosed in quotes.

Since the label modifier only modifies a single occurrence of a variable in a TABLE statement, it allows the use of different labels for a variable that appears multiple times in the TABULATE procedure output, including the universal class variable ALL.

Example:

```
PROC TABULATE DATA=MORTGAGE LOANS FORMAT=DOLLAR10. ;
  CLASS HOMETYPE LOANTYPE;
  VAR LOAN AMT;
  TABLE HOMETYPE=" ALL="All Home Types", LOANTYPE; 
  KEYLABEL SUM=" "; 
  TITLE "Total Loan Amounts for Each Combination of Home Type and Loan Type";
RUN;
```

Total Loan Amounts for Each Combination of Home Type and Loan Type

<table>
<thead>
<tr>
<th></th>
<th>All Loan Types</th>
<th>30-YR</th>
<th>ADJUSTABLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONDOMINIUM</td>
<td>$358,300</td>
<td>$103,900</td>
<td>$254,400</td>
</tr>
<tr>
<td>SINGLE FAMILY</td>
<td>$4,489,500</td>
<td>$2,833,200</td>
<td>$1,656,300</td>
</tr>
<tr>
<td>TOWNHOUSE</td>
<td>$1,364,600</td>
<td>$706,700</td>
<td>$657,900</td>
</tr>
<tr>
<td>All Home Types</td>
<td>$6,212,400</td>
<td>$3,643,800</td>
<td>$2,568,600</td>
</tr>
</tbody>
</table>

```
PROC TABULATE DATA=UTILITY CUSTOMER FORMAT=12. ;
  CLASS AREA TYPE;
  VAR METER93;
  TABLE AREA=" ALL="All Jurisdictions", CUSTOMER TYPE= "METERS93=" " *(ALL="All Customer Types" TYPE) ; 
  KEYLABEL SUM=" "; 
  TITLE "Number of Meters in each jurisdiction for each type of customer";
RUN;
```

Number of Meters in each jurisdiction for each type of customer

<table>
<thead>
<tr>
<th></th>
<th>All Customer Types</th>
<th>Customer Type</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>COMMERICAL</td>
<td>MULTI-FAMILY</td>
</tr>
<tr>
<td>VA - ARLINGTON</td>
<td>3</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>VA - ALEXANDRIA</td>
<td>6</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>MARYLAND</td>
<td>8</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>D.C.</td>
<td>8</td>
<td>.</td>
<td>1</td>
</tr>
<tr>
<td>All Jurisdictions</td>
<td>25</td>
<td>3</td>
<td>6</td>
</tr>
</tbody>
</table>