SESUG Speaker Sharing Program

To arrange for a SESUG speaker, contact Marje Fecht at Marje.Fecht@prowerk.com

List of Presentation Titles

- Advanced DATA Step Techniques
- Assessing SAS Skill Level during the Interviewing Process
- AutoTrader.com: From Chevette to Corvette
- Best Practices for Automated Testing and Real Time Notification in SAS Applications
- Creating Meaningful Data from Web Logs using Base SAS^R
- Dictionary Tables: Essential Tools for Serious Applications
- Don't be a Slave to your SAS Programs
- Elementary Statistics Using Base SAS
- Exceptional Exception Reports
- Fun with Functions
- Getting Data into SAS$^®$: INFILE and INPUT
- Here's the Data, Here's the Report I Want - How Do I Get There?
- How SAS Thinks
    - *OR*   Why the DATA Step Does What It Does
    - *OR*   Anatomy of a DATA Step
    - *OR*   DATA Step Essentials
- How to Incorporate Old SAS$^®$ Data into a New DATA Step, or "What is S-M-U?"
- 'LAG with a WHERE' and other DATA Step Stories
- Proc Format, a Speedy Alternative to Sort/Merge
- Program Comprehension: A Strategy for the Bewildered
- Pruning the SASLOG – Digging into the Roots of NOTEs, WARNINGs, and ERRORs
- Real Time Decision Support: Creating a Flexible Architecture for Real Time Analytics
- Rules for Tools - The SAS Utility Primer
- SAS® Consultants: A Sheep in Wolf's Clothing
- SAS Macros are the Cure for Quality Control Pains
- Summarizing Data with Base SAS^R PROCs
- Testing and Validating SAS Programs
    - *OR*   Beyond Debugging: Program Validation
    - *OR*   Programming on Purpose
- The Elements of SAS Programming Style
- The SAS Debugging Primer
- The Ultimate Match Merge: Hiring the Best SAS Programmers
- The Utter Simplicity of the TABULATE Procedure
- The Utter Simplicity of the TABULATE Procedure – A Sequel
- The Utter "Simplicity?" of the TABULATE Procedure – The Final Chapter?
- The Workhorses of Customized Report Writing - PROC TABULATE and PROC REPORT
- Version 9 Programming Enhancements
- Writing the "Best" Program: The How and When of Efficient Programming
- XML and SAS: An Advanced Tutorial

Last Update: 16Oct2004

List of Abstracts

Advanced DATA Step Techniques
Neil Howard

Understanding the intricacies of the DATA step can make all the difference in your SAS programming. This tutorial focuses on the more advanced techniques that capitalize on the power of the DATA step and working with (and around) the default actions. Topics include:

- ➢ compile versus execution time activities;
- ➢ organizing your data to maximize execution;
- ➢ data defaults, data conversions;
- ➢ missing values, formatting values;
- ➢ ordering variables;
- ➢ functions for:
- ▪ editing data,
- ▪ assigning values,
- ▪ shortening expressions
- ▪ performing table lookup;
- ➢ data management;
- ➢ effectively creating SAS data sets;
- ➢ the logic of the MERGE;
- ➢ efficiency techniques.

'Real world' examples are presented to illustrate each topic.

Assessing SAS Skill Level during the Interviewing Process
Jenine Eason

This paper will provide guidelines and tools that will assist in interviewing a candidate for a SAS programming position, even for the non-programming interviewer. Skill level categories will be defined: Beginner, Intermediate, and Advanced. Specific interviewing approaches, suggested topics, questions and tests will be explored. We'll also cover tactics to get the right candidates to apply in the first place.

Last Update: 16Oct2004

AutoTrader.com: From Chevette to Corvette
Jenine Eason

AutoTrader.com has been extremely successful in applying SAS to manage and report voluminous business information since April 1999. Since receiving the Enterprise Computing Award at SUGI 25, there have been many overwhelming successes as well as a few bumps in the road.

This paper will review the challenges and solutions that the development team at AutoTrader.com dealt with while successfully rolling out their project. For example, understanding the dot.com reporting pitfalls – what appeared to be straightforward reporting was revealed to be treacherous and vague. Forecasting storage capacities became a never-ending challenge. Creating timely reporting for the user seemed easily deliverable – but as the site grew and our audience became more sophisticated, needs changed...quickly.

AutoTrader.com successfully rose to each new level. This presentation will layout the map for collecting data from multiple sources so that our clients had one reporting source. It will explore discoveries using advanced SAS tools, Java drill down graphs, sophisticated MDDBs, ODS, dynamic Internet front ends, and HTML. An increasing demand for advanced site usability and a more professional look prompted us to delve deep into these advanced SAS tools.

Best Practices for Automated Testing and Real Time Notification in SAS Applications
Greg Nelson

Data management is one of the cornerstones of SAS as a language. SAS programs that access, manage, analyze and report on data are often taken from vast libraries of tools that are used over and over again for consistency and desirable for their reuse in similar projects. Over time, the number of potential uses of any one program or macro is challenged by the amount of time it takes to test, retest and validate these programs. As these programs become part of the production eco-system in a development environment, it is important they their testability, robustness and manageability become "built-in" to the software development process.
This paper outlines as a specific approach to building in that process to each and every program to monitor the conditions SAS programs encounter and proactively test for and announce any validation issues. We will explore the concept of automated tests through assertions, events and their attributes, event status management, and automatic notification of events to interested parties. These concepts are presented from the perspective of the SAS programmer and the systems analyst.

Creating Meaningful Data from Web Logs using Base SAS[R]
Jenine Eason

Analyzing web logs can bring very valuable information to your company. Not only can data collected from web logs give insight to management about what is happening on a web site, it also provides exact data for trending that sales can use to increase profitability. Without using a pre-packaged tool, base SAS can be utilized to read and evaluate web logs. The greatest challenge is dissecting a web log and parsing its values. This paper will bring understanding to web logs by identifying its different pieces and how best to collect them. We'll explore some of the basic reporting metrics utilizing transformed web logs.

Last Update: 16Oct2004

Dictionary Tables: Essential Tools for Serious Applications
Frank Dilorio

Dictionary tables and views are some of the most useful and least heralded tools in the SAS System. They contain a wealth of information about SAS datasets, catalogs, system options, and external files. While much of this information is available via cumbersome procedure output datasets, dictionary tables and views (which we will simply refer to as "the tables") put this information at the disposal of the application developer in a straightforward manner. The use of dictionary tables in combination with the macro language and PROC SQL, two much more utilized SAS products, provides a powerful set of tools for the SAS programmer.

This paper: explains the organization of the tables, describes their content, and presents several short applications demonstrating how the tables can be used in combination with the macro language and PROC SQL to create more efficient and dynamic applications. The reader should come away from the discussion with an understanding of the benefits of the dictionary tables as well as some ideas for how they may be used in their programming environments.

Don't be a Slave to your SAS Programs
Marje Fecht

Do you spend a lot of time updating your programs to adjust inclusion criteria? Do your programs run forever while you patiently wait? This beginning tutorial focuses on maintenance - free, efficient coding techniques so that you can spend your work time being more productive!

Topics include:
- macro coding techniques
- efficient programming tips
- code reduction tricks
- maintenance-free programming suggestions.

Elementary Statistics Using Base SAS
Deborah Babcock Buck

Base SAS includes a number of procedures that will allow you to perform elementary statistical analyses. Most of these PROCs produce descriptive statistics, but there are some capabilities for inferential statistics (hypothesis testing) as well. This paper will discuss which Base SAS PROCs will provide you with the desired statistics and show examples of code and output for a research study.

Last Update: 16Oct2004

Exceptional Exception Reports

Gary McQuown

Exception Reports document how well your data meets your pre-defined business rules. Knowing what you have in terms of data quality is essential to understanding the usability of the data and the validity of any processes performed with that data. Typically an exception report presents statistics describing nonconforming data, lists the non-conforming data or both. This information may then be used to guide to clean the data, manage how the data is used or to explain anomalies in downstream processes. While the simplest exception reports only call attention to anomalies in the data, exceptional exception reports provide detailed documentation and specific information with little additional effort from the user.

This paper presents an overview of exception reports for data quality control and describes how they can be assembled from pieces and parts of fairly common SAS® code. Topics include exceptional exception reports vs. simple exception reports, the reasons to use exception reports as frequently as possible, the application of logic in the reports and issues that may cause problems. An example of the general use exception report can be downloaded and used as is or modified to meet your requirements.

Fun with Functions

Marje Fecht

Functions can be fun (and useful) if you know when, how, and why to use them. Functions can be frustrating if you don't know the tricks behind them.

In this presentation, we will explore the most commonly used functions to streamline data processing and reduce your programming effort. You will learn how to use summary functions, string functions, conversion functions, and date functions. We will also explore some of the 60 functions added to Version 9 SAS software.

This presentation will also help you understand WHERE to use functions, since they aren't "just for the DATA Step".

Getting Data into SAS®: INFILE and INPUT

Andrew T. Kuligowski

The SAS System has numerous capabilities to store, analyze, report, and present data. However, those features are useless unless that data is stored in, or can be accessed by, the SAS System. This presentation includes an introduction to the INPUT and INFILE statements, which combine to provide a simple, yet powerful, method to pass data into the SAS System. Matters to be addressed are reading fixed and variable length files, .CSV files, in-stream data, informats, and more.

Last Update: 16Oct2004

Here's the Data, Here's the Report I Want - How Do I Get There?
Deborah Babcock Buck

SAS[R] programmers are often given a set of data and asked to generate a specific report. This can be a daunting task, especially for a relatively new SAS user. Where to start? This process involves two separate, but related phases. Phase one concerns the data itself. What form are the data in - is it raw data, an existing SAS data set or sets, or data from another software program? Does the data need to be manipulated, restructured or summarized to provide the information needed for the SAS procedure? Phase two involves determining which PROC will provide the output in the desired form. Knowing whether the report needs a specific layout, customized information, or statistics is essential in deciding on the most appropriate PROC. This paper will present some guidelines to follow for producing the desired report in a logical and organized manner using Base SAS.

How SAS Thinks *OR*
Why the DATA Step Does What It Does *OR*
Anatomy of a DATA Step *OR*
DATA Step Essentials
Neil Howard

The DATA step is the most powerful tool in the SAS System. Understanding the internals of DATA step processing, what is happening and why, is crucial in controlling your programs. This tutorial provides answers to the DATA step mysteries: what is the Program Data Vector (PDV), what are automatic SAS variables and how are they used, what is the SAS Supervisor, why you need to understand the internals of DATA step processing, what happens at program compile time, what's actually happening at execution time, how the DATA step functions as a read/write loop, what is the logic behind a MERGE, how are variable attributes captured and stored, and more. By understanding DATA step processing, you can debug your programs and interpret your results with confidence.

How to Incorporate Old SAS[®] Data into a New DATA Step, or "What is S-M-U?"
Andrew T. Kuligowski

S-M-U. Some people will see these three letters and immediately think of the abbreviation for a private university and associated football team in Texas. Others might treat them as a three-letter word, and recall a whimsical cartoon character created by Al Capp many years ago. However, in the world of the SAS[®] user, these three letters represent the building blocks for processing SAS datasets through the SAS DATA step. S, M, and U are first letters in the words SET, MERGE, and UPDATE – the 3 commands used to introduce SAS data into a DATA step.

This presentation will discuss the syntax for the SET, MERGE, and UPDATE commands. It will compare and contrast these 3 commands. Finally, it will provide appropriate uses for each command, along with basic examples that will illustrate the main points of the presentation.

Last Update: 16Oct2004

'LAG with a WHERE' and other DATA Step Stories
Neil Howard

This tutorial illustrates the importance of understanding the internals of DATA step processing, particularly when you invoke mysterious functions like LAG, code RETAIN statements, rely on WHERE constructs, and start combining various features of the DATA step language. This anthology includes such spellbinding stories as "LAG with a WHERE", "When RETAIN Doesn't Retain", "To LAG or to LEAD", "Don't Order My Variables Around", "A DIFferent LAG", and "The Case of the Missing Values".

Proc Format, a Speedy Alternative to Sort/Merge
Jenine Eason

Many users of SAS system software, especially those working with large data sets, are often confronted with several challenges. How can one reduce the data set size and reduce the amount of time required retrieving Specific data? In this paper, we attempt to do both using a matching method utilizing Proc Format to replace the CPU heavy Sort/Merge. It is ideal for situations when a key from one file is needed to extract data from another file. It is more apparently useful when at least one of the files is quite large. This method has been proved time and again to decrease CPU by 70 - 80%. This paper is intended for the intermediate to advanced SAS user. It is effective on all platforms and users of Base SAS.

Program Comprehension: A Strategy for the Bewildered
Frank Dilorio

Here, unfortunately, is a not uncommon workplace scenario. A neophyte SAS programmer is assigned to maintain, debug, or enhance an application. The atmosphere is sink or swim, the system is complex, the code is sophisticated, the documentation is scant, and the programmer is bewildered. Questions slowly take shape. "What, exactly, am I supposed to do?" "What part(s) of the application need my attention?" "Will changing program X affect program Y?" And, most critically, "Where do I start?"

What the programmer needs is a strategy for understanding the program, then finding its "sweet spots" as efficiently as possible.

This paper presents a generalized approach for programmers, particularly SAS "newbies", to develop an understanding of how applications work. It also shows how to translate this comprehension into effective coding. The paper identifies and discusses the rationale for questions the programmer should ask about: task definition, program-level code, supporting code, system design and specification documents, and required domain knowledge.

Attendees should come away with a better understanding of how to frame the programming problem and effectively gather the resources needed to obtain a solution. They will also come to believe that the coding of, say, a DATA step is usually simple, but the real art of programming is learning what to code, and why.

Pruning the SASLOG – Digging into the Roots of NOTEs, WARNINGs, and ERRORs
Andrew T. Kuligowski

You've sat through constant design meetings. You've endured countless requests for "just one more little change". You even managed to find a creative solution to that nagging technical problem. But, you persevered, and despite all of the obstacles, you've managed to eliminate the final syntax error in your newest SAS routine. Time to sit back and relax -- uh, not quite ...

The primary focus of this presentation will be on techniques to ensure comprehension of your input data. We will look at several messages that are often found in the SASLOG, such as:

NOTE: MERGE statement has more than one data set with repeats of BY values.

that imply that there may be gaps in your knowledge of your data! Special emphasis will be placed on the use of ad-hoc queries to assist in finding data anomalies that can cause problems with your SAS code. It is assumed that the reader has a basic understanding of the SASLOG, including its composition, format, and the SAS system options which control its content.

Real Time Decision Support: Creating a Flexible Architecture for Real Time Analytics
(Note: appropriate as a Keynote/ Plenary Session presentation)
Greg Nelson

Leaders have focused their entire careers on their ability to gather, assess, evaluate and assimilate data to effectively drive change. The deployment of enterprise systems and strategic initiatives to support customer intimacy and organizational preparedness has often led to the development of data warehousing and business intelligence applications that optimize the data paths between those who know and those who should know. The end result of much of this effort is a complete infrastructure designed to move data through the enterprise.   Drip feeds, wipe and load, "slowing changing" dimension management, swim-lanes, parallelization, data optimization – all technical details that obscure the fact that data is still 12 hours old. This presentation focuses on the things that we can do today to make data movement happen so that decisions can be made with better quality, in near real time.   In addition, attention will be paid to when we should drive for real-time decision support and when it might not be appropriate. Finally, we will discuss a framework that supports low cost, incremental improvements in your information architecture at the same time optimizing the business processes to ensure information transparency across the enterprise.

Last Update: 16Oct2004

Rules for Tools - The SAS Utility Primer
Frank Dilorio

Let's start with the premise that good programmers are lazy by nature. They want to use tools such as formats and ODS for execution-time efficiency or to pretty-up our output, functions to perform calculations, and so on. Another hallmark of a good programmer is a keen eye for pattern recognition. Rather than rewrite basically the same program over and over, they identify similarities and parameterize the program, making it into a general-purpose program, a "utility."

This paper steps through the life cycle of a simple utility. It starts with "naïve" code that doesn't exploit program similarities, then illustrates how a general-purpose utility may be developed. It ends with the initial program becoming a call to a simple, powerful routine in a macro library. The transition from simple, brute-force programming into a compact, general- purpose utility isn't a random event. The last sections of the paper present a set of design principles for utilities.

Although we focus on Base SAS in Version 9.0, the principles and techniques are readily extended across SAS versions and products. The reader will come away from this paper with an appreciation of both the process and the tool set required to build generalized programs.

SAS Macros are the Cure for Quality Control Pains
Gary McQuown

The purpose of this presentation is three fold. The first is to introduce the reader to the concept of Data Quality Control and it's value to any IT project or process. This process is a frequent topic at "high levels" but seldom discussed with those who do the actual programming where it will do the most good.

Secondly it is intended to encourage the use of SAS macros when and where they are most appropriate. Macros lend themselves to tasks that are repetitious and well defined. They are ideal for quality control work because (if designed and run correctly) they greatly reduce the
introduction of human error the resources required. As we all know, documentation, error checking, and data cleaning all essential tasks, but they are all to often delayed until the very end or delegated to those with the least understanding of the process.

Lastly it is intended to direct the reader to the wealth of SAS Macro related code available on the web. The quantity and quality of SAS code placed in public domain is quite impressive and continues to grow. When supplemented with the ongoing exchanges on SAS-L, the volume of "free" assistance and material is truly amazing.

Last Update: 16Oct2004

SAS® Consultants: A Sheep in Wolf's Clothing
Gary McQuown

Thinking of hanging out your shingle and going it alone? Look before you leap and make sure you know both the risks as well as the rewards of being a SAS Consultant. Although employees and consultants may write identical code; their behavior, responsibilities, risks and rewards can be very different. This article will explore the fundamental differences between SAS Consultants (independent contractors or 1099's) and SAS Programmers (employees or W2's), and address the most common questions dealing with self employment and the transition process.

Working as a SAS Consultant carries certain risks as well as rewards. Anyone contemplating such a career should carefully and objectively consider all of his or her options.   If he or she is determined to make the transition, it is extremely important that they understand and carefully plan the process.

Summarizing Data with Base SAS^R PROCs
Deborah Babcock Buck

Base SAS provides a number of procedures designed to aid the SAS user in developing data summary reports. These procedures include MEANS, UNIVARIATE, FREQ, REPORT and TABULATE. Additionally, PROC FORMAT is also available to modify the appearance of data values or combine values into desired categories within PROCs without changing actual data values or creating new variables.

This presentation explores various common types of summary reports and what factors you should consider in deciding which procedure is best suited to your reporting needs. It is not meant to be a detailed tutorial on PROC step programming code for all of these procedures, but rather a guide on how to decide which procedures fits the requirements for a given report.

Summary reports generated using Base SAS PROCs are presented along with the code producing these reports.

Testing and Validating SAS Programs *OR*
Beyond Debugging: Program Validation *OR*
Programming on Purpose
Neil Howard

"Act in haste and repent at leisure; code too soon, and debug forever." Raymond Kennington

Overheard at an interview for a SAS programming position: "But you don't have to <u>test</u> SAS programs!!!" As the interviewers quickly escort the confused candidate out the door, they recall how often it is assumed that a fourth generation language "does so much for you" that you don't have to test the code. The SAS system is easy to use, and the learning curve to productivity is relatively short. But SAS is just as easy to ABUSE. Programmers and analysts must not lose sight of the indisputable facts: data is seldom clean, logic is too often faulty, and fingers walk clumsily over keyboards. Condition codes and clean logs are not accurate indicators of successful programs.

Good debuggers make good programmers. And good analysts and problem-solvers make good programmers. And just because a SAS® program is free of errors, warnings, notes, and bugs does not guarantee that the program is doing what it is supposed to do. This paper addresses the process of debugging, testing and validating SAS programs and the tools available in the SAS language to facilitate the process. It covers techniques for ensuring that the logic and intent of the program is correct, that the requirements and design specifications are met, and that data errors are detected. Since as much as 80% of a programmer's time is invested in testing and validation, it's important to develop a personal approach to testing and a routine testing methodology in your group.

The Elements of SAS Programming Style
Frank Dilorio

The generalized nature of SAS software almost guarantees that "n" users will develop "n" unique solutions to even basic tasks. The gap between the task correctly performed by the programs and the disparate code is, for the most part, due to programming style. This presentation discusses a set of generalized programming style guidelines useful to both experienced and novice SAS programmers.

It first investigates general principles of program design, those aspects of the analysis and coding process common to all aspects of SAS programming. The next sections focus on coding guidelines for the DATA step and procedures. Finally, debugging techniques are addressed.

The presentation contends that "good" programming style usually results in programs which are more effective in terms of both human and machine resources. The intent is not to pronounce one style good and another lacking, but to simply outline an experienced user's guidelines and gently prod other users to examine their programming habits. These habits will become critical to the success of organizations as SAS software becomes embedded in more environments and organizations.

The SAS Debugging Primer
Frank Dilorio

Meet an accomplished SAS programmer and you meet someone who's probably learned by making (and fixing) lots of mistakes along the way. The breadth of the SAS System's target applications, the variety of its "dialects" (Base SAS, macro, SCL, IML, SQL), and the quirky procedural/non-procedural environmental mix conspire to make mastery of the SAS System a slippery slope to ascend. Debugging is the art of gracefully recovering and learning from falls during the ascent.

This paper discusses techniques for debugging SAS programs. Its purpose is two-fold. First, it provides behavioral and technical tips for fixing code (how to read error messages in the SAS Log, knowing when there is a problem with the program even if SAS says there isn't, using the DATA step debugger, identifying system options, using PROCs for data validation, using macro variables to control debugging output, etc.) The second focus of the paper is its presentation of design and coding methods that make the programming process more reliable, thus reducing the need for debugging in the first place.

The paper's target audience is relative newcomers to the SAS System. More seasoned users may find or rediscover some of the techniques and features being discussed. Emphasis is placed on Base SAS and the macro language, although the techniques themselves are applicable to SCL and other products.

The Ultimate Match Merge: Hiring the Best SAS Programmers
Neil Howard

'The closest to perfection a person ever becomes is when he fills out a job application form.' (Stanley J. Randall)

You are interviewing a candidate in a sharp navy suit, with an impressive resume and ten years of SAS experience.   But, how do you tell if this ten years of experience is really ten years, or the same year over and over again? If, indeed, past behavior or past job performance is the best indicator of future performance, how is the SAS community probing for the facts? What are our assessment criteria?

The author surveyed select SAS programmers, analysts, trainers, human resource experts, and managers as a reality check into our performance as interviewers and our experience as interviewees. What techniques are being used to interview and assess SAS talent?

This paper explores the results of this survey, looking at the following interview techniques: identifying resume red flags, conducting useful phone interviews, structuring general face-to-face interviews, "measuring" intangibles, proficiency testing, and code walk-throughs. The paper includes a collection of technical SAS interview questions gathered from past SUGI papers, SAS users and SAS-L contributors; and discusses the development of probing technical questions, metrics, and standards.

The Utter Simplicity of the TABULATE Procedure
Dan Bruns

When I first started using SAS in a university environment in 1972, I was very excited about how much SAS could help me. As I learned more and more and moved to the real world of the employed I needed more flexibility in my reporting – PROC PRINT had some and PROC MEANS and FREQ had basically none. And SUMMARY needed too much DATA step manipulation. What I needed was a marriage of all these with the flexibility to control what I wanted and where and how! Thank the Gods for PROC TABULATE!! I was able to produce reports from massive amounts of data in practically any way I needed! The power of being able to simply rearrange a few variable names and change the complete look of the report was great.

The Utter Simplicity of the TABULATE Procedure – A Sequel
Dan Bruns

Here we are again TABULATE fans, but this time we are going to take a look at some of the more advanced (even obscure) features of the TABULATE procedure. In this tutorial we will take a closer look at the ALL variable (?), the PROC and TABLE statement options, subtotaling, wild formatting, and even a BRIEF introduction to the percentages mystery.

The Utter "Simplicity?" of the TABULATE Procedure – The Final Chapter?
Dan Bruns

Well, here we are again TABULATE fans. I believe I have exhausted this topic (to DEATH some folks say), so I thought I would put it to rest in this FINAL CHAPTER with a paper on the truly advanced features of the TABULATE procedure. The problem is these advanced features are anything but simple. In this tutorial we look at some simpler advanced features, like FORMCHAR, column and row titling, and formatting, and then the one that is really a bear to understand – percentages (PCTN and PCTSUM). Some of the new Version 8 features will also be covered.

The Workhorses of Customized Report Writing - PROC TABULATE and PROC REPORT
Deborah Babcock Buck

SAS[R] has a number of procedures that produce summary reports. These can include reports developed for assistance in data cleansing, for descriptive statistics, or for statistical analyses (hypothesis testing, modeling, prediction, etc.) Two of the Base SAS procedures that are very versatile in generating summary reports are PROC TABULATE and PROC REPORT.

This paper attempts to introduce you to some of the capabilities of both of these procedures and to provide some guidance on which is the more appropriate for a given need. An example of the output from each of these procedures and the code that produced it are included. The code presented in this paper is valid for SAS Versions 6-9.

Last Update: 16Oct2004

Version 9 Programming Enhancements
Marje Fecht

Performance improvements are the well-publicized enhancement to Version 9, but what else has changed that impacts your "SAS Programs"? This Hands-On Workshop explores many new Version 9 features including:

- new functions that may eliminate the need for complex expressions
- changes and additions to informats and formats
- interface additions and improvements
- new export capabilities
- procedure enhancements
- and more. . .

This workshop is beneficial for Version 8 SAS users who are getting started with Version 9.

Writing the "Best" Program: The How and When of Efficient Programming
Frank Dilorio

It's relatively easy to write programs that optimize the use of CPU and other machine resources. There is a large and continually growing body of literature on the subject. What isn't as straightforward is knowing **when** to employ the techniques - blind implementation of tuning techniques is often not required by the task at hand and can sometimes even be counterproductive.

This paper addresses both the "how to" and "when to" aspects of writing efficient programs. It describes design and coding techniques that conserve hardware resource usage. It also identifies other, non-machine implications of their usage that could dissuade the programmer from their use. For example, using temporary array elements is more efficient than using named elements but has the documented-but-obscure behavior of retaining values across observations. Maintenance of such code by other than "seasoned" and up to date programmers can be unexpectedly problematic.

The concept of efficiency used in the paper includes all aspects of the program life cycle. We apply the "how and when" question to system design issues, system startup, DATA steps, procedures, and macros. Emphasis is on Base SAS software. The reader should finish the paper comfortable with the idea that the "best" program is not always the one that minimizes hardware resources.

XML and SAS: An Advanced Tutorial
Greg Nelson

One of the goals for SAS applications developers has been to develop three-tier and n-tier applications where the application logic (business rules) is separate from the data, which, in turn, is isolated from the user interface. In a previous paper (Barnes Nelson, 1999) we discussed how to implement this logic separation using the SAS Component Language. This paper extends that line of thinking by introducing SAS developers to XML. eXtensible Markup Language, or XML, is a protocol of sorts that can be described as a technique for separating data from its presentation. In this paper, we will discuss XML in the context of SAS applications and how it can be used in the preparation and presentation of data. We will explore some of the features of XML that makes it a good partner for SAS-based applications.