

Mastering the SAS[®] DS2 Procedure

Advanced Data-Wrangling Techniques

Second Edition



SAS University Edition

Mark Jordan



From *Mastering the SAS DS2 Procedure*.
Full book available for purchase [here](#).

Contents

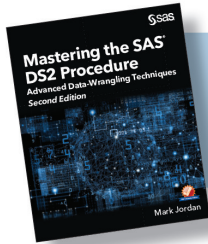
Dedication	iii
Foreword	ix
About This Book	xi
About the Author	xv
Chapter 1: Getting Started	1
1.1 Introduction.....	1
1.1.1 What is DS2?	1
1.1.2 Traditional SAS DATA Step versus DS2	2
1.1.3 What to Expect from This Book.....	6
1.1.4 Prerequisite Knowledge	7
1.2 Accessing SAS and Setting Up for Practice	7
Chapter 2: Introduction to the DS2 Language	9
2.1 Introduction.....	9
2.2 DS2 Programming Basics.....	9
2.2.1 General Considerations.....	9
2.2.2 Program Structure	11
2.2.3 Procedure Options and Global Statements	12
2.2.4 Program Blocks.....	18
2.2.5 Methods	19
2.2.6 User-Defined Methods	21
2.2.7 Variable Identifiers and Scope.....	25
2.2.8 Data Program Execution	30
2.3 Converting a SAS DATA Step to a DS2 Data Program	31
2.3.1 A Traditional SAS DATA Step	31
2.3.2 Considerations	31
2.3.3 The Equivalent DS2 Data Program.....	32
2.3.4 More Complex Data Program Processing	32
2.3.5 Automatic Conversion with PROC DSTODS2.....	33
2.4 Review of Key Concepts.....	35

Chapter 3: DS2 Data Program Details	37
3.1 Introduction	37
3.2 DS2 Data Programs versus Base SAS DATA Steps	37
3.2.1 General Considerations	37
3.2.2 The Seven Subtle Dissimilarities.....	40
3.2.3 DS2 “Missing” Features	55
3.3 Data Types in DS2.....	61
3.3.1 DS2 and ANSI Data Types	61
3.3.2 Automatic Data Type Conversion.....	62
3.3.3 Non-coercible Data Types	64
3.3.4 Processing SAS Missing and ANSI Null Values.....	66
3.4 Review of Key Concepts	71
Chapter 4: User-Defined Methods and Packages	73
4.1 Introduction	73
4.2 Diving into User-Defined Methods.....	74
4.2.1 Overview.....	74
4.2.2 Designing a User-Defined Method	74
4.3 User-Defined Packages.....	79
4.3.1 General Considerations	79
4.3.2 User-Defined Package Specifics	79
4.4 Object-Oriented Programming with DS2 Packages.....	84
4.4.1 General Considerations	84
4.4.2 Designing an Object.....	84
4.4.3 Using Objects as Building Blocks.....	93
4.4 Review of Key Concepts	96
Chapter 5: Predefined Packages	99
5.1 Introduction	99
5.2 Executing FCMP Functions in DS2	100
5.2.1 The FCMP Package.....	100
5.2.2 FCMP Package Example	101
5.3 The Hash and Hiter (Hash Iterator) Packages	106
5.3.1 General	106
5.3.2 Hash Package Example	106
5.3.3 Hash Iterator Package Example	109
5.4 The HTTP and JSON Packages	111
5.4.1 General	111

5.4.2 HTTP Package Specifics	112
5.4.3 JSON Package Specifics.....	116
5.4.4 HTTP and JSON Packages Example.....	120
5.5 The Matrix Package	124
5.5.1 General.....	124
5.5.2 Matrix Package Example.....	127
5.6 The PCRXFIND and PCRXREPLACE Packages	129
5.6.1 General.....	129
5.6.2 PCRXFIND and PCRXREPLACE Package Examples	131
5.7 The SQLSTMT Package	134
5.7.1 General.....	134
5.7.2 SQLSTMT Package Example	136
5.8 The TZ (Time Zone) Package	142
5.8.1 General.....	142
5.8.2 TZ Package Example.....	143
5.9 Review of Key Concepts.....	144
Chapter 6: Parallel Processing in DS2	145
6.1 Introduction.....	145
6.2 Understanding Threaded Processing.....	146
6.2.1 The Need for Speed	146
6.2.2 Loading Data to and from RAM.....	146
6.2.3 Manipulating Data in RAM	146
6.3 DS2 Thread Programs.....	147
6.3.1 Writing DS2 Thread Programs.....	147
6.3.2 Parallel Processing Data with DS2 Threads.....	149
6.4 DS2 and the SAS In-Database Code Accelerator	152
6.4.1 DS2 Program In-Database Processing.....	152
6.5 DS2 and SAS® Viya® and SAS Cloud Analytic Services (CAS)	153
6.5.1 A Brief Introduction to SAS Viya and CAS.....	153
6.5.2 Running DS2 Programs in CAS	153
6.6 Review of Key Concepts.....	160
Chapter 7: Performance Tuning in DS2	161
7.1 Introduction.....	161
7.2 DS2_OPTIONS Statement.....	162
7.2.1 TRACE Option	162
7.3 Analyzing Performance with the SAS Log	163
7.3.1 Obtaining Performance Statistics	163

7.3.2 Analyzing Performance Statistics.....	164
7.3.3 Tuning Your Code.....	165
7.4 Learning and Troubleshooting Resources	165
7.4.1 SAS Learning Resources	165
7.4.2 SAS Support Communities	166
7.4.3 SAS Technical Support	166
7.5 Review of Key Concepts	167
7.6 Connecting with the Author	167
Index	169

From *Mastering the SAS® DS2 Procedure: Advanced Data-Wrangling Techniques, Second Edition* by Mark Jordan.
Copyright © 2018, SAS Institute Inc., Cary, North Carolina, USA. ALL RIGHTS RESERVED.



From *Mastering the SAS DS2 Procedure*.
Full book available for purchase [here](#).

Chapter 1: Getting Started

1.1 Introduction.....	1
1.1.1 What is DS2?	1
1.1.2 Traditional SAS DATA Step versus DS2.....	2
1.1.3 What to Expect from This Book.....	6
1.1.4 Prerequisite Knowledge	7
1.2 Accessing SAS and Setting Up for Practice	7

1.1 Introduction

Today’s data scientists deal with ever larger data sets from a widening variety of data sources, and the computations required to process that data are continually becoming more complex. As SAS has been modernized with each new release, most SAS procedures (PROCs) have been rewritten to be thread-enabled, allowing them to use multiple CPUs on a single computer or even to push processing into massively parallel processing (MPP) computing environments such as Teradata, Hadoop, or the SAS High-Performance Analytics grid. But the DATA step, with its sequential, observation-by-observation approach to data manipulation, has remained stubbornly single threaded.

In the summer of 2013, SAS released SAS 9.4, which included a revolutionary new programming language named DS2. Each subsequent maintenance release of SAS 9.4 has added new features and functionality in the DS2 language. The second edition of this book is intended to expand coverage of the DS2 language to include several of these new features, and to provide more in-depth coverage on features that have proven to be extraordinarily useful over the last couple of years.

1.1.1 What is DS2?

DS2 is basically DATA step programming redesigned from the ground up with several important goals. I like to think of DS2 as a language that combines the power and control of the Base SAS DATA step programming language with the simplicity of SQL and throws in just enough object-oriented features to make simple, reusable code modules a reality. With DS2 you can perform extremely complex data manipulation and transformation by writing intuitive, succinct, and compact programs. It always amazes me how much you can accomplish with just a little code!

Here is what DS2 can do:

- natively process American National Standards Institute (ANSI) SQL data types for better integration with external data stores
- provide modern structured programming constructs, making it simple to extend the functionality of the DS2 language with reusable code modules

- tightly integrate with SQL
- provide simple, safe syntax for multi-threaded processing to accelerate CPU-intensive tasks

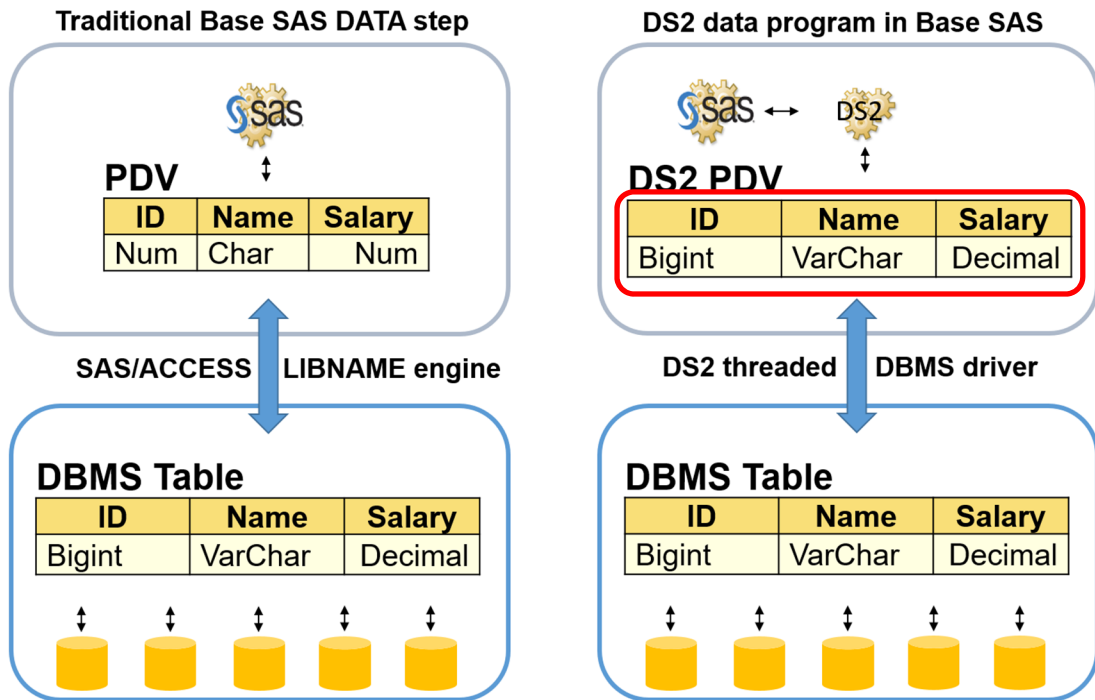
2017 brought SAS 9.4 maintenance release 5 (SAS 9.4M5), and the DS2 language had further matured, incorporating new features and functionality. Here are a few of the features that made me decide to write a second edition of this book:

- PROC DSTODS2—a new procedure to help you convert existing DATA step programs to DS2 data programs.
- Addition of some useful global statements to PROC DS2.
- Support for the OF keyword when using variable array references as variable lists.
- Better CONNECT string documentation, and simpler syntax to limit which libraries DS2 connects to upon invocation.
- The new MERGE statement (which is very different from a DATA step MERGE...).
- The new PCRX packages, which allow using regular expressions to process text with better performance than when using PRX functions.
- A new LIBNAME engine for JSON files that makes reading JSON easier in traditional DATA step programs than using the JSON package in DS2, but is not as flexible.

1.1.2 Traditional SAS DATA Step versus DS2

If you have a SAS/ACCESS license for a supported database management system (DBMS), the traditional SAS DATA step can process DBMS data, but native data types are first translated to SAS 8-byte floating-point numeric or fixed-width character data types by the LIBNAME engine. This causes a loss of precision when dealing with higher-precision ANSI numeric data types, such as BIGINT or DECIMAL. DS2 is capable of directly manipulating ANSI data types—including multi-byte character types—even when processing on the SAS compute platform. Figure 1.1 compares and contrasts traditional Base SAS DATA step processing with DS2 data program processing, illustrated by a basic example.

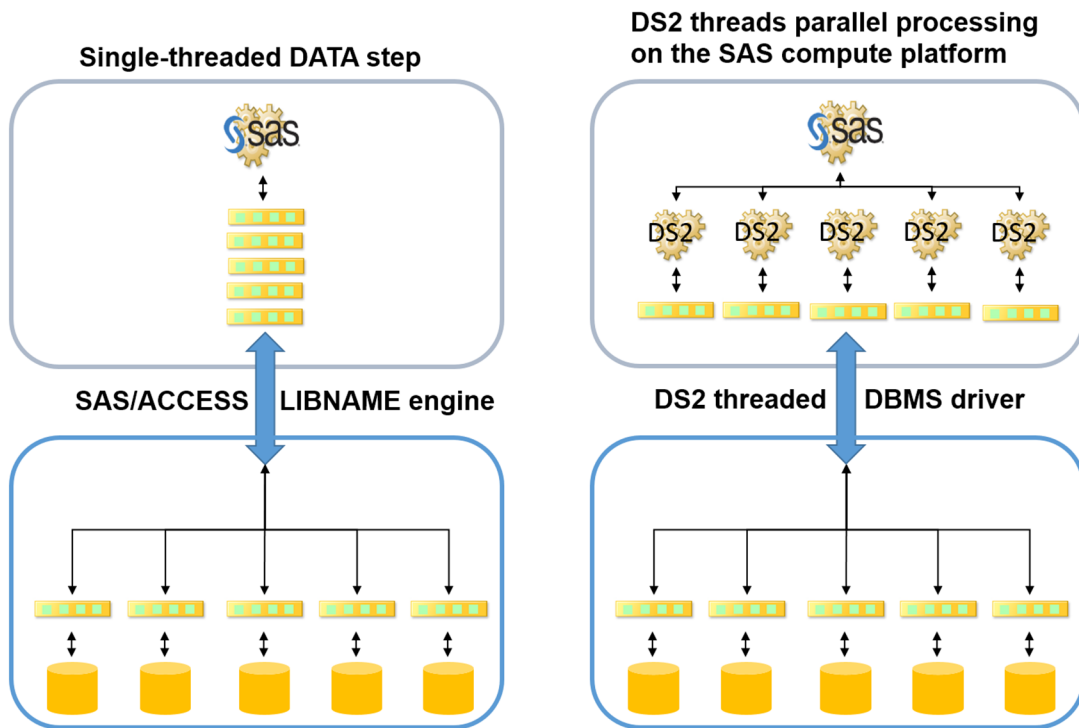
Figure 1.1: Traditional DATA Step Processing versus DS2 Data Program Processing



As you can see in Figure 1.1, when the traditional SAS DATA step accesses DBMS data via the SAS/ACCESS engine using a LIBNAME statement, DBMS data types are automatically converted to fixed-width character or double-precision, floating-point numeric. In contrast, the DS2 data program accesses the RDBMS data via a special driver that is associated with SAS/ACCESS software and can therefore process the data in its native data type.

The SAS DATA step is essentially a data-driven loop: reading, manipulating, and writing out one observation at a time. If the process is computationally complex, it can easily become CPU bound; that is, data can be read into memory faster than it can be processed. If the DATA step elapsed (clock) time in the SAS log is about the same as the CPU time, your process is most likely CPU bound. DS2 can accelerate CPU-bound processing by processing data rows in parallel using DS2 threads. Figure 1.2 contrasts traditional SAS DATA step, single-threaded processing with multi-threaded processing using DS2 thread and data programs.

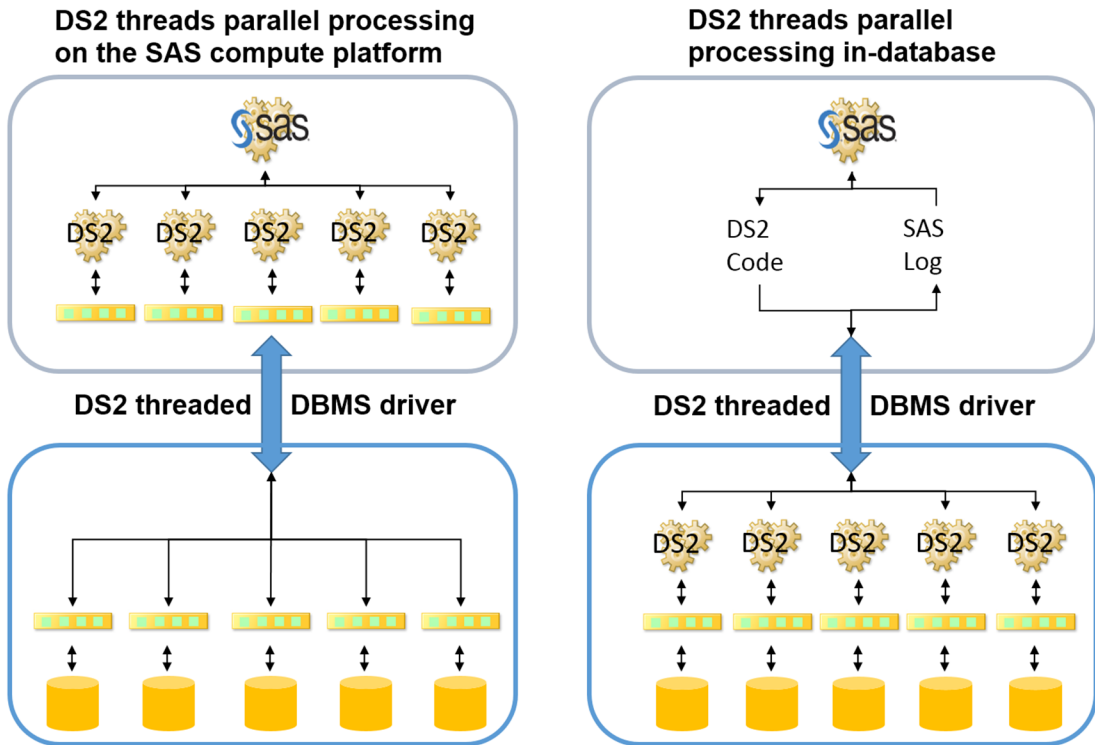
Figure 1.2: Serial Processing in the DATA Step versus DS2 Parallel Processing



As you can see in Figure 1.2, the traditional SAS DATA step must process each row of data sequentially. By contrast, DS2 can use thread and data programs to process multiple rows of data simultaneously.

Notice that both processes use a single read thread, so if the bottleneck is getting data from disk or the DBMS into memory on the SAS compute platform for processing, then threaded processing on the SAS compute platform will not improve overall performance. This situation is referred to as an “I/O bound” operation. DS2 uses a single read thread to feed multiple compute threads when processing in Base SAS to ensure that each row of data is distributed to only one compute thread for processing. Similarly, if I/O is the bottleneck and computations are taking place on the SAS platform, DS2 is unlikely to improve performance.

However, because today’s DBMS data is enormous, data movement should be completely avoided whenever possible. With DS2, in a properly provisioned and configured SAS installation that includes the SAS In-Database Code Accelerator, your DS2 programs can actually execute on the database hardware in the SAS Embedded Process without having to move data to the SAS compute platform at all. Figure 1.3 compares DS2 data program threaded processing on the SAS compute platform to in-database processing with DS2 and the SAS In-Database Code Accelerator.

Figure 1.3: Parallel Processing with Threads: SAS Compute Platform versus In-Database

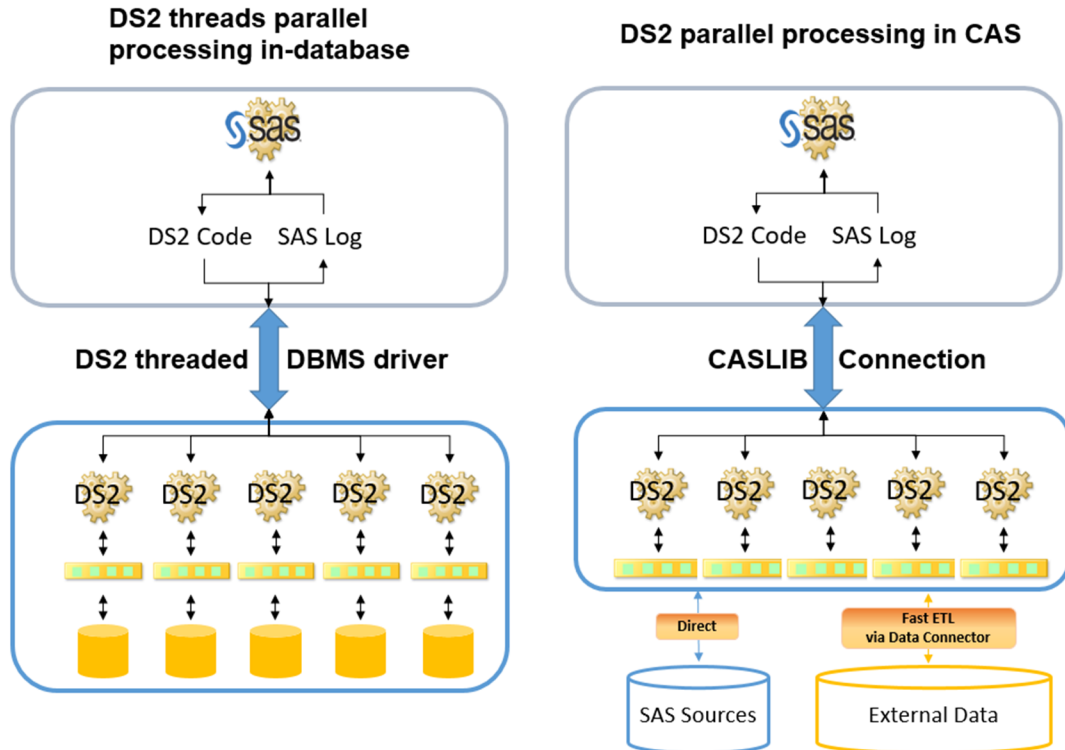
As you can see in Figure 1.3, using DS2 thread and data programs with the SAS In-Database Code Accelerator enables the DS2 code to compile and execute on the massively parallel DBMS hardware. If the process reads from a DBMS table and also writes to a DBMS table, then only the code goes into the DBMS, and only the SAS log comes out. All processing takes place in the DBMS. This concept of “taking the code to the data” instead of the traditional “bringing the data to the code” greatly reduces the amount of data movement that is required for processing. It also extends the computational capabilities of the DBMS to include SAS functions and processing logic, and takes full advantage of the massively parallel processing (MPP) capabilities of the DBMS. If you are a SAS programmer or a data scientist in an environment that includes SAS, you will find that DS2 quickly becomes a must-have tool for data manipulation.

This edition wouldn’t be complete without a discussion of the new SAS® Viya® architecture with its SAS Cloud Analytic Services (CAS). CAS distributed processing works a lot like in-database processing, with two significant advantages:

1. Enormous data sets can be persisted in memory. This means that, once loaded, subsequent passes on the data can be made without having to reload the data from off-line storage. If the data is too large to load into memory all at once, you don’t need to modify your program—CAS handles the complexity behind the scenes to maximize throughput.
2. When in-database processing with the SAS In-Database Code Accelerator, only data that resides in the database is eligible for distributed processing on the database hardware. CAS can access data from a wide variety of sources through native direct access or external source data connectors.

DS2 running in CAS enables in-database-style parallel processing, but with the ability to source data from a variety of data stores, while the in-memory capabilities minimize physical I/O operations, as shown in Figure 1.4.

Figure 1.4: Parallel Processing with Threads: In-Database versus CAS



1.1.3 What to Expect from This Book

Data wrangling, as I use the term, is more than just cleaning up data to prepare it for analysis. A data wrangler acquires data from diverse sources, then structures, organizes, and combines it in unique ways to facilitate analysis and obtain new insights. This book teaches you to wrangle data using DS2, highlighting the similarities and differences between DS2 data programs and traditional DATA step processing, as well as leveraging DS2's parallel-processing power to boost your data-wrangling speed.

Here is what you will be able to do after you finish reading this book:

- identify the types of processes for which the language was designed and understand the conditions indicating that DS2 is a good choice when attempting to improve the performance of existing DATA step processes
- identify which programming statements and functions are shared between DATA step and DS2 data programs
- identify the DATA step functionality not available in DS2 and understand why it was not included in the DS2 language
- know what new DS2 program functionality is not available in the traditional DATA step

- directly manipulate ANSI data types in a DS2 program
- understand the implications of handling data that contains both SAS missing and ANSI null values in the same process
- convert a Base SAS data manipulation process from DATA step to a DS2 data program
- understand the DS2 system methods and how they relate to traditional DATA step programming constructs
- create custom DS2 methods, extending the functionality of the DS2 language
- store custom DS2 methods in packages and reuse them in subsequent DS2 programs
- use DS2 packages to create object-oriented programs
- use predefined DS2 packages to add extra functionality to your DS2 data programs
- create DS2 thread programs and execute them from a DS2 data program for parallel processing of data records
- use BY-group and *FIRST.variable* and *LAST.variable* processing in a DS2 data program or thread to perform custom data summarizations, without requiring a presort of the data
- determine whether your system has the capability to execute DS2 programs in-database and, if so, execute your DS2 thread programs in parallel, fully distributed on an MPP DBMS platform

1.1.4 Prerequisite Knowledge

This book was written with the seasoned Base SAS programmer in mind. You can acquire the prerequisite knowledge from other SAS Press books, such as *An Introduction to SAS University Edition* by Ron Cody or *The Little SAS Book: A Primer* by Lora Delwiche and Susan Slaughter.

Before diving in, you'll want to be familiar with the following key concepts:

- DATA step programming, in general
- SAS libraries
- accessing data with a LIBNAME statement
- reading and writing SAS data sets
- the role of the program data vector (PDV) in DATA step processing
- conditional processing techniques
- arrays
- iterative processing (DO loops)
- macro processing, in general
- assigning values to macro variables
- resolving macro variables in SAS code
- timing of macro process execution versus execution of other SAS code
- SQL joins

1.2 Accessing SAS and Setting Up for Practice

If you do not currently have access to SAS software, you can use the robust learning community online known as SAS Analytics U. From the SAS Analytics U website, you can download a free, up-to-date, and fully functional copy of SAS University Edition, which is provided as a virtual machine (VM). The SAS University Edition VM includes a completely installed, configured, well-provisioned SAS server.

The examples in this book were all created and executed using SAS University Edition, with the exception of the sections requiring DBMS access. You can get your own free copy of SAS University Edition at http://go.sas.com/free_sas.

Getting Ready to Practice

1. Download the ZIP file containing the data for this book from <http://support.sas.com/jordan>.
2. Unzip the files to a location available to SAS. If you are using SAS University Edition, the shared folder you designated when setting up your SAS environment is a good location for these files.
3. In SAS, open the program **_setup.sas**, follow the directions in the program comments to modify the code for your SAS environment, and then submit the program. You will need to run this program only once.

You are now ready to run the sample programs that are included with this book.

If you exit SAS between study sessions, it is easy to return. When you start SAS again, just run the program named **libnames.sas** in order to re-establish your connection to the appropriate SAS libraries before working with the other programs from this book. As an aside, if you have difficulty re-establishing your SAS library connections with the **libnames.sas** program, there is no harm in rerunning **_setup.sas**. It just takes a little longer to run.

About This Book

Purpose

This book will take you from complete novice to confident competence with the new SAS programming language, DS2.

Is This Book for You?

Are you comfortable with traditional Base SAS DATA step and SQL processing? Want to supercharge your data preparation? Then DS2, the new SAS programming language that integrates the power and control of the DATA step with the ease, flexibility, and rich data type palette of SQL might be just what you are looking for! DS2 plays well with multiple data sources, and is exceptionally well suited for manipulating data on massively parallel platforms (MPPs), including SAS Viya's SAS Cloud Analytic Services (CAS), Hadoop, Greenplum, and Teradata.

DS2 gives you the power to accomplish tasks such as these:

- process traditional SAS data sets and DBMS data tables containing full-precision ANSI data types, all in the same DS2 data program.
- easily create and share reusable code modules.
- create custom objects and use object-oriented programming techniques.
- safely and simply process multiple rows of data in parallel using Base SAS.
- manipulate data with DS2 programs while fully leveraging the massively parallel processing capabilities of CAS.
- Execute DS2 programs in-database on Hadoop, Greenplum, and Teradata.

Prerequisites

This book assumes reader proficiency in these topics:

- DATA step programming:
 - SAS libraries
 - accessing data with a LIBNAME statement
 - reading and writing SAS data sets
 - the role of the program data vector (PDV) in DATA step processing
 - conditional processing techniques
 - arrays

- iterative processing (DO loops)
- macro processing:
 - assigning values to macro variables
 - resolving macro variables in SAS code
 - timing of macro process execution versus execution of other SAS code
- SQL joins

Scope of This Book

This book covers the following topics:

- the types of processes for which the language was designed and the conditions indicating when DS2 would be a good choice for improving process performance
- programming statements and functions shared between traditional DATA steps and DS2 data programs
- DATA step functionality that is not available in DS2
- new DS2 functionality that is not available in the traditional DATA step
- manipulating ANSI data types at full precision in DS2
- converting traditional Base SAS DATA steps to DS2 data programs
- creating reusable code modules in DS2 and packaging them for easy reuse
- using DS2 packages to create object-oriented programs
- creating DS2 thread programs for parallel processing of data records
- executing DS2 thread programs distributed on MPP DBMS platforms

About the Examples

Software Used to Develop the Book's Content

SAS 9.4M5

The majority of the programs in this book use only Base SAS and were developed with SAS University Edition.

- The programs that read and write to DBMS also required these solutions:
 - SAS/ACCESS Interface to Teradata
 - SAS In-Database Code Accelerator for Teradata

All DBMS data was stored in Teradata Express 15.0.

The primary references used in writing this book include the following SAS 9.4 documentation:

- [SAS® 9.4 DS2 Language Reference](#)
- [SAS® 9.4 DATA Step Statements: Reference](#)

- [SAS® 9.4 Functions and CALL Routines: Reference](#)
- [SAS® 9.4 FedSQL Language Reference](#)
- [DS2 Programming: Essentials course notes](#)

Example Code and Data

To access this book's example code and data, go to the author page at <http://support.sas.com/publishing/authors>. Select the name of the author. Then, select "Example Code and Data" from the list of links.

If you are unable to access the code through the website, send email to saspress@sas.com.

SAS University Edition



If you are using SAS University Edition to access data and run your programs, check the SAS University Edition page to ensure that the software contains the product or products that you need to run the code.

Most of the programs in this book require only Base SAS to execute and will run just fine in SAS University Edition. However, a few programs read from or write to Teradata tables. These programs cannot be executed in SAS University Edition, because the SAS/ACCESS Interface to Teradata is not included. A list of the SAS software products that are included with SAS University Edition is available at <http://support.sas.com/software/products/university-edition/index.html>.

Output and Graphics Used in This Book

All output and SAS graphics in this book were produced by executing the associated program code from SAS Studio on SAS 9.4M5.

Additional Help

Although this book illustrates many analyses regularly performed in businesses across industries, questions specific to your aims and issues may arise. To fully support you, SAS and SAS Press offer you the following resources:

- For questions about topics covered in this book, contact the author through SAS Press:
 - Send questions by email to saspress@sas.com; include the book title in your correspondence.
 - Submit feedback on the author's page at http://support.sas.com/author_feedback.
- For questions about topics beyond the scope of this book, post queries to the relevant SAS Support Communities at <https://communities.sas.com/welcome>. The Base SAS programming community is the appropriate community for questions about DS2.
- SAS maintains a comprehensive website with up-to-date information. One page that is particularly useful to both the novice and to the seasoned SAS user is the Knowledge Base. Search for relevant notes in the "Samples and SAS Notes" section of the Knowledge Base at <http://support.sas.com/resources>.
- Registered SAS users or their organizations can access SAS Customer Support at <http://support.sas.com>. Here you can pose specific questions to SAS Customer Support; under

Support, click *Submit a Problem*. You must provide an email address to which replies can be sent, identify your organization, and provide a customer site number or license information. This information can be found in your SAS logs.

Keep in Touch

We look forward to hearing from you. We invite questions, comments, and concerns. If you want to contact us about a specific book, please include the book title in your correspondence.

Contact the Author through SAS Press

- By email: saspress@sas.com
- Via the web: http://support.sas.com/author_feedback

Purchase SAS Books

- Visit sas.com/store/books.
- Phone 1-800-727-0025
- Email sasbook@sas.com

Subscribe to the SAS Learning Report

Receive up-to-date information about SAS training, certification, and publications via email by subscribing to the SAS Learning Report monthly eNewsletter. Read the archives and subscribe today at <http://support.sas.com/community/newsletters/training!>

Publish with SAS

SAS is recruiting authors! Are you interested in writing a book? Visit <http://support.sas.com/saspress> for more information.

About The Author



Mark Jordan (a.k.a. “SAS Jedi”) grew up in northeast Brazil as the son of Baptist missionaries. He served 20 years as a US Navy submariner, pursuing his passion for programming as a hobby. Upon retiring from the Navy, he turned his hobby into a dream job, working as a SAS programmer for nine years in manufacturing and financial services before coming to SAS in 2003. As a Principal Technical Training Consultant for SAS, Mark has written and teaches a broad spectrum of Foundation SAS programming classes.

Mark currently lives in Toano, VA with his wife, Lori, and their cat, the amazing Tiger Man. When he isn't teaching, Mark sporadically posts “Jedi SAS Tricks” on the SAS Learning Post blog and enjoys playing with his grandchildren and great-grandchildren, hanging out at the beach, and reading science fiction novels. His secret obsession is flying toys—kites, rockets, drones—and though he tries to convince his wife that they are for the grandkids, she isn't buying it.

You can find his blog at <http://sasjedi.tips> and you can follow him on Twitter @SASJedi. Learn more about this author by visiting his author page at <http://support.sas.com/jordan>. There you can download free book excerpts, access example code and data, read the latest reviews, get updates, and more.

Index

A

- ANSI data types 61–62
- ANSI null values, processing 66–71
- ANSI SQL quoting standards 47–49
- ANSIMODE option 70–71
- APPEND procedure 139
- ARRAY statement 58–60
- assignment operator (:=) 59
- ATTRIB statement 58
- automatic data type conversion 33–35, 36, 62–64

B

- benchmarking 167
- BIGINT data type 62, 63
- BINARY data type 62, 65
- BINDRESULTS method 136
- BY statement 35

C

- CALL routines 101
- CAS (SAS Cloud Analytic Services) 5–6, 153–160
- CHAR data type 61, 68–69
- Cody, Ron
 - An Introduction to SAS University Edition* 7
- ColumnNumber parameter 117
- COMMENT statement 60
- CONTENTS method 80, 82, 96
- CPU bound 146, 160
- CPU Time statistic 164
- CREATEGETMETHOD method 114
- CREATEPARSER method 116
- cross-library table updates 137–141
- cursor controls (@) 45–46

D

- data
 - loading to and from RAM 146
 - manipulating in RAM 146–147
 - overwriting 42–43
- data parameter 107

- Data Program block 11
- data programs
 - about 18
 - versus* base SAS DATA steps 37–60
 - executing 30–31
- DATA statement 11–12, 35, 42–43
- data types
 - See also specific data types*
 - automatic conversion 62–64
 - in DS2 61–71
 - non-coercible 64–66
- data wrangling 6
- database management system (DBMS) 2–7
- DATA_NULL_program 137
- datasrc parameter 107
- DATE data type 62, 65, 142
- DBMS (database management system) 2–7
- DECIMAL data type 62, 63, 70–71
- declarative statements 38
- DECLARE PACKAGE statement 84
- DECLARE statement 26, 36, 58
- declaring
 - with an SQLSTMT package 134–136
 - a matrix 124–125
 - PCRXFIND package 130
 - PCRXREPLACE package 130–131
 - using TZ package 142
 - variables 43–45
- DELETE method 60, 79, 84
- Delwiche, Lora
 - The Little SAS Book: A Primer* 7
- DIVBYZERO=ERROR|IGNORE option 16
- DO loop 38, 74, 140
- DO UNTIL loop 118, 136
- DO WHILE loop 136
- DOUBLE data type 61, 63, 68–69, 71
- DROP PACKAGE statement 15
- DROP statement 27, 38
- DROP THREAD statement 16

DS2

See also specific topics

about 1–2

ANSI mode 70–71

converting SAS DATA step to 31–35

versus DATA step 10

data types 61–62

versus SAS DATA step 2–6

SAS mode 68–69

DS2 procedure 11–17, 43–44, 70–71

DS2COND= option 26

DS2_OPTIONS statement 16–17, 162–163, 167

DS2_OPTIONS TYPEWARN statement 62, 64

DSTODS2 procedure, automatic conversion with 33–35, 36

duplicate parameter 107–108

E

END statement 12, 20, 140

ENDDATA statement 11–12, 18, 35

ENDPACKAGE statement 11–12, 35

END=SET statement 21

ENDTHREAD statement 11–12, 19, 35

ERROR option 44

executable statements 12, 35, 40–41

EXECUTE method 135

EXECUTEMETHOD method 114

F

FCMP package 100–105, 144

FCMP procedure 10, 100–101, 144

FedSQL procedure 49, 57–58, 134

FETCH method 136

FILENAME URL statement 111

FIND method 109

FIRST method 109

FLOAT data type 61

FORMAT statement 58

FORWARD statement 93

FREQ procedure 68, 118–119

FULLTIMER 163–164, 167

functions

creating custom with FCMP procedure 101–102

INPUT 63

MISSING 69, 70–71

NULL 69

TO_DATE 65–66, 142

TO_DOUBLE 66

TO_TIME 65–66, 142

TO_TIMESTAMP 65–66, 142

fundamentals

about 37, 71

data programs *versus* base SAS DATA steps 37–60

DS2 data types 61–71

G

general considerations for programming 9–10

GET method 112

GETNEXTTOKEN method 116–118

GETRESPONSEBPDYASSTRING method 115

global statements 12–17

GOTO statement 38

H

hash objects

looking up values in data sets with 109–111

using as lookup tables 109

HASH package 106–109, 144

hashepx parameter 107

HAVING clause 58

HEAD method 112

HITER package 100, 109–111, 144

HTTP package 100, 111–124, 144

I

IN method 125

INFORMAT statement 58

INIT method 19, 20, 31, 35, 79, 108, 110, 147

IN_OUT parameters 22–25, 35–36, 74–76, 96

INPUT function 63

input/output bound 146

instantiating

with an SQLSTMT package 134–136

a matrix 124–125

PCRXFIND package 130

PCRXREPLACE package 130–131

using TZ package 142

INTEGER data type 62, 63

INTEREST method 82–83

interest methods 75–78

An Introduction to SAS University Edition (Cody) 7

Involuntary Context Switches statistic 164

IS methods 117

J

JSON package 100, 111–124, 144

JSON procedure 111, 116

K

KEEP statement 27, 38

keys parameter 106

keywords, reserved 46–47
Kiva API 111–115, 120–124

L

LABEL statement 58
LAST method 109
learning resources 165–166
LENGTH statement 58
LIBNAME statement 2–3, 61, 111, 136
LIBS= option 13–15
line feeds (/) 45–46
LineNumber parameter 117
LINK statement 60
The Little SAS Book: A Primer (Delwiche and Slaughter) 7
loading a matrix 124–125
Logger package 100
%LOGPARSE macro 164
lookup tables, using hash objects as 109

M

massively parallel processing (MPP) capabilities 5
Matrix package 100, 124–129
MEANS procedure 67–68
memory location 22
Memory statistic 164
MERGE statement 35, 49–55
METHOD block 36
method code block, executable statements and 40–41
METHOD statement 12, 19
methods
 See also specific methods
 overloading 78–79
 programming 19–21
“missing” features
 about 55, 71
 ARRAY statement 58–60
 ATTRIB statement 58
 COMMENT statement 60
 DELETE statement 60
 FORMAT statement 58
 INFORMAT statement 58
 LABEL statement 58
 LENGTH statement 58
 LINK statement 60
 MODIFY statement 58
 UPDATE statement 58
 WHERE statement 56–58
MISSING function 69, 70–71
missing values, processing 66–71
MISSING_NOTE option 16
MODIFY statement 58

MPP (massively parallel processing) capabilities 5, 152
multidata parameter 108

N

name prefix variable 39
name variable list 38
NCHAR data type 61
NEXT method 109
NOLIBS option 13–15
non-coercible data types 64–66
NONE option 44
NOTE option 44
NULL function 69
numbered range variable list 39
NVARCHAR data type 61

O

object-oriented programming, with DS2 packages 84–96
objects, using as building blocks 93–96
ODS (Output Delivery System) 18
online training 165–166
options
 ANSIMODE 70–71
 DIVBYZERO=ERROR|IGNORE 16
 DS2 procedure 12–17
 DS2COND= 26
 ERROR 44
 LIBS= 13–15
 MISSING_NOTE 16
 NOLIBS 13–15
 NONE 44
 NOTE 44
 OVERWRITE=YES 42–43
 SAS 16
 SCOND 26, 43–44
 SCOND=ERROR|WARNING|NOTE|NONE 17
 THREADS= 148, 151, 153, 165
 TRACE 17, 162–163, 167
 TYPEWARN 17
 WARNING 43–45

ordered parameter 107
ordered view 106
Output Delivery System (ODS) 18
OUTPUT statement 35, 38, 137, 140
OVERWRITE=YES option 42–43
overwriting data 42–43

P

Package Program block 11
package programs 18–19
PACKAGE statement 11–12, 18, 35, 79–81

packages

- FCMP 100–105, 144
- HASH 106–109, 144
- HITER 100, 109–111, 144
- HTTP 100, 111–124, 144
- JSON 100, 111–124, 144
- Logger 100
- Matrix 100, 124–129
- PCRXFIND 100, 129–134, 144
- PCRXREPLACE 100, 129–134, 144
- SQLSTMT 100, 134–141, 144
- storing user-defined methods in 79–81
- TZ 100, 142–143, 144

Page Swaps statistic 164

parallel processing

- about 145, 160
- SAS Cloud Analytic Services (CAS) 153–160
- SAS In-Database Code Accelerator 152–153
- SAS Viya 153–160
- Thread programs 147–151
- threaded processing 146–147

ParseFlags parameter 117

parsing API response using JSON package 121–122

PCRXFIND package 100, 129–134, 144

PCRXREPLACE package 100, 129–134, 144

PDV (program data vector) 10, 126–127

PDV contamination 30

performance tuning

- about 161, 167
- analyzing with SAS log 163–165
- DS2_OPTIONS statement 162–163
- learning resources 165–166
- troubleshooting 165–166

POST method 112, 116

predefined packages

- about 99–100, 144
- executing FCMP functions 100–105
- FCMP 100–105, 144
- HASH 106–109, 144
- HITER 100, 109–111, 144
- HTTP 100, 111–124, 144
- JSON 100, 111–124, 144
- Matrix 100, 124–129
- PCRXFIND package 100, 129–134, 144
- PCRXREPLACE package 100, 129–134, 144
- SQLSTMT 100, 134–141, 144
- TZ 100, 142–143, 144

PREV method 109

procedures

- APPEND 139
- DS2 11–17, 43–44, 70–71

DSTODS2 33–35, 36

FCMP 10, 100–101, 144

FedSQL 49, 57–58, 134

FREQ 68, 118–119

JSON 111, 116

MEANS 67–68

SQL 49, 57–58

processing

ANSI null values 66–71

SAS missing values 66–71

threaded 146–147

program blocks 18–19

program data vector (PDV) 10, 126–127

program flow control statements 38

programming

data program execution 30–31

general considerations 9–10

methods 19–21

program blocks 18–19

structure of 11–12

system methods 19–21

user-defined methods 21–25

variable identifiers and scope 25–30

PUT statement 45–46, 63, 71, 115

Q

quality assurance (QA) testing 91

querying Kiva API using HTML package 120–121

question mark (?) 113, 134

QUIT statement 11

quotation marks 47

R

RAID (redundant array of independent disk) 146

RAM

loading data to and from 146

manipulating data in 146–147

RC parameter 117

REAL data type 61

Real Time statistic 164

redundant array of independent disk (RAID) 146

reserved keywords 46–47

RETAIN statement 38

RETURN statement 38

RUN method 19, 20, 27, 28, 29, 31, 35, 79, 140, 147

RUN statement 16

S

SAS, accessing 7–8

SAS 9.4 FedSQL language Reference 58

SAS Blogs (website) 165

SAS Cloud Analytic Services (CAS) 5–6, 153–160

- SAS DATA step
 - about 2–6
 - converting to DS2 data program 31–35
 - versus data programs 37–60
 - versus DS2 10
 - traditional 31
 - SAS DS2 Language Reference* 164
 - SAS In-Database Code Accelerator 4–5, 152–153
 - SAS log, analyzing performance with 163–165
 - SAS Logging: Configuration and Programming Reference* 164
 - SAS National Language Support (NLS): Reference Guide* 142
 - SAS option 16
 - SAS Support Communities (website) 166
 - SAS Technical Support (website) 164, 166–167
 - SAS University Edition (website) 8
 - SAS Viya 153–160
 - SCOND option 26, 43–44
 - SCOND=ERROR|WARNING|NOTE|NONE option 17
 - scope, variable identifiers and 25–30
 - SET FROM statement 19, 147, 148–149, 165
 - SET statement 26, 38, 43–45, 56, 71, 140
 - SETPARMS method 147, 148
 - SETPARSERINPUT method 116
 - setup 7–8
 - single quotation marks 47
 - Slaughter, Susan
 - The Little SAS Book: A Primer* 7
 - SMALLINT data type 62
 - solid state disk (SSD) 146
 - SQL procedure 49, 57–58
 - SQL query 10
 - SQLSTMT package 100, 134–141, 144
 - SSD (solid state disk) 146
 - statements
 - BY 35
 - ARRAY 58–60
 - ATTRIB 58
 - COMMENT 60
 - DATA 11–12, 35, 42–43
 - declarative 12, 38
 - DECLARE 26, 36, 58
 - DECLARE PACKAGE 84
 - DELETE 60
 - DROP 27, 38
 - DROP PACKAGE 15
 - DROP THREAD 16
 - DS2_OPTIONS 16–17, 162–163, 167
 - DS2_OPTIONS TYPEWARN 62, 64
 - END 12, 20, 140
 - ENDDATA 11–12, 18, 35
 - ENDPACKAGE 11–12, 35
 - END=SET 21
 - ENDTHREAD 11–12, 19, 35
 - executable 12, 35, 40–41
 - FILENAME URL 111
 - FORMAT 58
 - FORWARD 93
 - global 12–17
 - GOTO 38
 - INFORMAT 58
 - KEEP 27, 38
 - LABEL 58
 - LENGTH 58
 - LIBNAME 3, 136
 - LINK 60
 - MERGE statement 35, 49–55
 - METHOD 12, 19
 - MODIFY 58
 - OUTPUT 35, 38, 137, 140
 - PACKAGE 11–12, 18, 35, 79–81
 - program flow control 38
 - PUT 45–46, 63, 71, 115
 - QUIT 11
 - RETAIN 38
 - RETURN 38
 - RUN 16
 - SET 26, 38, 43–45, 56, 71, 140
 - SET FROM 19, 147, 148–149, 165
 - STOP 38
 - THREAD 11–12, 19, 35
 - UPDATE 58, 137
 - VARARRAY 58, 125, 126–127
 - WHERE 56–58
 - STOP statement 38
 - structure, of programming 11–12
 - suminc parameter 108
 - support communities 166
 - System CPU Time statistic 164
 - system methods 19–21, 96
- T**
- technical support 166–167
 - TERM method 19, 21, 27, 29, 31, 35, 79, 128, 147
 - thread, defined 147
 - Thread Program block 11
 - Thread programs 19, 147–151
 - THREAD statement 11–12, 19, 35
 - threaded processing 146–147
 - THREADS= option 148, 151, 153, 165
 - TIME data type 62, 65, 142
 - Time Zone package
 - See* TZ package

- TIMESTAMP data type 62, 65, 142
- TINYINT data type 62
- TOARRAY method 126–127
- TO_DATE function 65–66, 142
- TO_DOUBLE function 66
- Token parameter 117
- TokenType parameter 117
- TO_TIME function 65–66, 142
- TO_TIMESTAMP function 65–66, 142
- TOVARARRAY method 126
- TRACE option 17, 162–163, 167
- TRANS method 126–127
- troubleshooting 64, 165–166
- %TSLIT macro 48–49, 71
- type variable 39
- TYPEWARN option 17
- TZ package 100, 142–143, 144

U

- UPDATE statement 58, 137
- User CPU Time statistic 164
- user-defined methods
 - about 21–25, 73–74, 96
 - designing 74–79
 - packages 79–83

V

- values
 - manipulating from a matrix 126–127
 - retrieving from a matrix 126–127
- VARARRAY statement 58, 125, 126–127
- VARBINARY data type 62, 65
- VARCHAR data type 61, 69, 70–71, 120
- variable encapsulation 30
- variable identifiers, scope and 25–30
- variables, declaring 43–45

W

- WARNING option 43–45
- WHERE statement 56–58
- WRITE methods 116

Y

- YouTube channel 165

Ready to take your SAS[®] and JMP[®] skills up a notch?



Be among the first to know about new books,
special events, and exclusive discounts.

support.sas.com/newbooks

Share your expertise. Write a book with SAS.

support.sas.com/publish



sas.com/books
for additional books and resources.

sas
THE POWER TO KNOW.

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. * Indicates USA registration.
Other brand and product names are trademarks of their respective companies. © 2017 SAS Institute Inc. All rights reserved. M1588358 US.0217

