Contents

About This Book .......................................................................................................................... xv
About The Author ......................................................................................................................... xix
Acknowledgments ....................................................................................................................... xxi
Chapter 1: Designing Database Tables ..................................................................................... 1
  Introduction ................................................................................................................................. 2
  Database Design ......................................................................................................................... 2
    Conceptual View ....................................................................................................................... 2
    Table Definitions ..................................................................................................................... 3
    Redundant Information ............................................................................................................ 3
    Normalization ........................................................................................................................... 4
    Normalization Strategies ......................................................................................................... 5
  Column Names and Reserved Words ......................................................................................... 7
    ANSI SQL Reserved Words ..................................................................................................... 8
    SQL Code .................................................................................................................................. 8
  Data Integrity .............................................................................................................................. 8
    Referential Integrity .................................................................................................................. 9
  Database Tables Used in This Book ....................................................................................... 9
  CUSTOMERS Table ................................................................................................................. 9
  INVENTORY Table .................................................................................................................... 10
  INVOICE Table ....................................................................................................................... 10
  MANUFACTURERS Table ....................................................................................................... 10
  PRODUCTS Table .................................................................................................................... 11
  PURCHASES Table .................................................................................................................. 11
  Table Contents ......................................................................................................................... 12
  The Database Structure ........................................................................................................... 14
  Sample Database Tables ........................................................................................................ 14
  Summary .................................................................................................................................... 21

# Chapter 2: Working with Data in PROC SQL

**Introduction**

**Overview of Data Types**
- **Numeric Data**
- **Date and Time Column Definitions**
- **Character Data**
- **Missing Values and NULL**
- **Arithmetic and Missing Data**

**SQL Keywords**

**SQL Operators and Functions**
- **Comparison Operators**
- **Logical Operators**
- **Arithmetic Operators**
- **Character String Operators and Functions**

**Summarizing Data**

**Predicates**

**Dictionary Tables**
- **Dictionary Tables and Metadata**
- **Displaying Dictionary Table Definitions**
- **Dictionary Table Column Names**
- **Accessing a Dictionary Table's Contents**

**Summary**

# Chapter 3: Formatting Output

**Introduction**

**Formatting Output**
- **Writing a Blank Line between Each Row**
- **Displaying Row Numbers**
- **Using the FORMAT= Column Modifier to Format Output**
- **Concatenating Character Strings**
- **Inserting Text and Constants between Columns**
- **Using Scalar Expressions with Selected Columns**
- **Ordering Output by Columns**
- **Grouping Data with Summary Functions**
- **Grouping Data and Sorting**
### Contents

- Subsetting Groups with the HAVING Clause ................................................................. 110
- Formatting Output with the Output Delivery System .................................................. 112
  - ODS and Output Formats ......................................................................................... 113
  - Sending Output to a SAS Data Set ......................................................................... 114
  - Converting Output to Rich Text Format ................................................................. 115
  - Exporting Data and Output to Excel .................................................................... 116
- Delivering Results to the Web .................................................................................. 118
- Summary .................................................................................................................. 119

**Chapter 4: Coding PROC SQL Logic** ..................................................................... 121

- Introduction ............................................................................................................. 122
- Conditional Logic .................................................................................................... 122
  - SQL Code ............................................................................................................... 122
  - SQL Code ............................................................................................................... 122
  - SQL Code ............................................................................................................... 123
  - SQL Code ............................................................................................................... 123
- CASE Expressions ................................................................................................... 123
  - Simple Case Expression ....................................................................................... 124
  - Searched CASE Expression .................................................................................. 137
  - Case Logic versus COALESCE Expression .......................................................... 142
  - Assigning Labels and Grouping Data .................................................................. 143
  - Logic and Nulls ....................................................................................................... 146
- Interfacing PROC SQL with the Macro Language ................................................... 148
  - Exploring Macro Variables and Values ................................................................. 149
  - Creating Multiple Macro Variables ..................................................................... 153
  - Using Automatic Macro Variables to Control Processing .................................... 156
  - Building Macro Tools and Applications ............................................................... 158
  - Creating Simple Macro Tools ............................................................................... 158
  - Cross-Referencing Columns .................................................................................. 158
  - Determining the Number of Rows in a Table ....................................................... 159
  - Identifying Duplicate Rows in a Table .................................................................. 160
- Summary .................................................................................................................. 161

**Chapter 5: Creating, Populating, and Deleting Tables** ........................................ 163

- Introduction ............................................................................................................. 164
  - Creating Tables ..................................................................................................... 164
### Contents

- Outer Joins ................................................................................................................................. 255
  - Left Outer Joins ...................................................................................................................... 255
  - Right Outer Joins .................................................................................................................. 258
  - Full Outer Joins .................................................................................................................... 259
- Subqueries .................................................................................................................................... 261
  - Alternate Approaches to Subqueries .................................................................................. 261
  - Passing a Single Value with a Subquery ........................................................................... 262
  - Passing More Than One Row with a Subquery ................................................................. 266
  - Comparing a Set of Values ................................................................................................. 267
  - Correlated Subqueries ........................................................................................................ 269
- Set Operations ............................................................................................................................ 271
  - Rules for Set Operators ...................................................................................................... 271
  - Set Operators and Precedence .......................................................................................... 272
  - Accessing Rows from the Intersection of Two Queries .................................................. 272
  - Accessing Rows from the Combination of Two Queries ................................................ 274
  - Concatenating Rows from Two Queries ........................................................................... 276
  - Comparing Rows from Two Queries ................................................................................. 278
- Complex Query Applications ...................................................................................................... 280
  - One-to-One, One-to-Many, Many-to-One, and Many-to-Many Relationships .............. 280
  - Processing First, Last, and Between Rows for BY-and Groups ..................................... 285
  - Determining the Number of Rows in an Input Table ....................................................... 290
  - Identifying Tables with the Most Indexes ........................................................................ 291
  - Summary ............................................................................................................................... 293

### Chapter 8: Working with Views ................................................................................................. 295

- Introduction ............................................................................................................................... 296
- Views—Windows to Your Data ................................................................................................. 296
  - What Views Aren’t ................................................................................................................ 297
  - Types of Views ..................................................................................................................... 297
  - Creating Views ..................................................................................................................... 299
  - Displaying a View’s Contents ............................................................................................. 300
  - Describing View Definitions .............................................................................................. 301
  - Creating and Using Views in SAS ...................................................................................... 302
  - Views and SAS Procedures ................................................................................................. 303
  - Views and DATA Steps ....................................................................................................... 305
- Eliminating Redundancy ......................................................................................................... 307
Contents

Restricting Data Access—Security ................................................................. 307
Hiding Logic Complexities ........................................................................ 308
Nesting Views .............................................................................................. 310
Updatable Views .......................................................................................... 312
  Inserting New Rows of Data .................................................................. 313
  Updating Existing Rows of Data ............................................................. 317
  Deleting Rows of Data .......................................................................... 320
Deleting Views ............................................................................................ 321
Summary ...................................................................................................... 322

Chapter 9: Troubleshooting and Debugging .............................................. 323
Introduction ............................................................................................... 323
The World of Bugs ..................................................................................... 324
The Debugging Process ............................................................................. 324
Types of Problems ..................................................................................... 326
Troubleshooting and Debugging Techniques ............................................. 327
  Validating Queries with the VALIDATE Statement ............................... 327
  Documented PROC SQL Options and Statement .................................. 328
Undocumented PROC SQL Options ....................................................... 342
  Macro Variables .................................................................................... 343
  Troubleshooting and Debugging Examples .......................................... 345
Summary .................................................................................................... 350

Chapter 10: Tuning for Performance and Efficiency ................................. 351
Introduction ............................................................................................... 351
Understanding Performance Tuning ......................................................... 352
Sorting and Performance .......................................................................... 352
User-Specified Sorting (SORTPGM= System Options) .............................. 353
  Automatic Sorting ............................................................................... 353
Grouping and Performance ...................................................................... 354
Splitting Tables ......................................................................................... 354
Indexes and Performance ......................................................................... 354
Reviewing CONTENTS Output and System Messages .............................. 355
Optimizing WHERE Clause Processing with Indexes ............................... 358
  Constructing Efficient Logic Conditions ............................................. 359
  Avoiding UNIONs ................................................................................ 361
Summary ..................................................................................................................................... 365
Index ........................................................................................................................................ 367

Chapter 1: Designing Database Tables

Introduction .................................................................................................................... 2

Database Design ............................................................................................................. 2
   Conceptual View ........................................................................................................ 2
   Table Definitions ...................................................................................................... 3
   Redundant Information ............................................................................................ 3
   Normalization ......................................................................................................... 4
   Normalization Strategies ....................................................................................... 5

Column Names and Reserved Words .......................................................................... 7
   ANSI SQL Reserved Words .................................................................................... 8
   SQL Code .................................................................................................................. 8

Data Integrity .............................................................................................................. 8
   Referential Integrity ............................................................................................. 9

Database Tables Used in This Book .......................................................................... 9
   CUSTOMERS Table .................................................................................................. 9
   INVENTORY Table .................................................................................................. 10
   INVOICE Table ...................................................................................................... 10
   MANUFACTURERS Table ..................................................................................... 10
   PRODUCTS Table ................................................................................................... 11
   PURCHASES Table ............................................................................................... 11

Table Contents ........................................................................................................... 12
   The Database Structure ....................................................................................... 14
   Sample Database Tables ....................................................................................... 14

Summary ..................................................................................................................... 21
Introduction

The area of database design is very important in relational processes. Much has been written on this subject, including entire textbooks and thousands of technical papers. No pretenses are made about the thoroughness of this very important subject in these pages. Rather, an attempt is made to provide a quick-start introduction for those readers who are unfamiliar with the issues and techniques of basic design principles. Readers needing more information are referred to the references listed in the back of this book. As you read this chapter, the following points should be kept in mind.

Database Design

Activities related to good database design require the identification of end-user requirements and involve defining the structure of data values on a physical level. Database design begins with a conceptual view of what is needed. The next step, called logical design, consists of developing a formal description of database entities and relationships to satisfy user requirements. Seldom does a database consist of a single table. Consequently, tables of interrelated information are created to enable more complex and powerful operations on data. The final step, referred to as physical design, represents the process of achieving optimal performance and storage requirements of the logical database.

Conceptual View

The health and well-being of a database depends on its database design. A database must be in balance with all of its components (or optimized) to avoid performance and operation bottlenecks. Database design doesn’t just happen and is not a process that occurs by chance. It involves planning, modeling, creating, monitoring, and adjusting to satisfy the endless assortment of user requirements without impeding resource requirements. Of central importance to database design is the process of planning. Planning is a valuable component that, when absent, causes a database to fall prey to a host of problems including poor performance and difficulty in operation. Database design consists of three distinct phases, as illustrated in Figure 1.1.
Chapter 1: Designing Database Tables

Figure 1.1: Three Distinct Phases of Database Design

<table>
<thead>
<tr>
<th>Conceptual Database Design</th>
<th>Conceptual Design Activities</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>- Identify all entities.</td>
</tr>
<tr>
<td></td>
<td>- Define entity attributes uniqueness and usefulness.</td>
</tr>
<tr>
<td></td>
<td>- Define attribute properties including data type and size.</td>
</tr>
<tr>
<td></td>
<td>- Define entities and attributes as related to one another.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Logical Database Design</th>
<th>Logical Design Activities</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>- Transform conceptual design criteria into relational form.</td>
</tr>
<tr>
<td></td>
<td>- Transform entities into tables.</td>
</tr>
<tr>
<td></td>
<td>- Transform entity attributes into table columns.</td>
</tr>
<tr>
<td></td>
<td>- Transform tables and columns using Normalization rules.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Physical Database Design</th>
<th>Physical Design Activities</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>- Assign one or more indexes (simple and composite).</td>
</tr>
<tr>
<td></td>
<td>- Tune indexes for maximum performance.</td>
</tr>
<tr>
<td></td>
<td>- Denormalize tables, if necessary, to improve access speeds.</td>
</tr>
</tbody>
</table>

Table Definitions

PROC SQL uses a model of data that is conceptually stored as multisets rather than as physical files. A physical file consists of one or more records ordered sequentially or some other way. Programming languages such as COBOL and FORTRAN evolved to process files of this type by performing operations one record at a time. These languages were generally designed and used to mimic the way people process paper forms.

PROC SQL was designed to work with multisets of data. Multisets have no order, and members of a multiset are of the same type using a data structure known as a table. For classification purposes, a table is a base table consisting of zero or more rows and one or more columns, or a table is a virtual table (called a view), which can be used the same way that a table can be used (see Chapter 8, “Working with Views”).

Redundant Information

One of the rules of good database design requires that data not be redundant or duplicated in the same database. The rationale for this conclusion originates from the belief that if data appears more than once in a database, then there is reason to believe that one of the pieces of data is likely to be in error. Furthermore, redundancy often leads to the following:

- Inconsistencies, because errors are more likely to result when facts are repeated.
- Update anomalies where the insertion, modification, or deletion of data may result in inconsistencies.

Another thing to watch for is the appearance of too many columns containing NULL values. When this occurs, the database is probably not designed properly. To alleviate potential table design
issues, a process referred to as normalizing is performed. When properly done, this ensures the complete absence of redundant information in a table.

**Normalization**

The development of an optimal database design is an important element in the life cycle of a database. Not only is it critical for achieving maximum performance and flexibility while working with tables and data, it is essential to the organization of data by reducing or minimizing redundancy in one or more database tables. The process of table design is frequently referred to by database developers and administrators as normalization.

The normalization process is used for reducing redundancy in a database by converting complex data structures into simple data structures. It is carried out for the following reasons:

- To organize the data to save space and to eliminate any duplication or repetition of data.
- To enable simple retrieval of data to satisfy query and report requests.
- To simplify data manipulation requests such as data insertions, updates, and deletions.
- To reduce the impact associated with reorganizing or restructuring data as new application requirements arise.

The normalization process attempts to simplify the relationship between columns in a database by splitting larger multicolumn tables into two or more smaller tables containing fewer columns. The rationale for doing this is contained in a set of data design guidelines called normal forms. The guidelines provide designers with a set of rules for converting one or two large database tables containing numerous columns into a normalized database consisting of multiple tables and only those columns that should be included in each table. The normalization process consists of multiple steps with each succeeding step subscribing to the rules of the previous steps.

Normalization helps to ensure that a database does not contain redundant information in two or more of its tables. In an application, normalization prevents the destruction of data or the creation of incorrect data in a database. What this means is that information of fact is represented only once in a database, and any possibility of it appearing more than once is not, or should not be, allowed.

As database designers and analysts proceed through the normalization process, many are not satisfied unless a database design is carried out to at least third normal form (3NF). Joe Celko in his popular book *SQL for Smarties: Advanced SQL Programming* (Morgan Kaufman, 1999), describes 3NF this way: “Databases are considered to be in 3NF when a column is dependent on the key, the whole key, and nothing but the key.”

While the normalization guidelines are extremely useful, some database purists actually go to great lengths to remove any and all table redundancies even at the expense of performance. This is in direct contrast to other database experts who follow the guidelines less rigidly in an attempt to improve the performance of a database by only going as far as third normal form (or 3NF). Whatever your preference, you should keep this thought in mind as you normalize database tables. A fully normalized database often requires a greater number of joins and can adversely affect the
speed of queries. Celko mentions that the process of joining multiple tables in a fully normalized database is costly, specifically affecting processing time and computer resources.

**Normalization Strategies**

After transforming entities and attributes from the conceptual design into a logical design, the tables and columns are created. This is when a process known as normalization occurs. Normalization refers to the process of making your database tables subscribe to certain rules. Many, if not most, database designers are satisfied when third normal form (3NF) is achieved and, for the objectives of this book, I will stop at 3NF, too. To help explain the various normalization steps, an example scenario follows.

**First Normal Form (1NF)**

First normal form (1NF) involves the elimination of data redundancy or repeating information from a table. A table is considered to be in first normal form when all of its columns describe the table completely and when each column in a row has only one value. A table satisfies 1NF when each column in a row has a single value and no repeating group information. Essentially, every table meets 1NF as long as an array, list, or other structure has not been defined. The following table illustrates a table satisfying the 1NF rule because it has only one value at each row-and-column intersection. The table is in ascending order by CUSTNUM and consists of customers and the purchases they made at an office supply store.

### Table 1.1: First Normal Form (1NF) Table

<table>
<thead>
<tr>
<th>CUSTNUM</th>
<th>CUSTNAME</th>
<th>CUSTCITY</th>
<th>ITEM</th>
<th>UNITS</th>
<th>UNITCOST</th>
<th>MANUCITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Smith</td>
<td>San Diego</td>
<td>Chair</td>
<td>1</td>
<td>$179.00</td>
<td>San Diego</td>
</tr>
<tr>
<td>1</td>
<td>Smith</td>
<td>San Diego</td>
<td>Pens</td>
<td>12</td>
<td>$0.89</td>
<td>Los Angeles</td>
</tr>
<tr>
<td>1</td>
<td>Smith</td>
<td>San Diego</td>
<td>Paper</td>
<td>4</td>
<td>$6.95</td>
<td>Washington</td>
</tr>
<tr>
<td>1</td>
<td>Smithe</td>
<td>San Diego</td>
<td>Stapler</td>
<td>1</td>
<td>$8.95</td>
<td>Los Angeles</td>
</tr>
<tr>
<td>7</td>
<td>Lafler</td>
<td>Spring Valley</td>
<td>Mouse Pad</td>
<td>1</td>
<td>$11.79</td>
<td>San Diego</td>
</tr>
<tr>
<td>7</td>
<td>Loffler</td>
<td>Spring Valley</td>
<td>Pens</td>
<td>24</td>
<td>$1.59</td>
<td>Los Angeles</td>
</tr>
<tr>
<td>13</td>
<td>Thompson</td>
<td>Miami</td>
<td>Markers</td>
<td>.</td>
<td>$0.99</td>
<td>Los Angeles</td>
</tr>
</tbody>
</table>

**Second Normal Form (2NF)**

Second normal form (2NF) addresses the relationships between sets of data. A table is said to be in second normal form when all the requirements of 1NF are met and a foreign key is used to link any data in one table which has relevance to another table. The very nature of leaving a table in first normal form (1NF) may present problems due to the repetition of some information in the table. One noticeable problem is that Table 1.1 has repetitive information in it. Another problem is that there are misspellings in the customer name. Although repeating information may be permissible with hierarchical file structures and other legacy type file structures, it does pose a potential data consistency problem as it relates to relational data.

To describe how data consistency problems can occur, let's say that a customer takes a new job and moves to a new city. In changing the customer’s city to the new location, it would be very easy to miss one or more occurrences of the customer’s city resulting in a customer residing incorrectly in...
two different cities. Assuming that our table is only meant to track one unique customer per city, this would definitely be a data consistency problem. Essentially, second normal form (2NF) is important because it says that every non-key column must depend on the entire primary key.

Tables that subscribe to 2NF prevent the need to make changes in more than one place. What this means in normalization terms is that tables in 2NF have no partial key dependencies. As a result, our database that consists of a single table that satisfies 1NF will need to be split into two separate tables in order to subscribe to the 2NF rule. Each table would contain the CUSTNUM column to connect the two tables. Unlike the single table in 1NF, the tables in 2NF allow a customer’s city to be easily changed whenever they move to another city because the CUSTCITY column only appears once. The tables in 2NF would be constructed as follows.

Table 1.2: CUSTOMERS Table

<table>
<thead>
<tr>
<th>CUSTNUM</th>
<th>CUSTNAME</th>
<th>CUSTCITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Smith</td>
<td>San Diego</td>
</tr>
<tr>
<td>1</td>
<td>Smithe</td>
<td>San Diego</td>
</tr>
<tr>
<td>7</td>
<td>Lafler</td>
<td>Spring Valley</td>
</tr>
<tr>
<td>13</td>
<td>Thompson</td>
<td>Miami</td>
</tr>
</tbody>
</table>

Table 1.3: PURCHASES Table

<table>
<thead>
<tr>
<th>CUSTNUM</th>
<th>ITEM</th>
<th>UNITS</th>
<th>UNITCOST</th>
<th>MANUCITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Chair</td>
<td>1</td>
<td>$179.00</td>
<td>San Diego</td>
</tr>
<tr>
<td>1</td>
<td>Pens</td>
<td>12</td>
<td>$0.89</td>
<td>Los Angeles</td>
</tr>
<tr>
<td>1</td>
<td>Paper</td>
<td>4</td>
<td>$6.95</td>
<td>Washington</td>
</tr>
<tr>
<td>1</td>
<td>Stapler</td>
<td>1</td>
<td>$8.95</td>
<td>Los Angeles</td>
</tr>
<tr>
<td>7</td>
<td>Mouse Pad</td>
<td>1</td>
<td>$11.79</td>
<td>San Diego</td>
</tr>
<tr>
<td>7</td>
<td>Pens</td>
<td>24</td>
<td>$1.59</td>
<td>Los Angeles</td>
</tr>
<tr>
<td>13</td>
<td>Markers</td>
<td>.</td>
<td>$0.99</td>
<td>Los Angeles</td>
</tr>
</tbody>
</table>

Third Normal Form (3NF)

Referring to the two tables constructed according to the rules of 2NF, you may have noticed that the PURCHASES table contains a column called MANUCITY. The MANUCITY column stores the city where the product manufacturer is headquartered. Keeping this column in the PURCHASES table violates the third normal form (3NF) because MANUCITY does not provide factual information about the primary key column (CUSTNUM) in the PURCHASES table. Consequently, tables are considered to be in third normal form (3NF) when each column is dependent on the key, the whole key, and nothing but the key. The tables in 3NF are constructed so the MANUCITY column would be in a table of its own as follows.
Chapter 1: Designing Database Tables

Table 1.4: CUSTOMERS Table

<table>
<thead>
<tr>
<th>CUSTNUM</th>
<th>CUSTNAME</th>
<th>CUSTCITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Smith</td>
<td>San Diego</td>
</tr>
<tr>
<td>1</td>
<td>Smithe</td>
<td>San Diego</td>
</tr>
<tr>
<td>7</td>
<td>Lafler</td>
<td>Spring Valley</td>
</tr>
<tr>
<td>13</td>
<td>Thompson</td>
<td>Miami</td>
</tr>
</tbody>
</table>

Table 1.5: PURCHASES Table

<table>
<thead>
<tr>
<th>CUSTNUM</th>
<th>ITEM</th>
<th>UNITS</th>
<th>UNITCOST</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Chair</td>
<td>1</td>
<td>$179.00</td>
</tr>
<tr>
<td>1</td>
<td>Pens</td>
<td>12</td>
<td>$0.89</td>
</tr>
<tr>
<td>1</td>
<td>Paper</td>
<td>4</td>
<td>$6.95</td>
</tr>
<tr>
<td>1</td>
<td>Stapler</td>
<td>1</td>
<td>$8.95</td>
</tr>
<tr>
<td>7</td>
<td>Mouse Pad</td>
<td>1</td>
<td>$11.79</td>
</tr>
<tr>
<td>7</td>
<td>Pens</td>
<td>24</td>
<td>$1.59</td>
</tr>
<tr>
<td>13</td>
<td>Markers</td>
<td></td>
<td>$0.99</td>
</tr>
</tbody>
</table>

Table 1.6: MANUFACTURERS Table

<table>
<thead>
<tr>
<th>MANUNUM</th>
<th>MANUCITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>101</td>
<td>San Diego</td>
</tr>
<tr>
<td>112</td>
<td>San Diego</td>
</tr>
<tr>
<td>210</td>
<td>Los Angeles</td>
</tr>
<tr>
<td>212</td>
<td>Los Angeles</td>
</tr>
<tr>
<td>213</td>
<td>Los Angeles</td>
</tr>
<tr>
<td>214</td>
<td>Los Angeles</td>
</tr>
<tr>
<td>401</td>
<td>Washington</td>
</tr>
</tbody>
</table>

Beyond Third Normal Form

In general, database designers are satisfied when their database tables subscribe to the rules in 3NF. But, it is not uncommon for others to normalize their database tables to fourth normal form (4NF) where independent one-to-many relationships between primary key and non-key columns are forbidden. Some database purists will even normalize to fifth normal form (5NF) where tables are split into the smallest pieces of information in an attempt to eliminate any and all table redundancies. Although constructing tables in 5NF may provide the greatest level of database integrity, it is neither practical nor desired by most database practitioners.

There is no absolute right or wrong reason for database designers to normalize beyond 3NF as long as they have considered all the performance issues that may arise by doing so. A common problem that occurs when database tables are normalized beyond 3NF is that a large number of small tables are generated. In these cases, an increase in time and computer resources frequently occurs because small tables must first be joined before a query, report, or statistic can be produced.

Column Names and Reserved Words

According to the American National Standards Institute (ANSI), SQL is the standard language used with relational database management systems. The ANSI Standard reserves a number of SQL keywords from being used as column names. The SAS SQL implementation is not as rigid, but
users should be aware of what reserved words exist to prevent unexpected and unintended results during SQL processing. Column names should conform to proper SAS naming conventions (as described in the SAS Language Reference), and they should not conflict with certain reserved words found in the SQL language. The following list identifies the reserved words found in the ANSI SQL standard.

### ANSI SQL Reserved Words

<table>
<thead>
<tr>
<th>AS</th>
<th>INNER</th>
<th>OUTER</th>
</tr>
</thead>
<tbody>
<tr>
<td>CASE</td>
<td>INTERSECT</td>
<td>RIGHT</td>
</tr>
<tr>
<td>EXCEPT</td>
<td>JOIN</td>
<td>UNION</td>
</tr>
<tr>
<td>FROM</td>
<td>LEFT</td>
<td>UPPER</td>
</tr>
<tr>
<td>FULL</td>
<td>LOWER</td>
<td>USER</td>
</tr>
<tr>
<td>GROUP</td>
<td>ON</td>
<td>WHEN</td>
</tr>
<tr>
<td>HAVING</td>
<td>ORDER</td>
<td>WHERE</td>
</tr>
</tbody>
</table>

You probably will not encounter too many conflicts between a column name and an SQL reserved word, but when you do you will need to follow a few simple rules to prevent processing errors from occurring. As was stated earlier, although PROC SQL’s naming conventions are not as rigid as other vendor’s implementations, care should still be exercised, in particular when PROC SQL code is transferred to other database environments expecting it to run error free. If a column name in an existing table conflicts with a reserved word, you have three options at your disposal:

1. Physically rename the column name in the table, as well as any references to the column.
2. Use the RENAME= data set option to rename the desired column in the current query.
3. Specify the PROC SQL option DQUOTE=ANSI, and surround the column name (reserved word) in double quotes, as illustrated below.

#### SQL Code

```sql
PROC SQL DQUOTE=ANSI;
SELECT *
FROM RESERVED_WORDS
 WHERE "WHERE"='EXAMPLE' ;
QUIT;
```

### Data Integrity

*Webster’s New World Dictionary* defines *integrity* as “the quality or state of being complete; perfect condition; reliable; soundness.” Data integrity is a critical element that every organization must promote and strive for. It is imperative that the data tables in a database environment be reliable, free of errors, and sound in every conceivable way. The existence of data errors, missing information, broken links, and other related problems in one or more tables can impact decision-making and information reporting activities resulting in a loss of confidence among users.
Applying a set of rules to the database structure and content can ensure the integrity of data resources. These rules consist of table and column constraints, and will be discussed in detail in Chapter 5, “Creating, Populating, and Deleting Tables.”

**Referential Integrity**

Referential integrity refers to the way in which database tables handle update and delete requests. Database tables frequently have a *primary key* where one or more columns have a unique value by which rows in a table can be identified and selected. Other tables may have one or more columns called a *foreign key* that are used to connect to some other table through its value. Database designers frequently apply rules to database tables to control what happens when a primary key value changes and its effect on one or more foreign key values in other tables. These referential integrity rules apply restrictions on the data that may be updated or deleted in tables.

Referential integrity ensures that rows in one table have corresponding rows in another table. This prevents lost linkages between data elements in one table and those of another enabling the integrity of data to always be maintained. Using the 3NF tables defined earlier, a foreign key called CUSTNUM can be defined in the PURCHASES table that corresponds to the primary key CUSTNUM column in the CUSTOMERS table. Users are referred to Chapter 5, “Creating, Populating, and Deleting Tables” for more details on assigning referential integrity constraints.

---

**Database Tables Used in This Book**

This section describes a database or library of tables that is used by an imaginary computer hardware and software wholesaler. The library consists of six tables: Customers, Inventory, Invoice, Manufacturers, Products, and Purchases. The examples used throughout this book are based on this library (database) of tables and are described and displayed below. An alphabetical description of each table used throughout this book appears below.

**CUSTOMERS Table**

The CUSTOMERS table contains customers that have purchased computer hardware and software products from a manufacturer. Each customer is uniquely identified with a customer number. A description of each column in the Customers table follows.

<table>
<thead>
<tr>
<th>Table 1.7: Description of Columns in the Customers Table</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUSTNUM</td>
</tr>
<tr>
<td>CUSTNAME</td>
</tr>
<tr>
<td>CUSTCITY</td>
</tr>
</tbody>
</table>
INVENTORY Table

The INVENTORY table contains customer inventory information consisting of computer hardware and software products. The Inventory table contains no historical data. As inventories are replenished, the old quantity is overwritten with the new quantity. A description of each column in the Inventory table follows.

Table 1.8: Description of Columns in the Inventory Table

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRODNUM</td>
<td>Unique number identifying product.</td>
</tr>
<tr>
<td>MANUNUM</td>
<td>Unique number identifying the manufacturer.</td>
</tr>
<tr>
<td>INVENQTY</td>
<td>Number of units of product in stock.</td>
</tr>
<tr>
<td>ORDDATE</td>
<td>Date product was last ordered.</td>
</tr>
<tr>
<td>INVENCST</td>
<td>Cost of inventory in customer’s stock room.</td>
</tr>
</tbody>
</table>

INVOICE Table

The INVOICE table contains information about customers who purchased products. Each invoice is uniquely identified with an invoice number. A description of each column in the Invoice table follows.

Table 1.9: Description of Columns in the Invoice Table

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>INVNUM</td>
<td>Unique number identifying the invoice.</td>
</tr>
<tr>
<td>MANUNUM</td>
<td>Unique number identifying the manufacturer.</td>
</tr>
<tr>
<td>CUSTNUM</td>
<td>Customer number.</td>
</tr>
<tr>
<td>PRODNUM</td>
<td>Product number.</td>
</tr>
<tr>
<td>INVQTY</td>
<td>Number of units sold.</td>
</tr>
<tr>
<td>INVPRICE</td>
<td>Unit price.</td>
</tr>
</tbody>
</table>

MANUFACTURERS Table

The MANUFACTURERS table contains companies who make computer hardware and software products. Two companies cannot have the same name. No historical data is kept in this table. If a company is sold or stops making computer hardware or software, then the manufacturer is dropped from the table. In the event that a manufacturer has an address change, the old address is overwritten with the new address. A description of each column in the Manufacturers table follows.
Table 1.10: Description of Columns in the Manufacturers Table

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MANUNUM</td>
<td>Unique number identifying the manufacturer.</td>
</tr>
<tr>
<td>MANUNAME</td>
<td>Name of manufacturer.</td>
</tr>
<tr>
<td>MANUCITY</td>
<td>City where manufacturer is located.</td>
</tr>
<tr>
<td>MANUSTAT</td>
<td>State where manufacturer is located.</td>
</tr>
</tbody>
</table>

PRODUCTS Table

The PRODUCTS table contains computer hardware and software products offered for sale by the manufacturer. Each product is uniquely identified with a product number. A description of each column in the Products table follows.

Table 1.11: Description of Columns in the Products Table

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRODNUM</td>
<td>Unique number identifying the product.</td>
</tr>
<tr>
<td>PRODNAME</td>
<td>Name of product.</td>
</tr>
<tr>
<td>MANUNUM</td>
<td>Unique number identifying the manufacturer.</td>
</tr>
<tr>
<td>PRODTYPE</td>
<td>Type of product.</td>
</tr>
<tr>
<td>PRODCOST</td>
<td>Cost of product.</td>
</tr>
</tbody>
</table>

PURCHASES Table

The PURCHASES table contains computer hardware and software products purchased by customers. Each product is uniquely identified with a product number. A description of each column in the Purchases table follows.

Table 1.12: Description of Columns in the Purchases Table

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUSTNUM</td>
<td>Unique number identifying the customer.</td>
</tr>
<tr>
<td>ITEM</td>
<td>Name of product.</td>
</tr>
<tr>
<td>UNITS</td>
<td>Number of items purchased by customer.</td>
</tr>
<tr>
<td>UNITCOST</td>
<td>Cost of product.</td>
</tr>
</tbody>
</table>
Table Contents

An alphabetical list of tables, variables, and attributes for each table is displayed below.

Output 1.1: Customers CONTENTS Output

<table>
<thead>
<tr>
<th>#</th>
<th>Variable</th>
<th>Type</th>
<th>Len</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>custcity</td>
<td>Char</td>
<td>20</td>
<td>Customer’s Home City</td>
</tr>
<tr>
<td>2</td>
<td>custname</td>
<td>Char</td>
<td>25</td>
<td>Customer Name</td>
</tr>
<tr>
<td>1</td>
<td>custnum</td>
<td>Num</td>
<td>3</td>
<td>Customer Number</td>
</tr>
</tbody>
</table>

Output 1.2: Inventory CONTENTS Output

<table>
<thead>
<tr>
<th>#</th>
<th>Variable</th>
<th>Type</th>
<th>Len</th>
<th>Format</th>
<th>Informat</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>invencst</td>
<td>Num</td>
<td>6</td>
<td>DOLLAR10.2</td>
<td></td>
<td>Inventory Cost</td>
</tr>
<tr>
<td>2</td>
<td>invcnty</td>
<td>Num</td>
<td>3</td>
<td></td>
<td></td>
<td>Inventory Quantity</td>
</tr>
<tr>
<td>5</td>
<td>manunum</td>
<td>Num</td>
<td>3</td>
<td></td>
<td></td>
<td>Manufacturer Number</td>
</tr>
<tr>
<td>3</td>
<td>orddate</td>
<td>Num</td>
<td>4</td>
<td>MMDDYY10.</td>
<td>MMDDYY10.</td>
<td>Date Inventory Last Ordered</td>
</tr>
<tr>
<td>1</td>
<td>proddnum</td>
<td>Num</td>
<td>3</td>
<td></td>
<td></td>
<td>Product Number</td>
</tr>
</tbody>
</table>

Output 1.3: Invoice CONTENTS Output

<table>
<thead>
<tr>
<th>#</th>
<th>Variable</th>
<th>Type</th>
<th>Len</th>
<th>Format</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>custnum</td>
<td>Num</td>
<td>3</td>
<td></td>
<td>Customer Number</td>
</tr>
<tr>
<td>1</td>
<td>invnum</td>
<td>Num</td>
<td>3</td>
<td></td>
<td>Invoice Number</td>
</tr>
<tr>
<td>5</td>
<td>invprice</td>
<td>Num</td>
<td>5</td>
<td>DOLLAR12.2</td>
<td>Invoice Unit Price</td>
</tr>
<tr>
<td>4</td>
<td>invqty</td>
<td>Num</td>
<td>3</td>
<td></td>
<td>Invoice Quantity - Units Sold</td>
</tr>
<tr>
<td>2</td>
<td>manunum</td>
<td>Num</td>
<td>3</td>
<td></td>
<td>Manufacturer Number</td>
</tr>
<tr>
<td>6</td>
<td>proddnum</td>
<td>Num</td>
<td>3</td>
<td></td>
<td>Product Number</td>
</tr>
</tbody>
</table>
### Output 1.4: Manufacturers CONTENTS Output

<table>
<thead>
<tr>
<th>#</th>
<th>Variable</th>
<th>Type</th>
<th>Len</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>manucity</td>
<td>Char</td>
<td>20</td>
<td>Manufacturer City</td>
</tr>
<tr>
<td>2</td>
<td>manuname</td>
<td>Char</td>
<td>25</td>
<td>Manufacturer Name</td>
</tr>
<tr>
<td>1</td>
<td>manunum</td>
<td>Num</td>
<td>3</td>
<td>Manufacturer Number</td>
</tr>
<tr>
<td>4</td>
<td>manustat</td>
<td>Char</td>
<td>2</td>
<td>Manufacturer State</td>
</tr>
</tbody>
</table>

### Output 1.5: Products CONTENTS Output

<table>
<thead>
<tr>
<th>#</th>
<th>Variable</th>
<th>Type</th>
<th>Len</th>
<th>Format</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>manunum</td>
<td>Num</td>
<td>3</td>
<td>DOLLAR9.2</td>
<td>Manufacturer Number</td>
</tr>
<tr>
<td>5</td>
<td>prodcost</td>
<td>Num</td>
<td>5</td>
<td>DOLLAR9.2</td>
<td>Product Cost</td>
</tr>
<tr>
<td>2</td>
<td>prodname</td>
<td>Char</td>
<td>25</td>
<td></td>
<td>Product Name</td>
</tr>
<tr>
<td>1</td>
<td>prodnum</td>
<td>Num</td>
<td>3</td>
<td></td>
<td>Product Number</td>
</tr>
<tr>
<td>4</td>
<td>prodtype</td>
<td>Char</td>
<td>15</td>
<td></td>
<td>Product Type</td>
</tr>
</tbody>
</table>

### Output 1.6: Purchases CONTENTS Output

<table>
<thead>
<tr>
<th>#</th>
<th>Variable</th>
<th>Type</th>
<th>Len</th>
<th>Format</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>custnum</td>
<td>Num</td>
<td>4</td>
<td></td>
<td>Custnum</td>
</tr>
<tr>
<td>2</td>
<td>prodnunm</td>
<td>Num</td>
<td>3</td>
<td></td>
<td>Prodnunm</td>
</tr>
<tr>
<td>4</td>
<td>unitcost</td>
<td>Num</td>
<td>4</td>
<td>DOLLAR12.2</td>
<td>Unitcost</td>
</tr>
<tr>
<td>3</td>
<td>units</td>
<td>Num</td>
<td>3</td>
<td></td>
<td>Units</td>
</tr>
</tbody>
</table>
The Database Structure
The logical relationship between each table, and the columns common to each, appear below.

Figure 1.2. Logical Database Structure

Sample Database Tables
The following tables: Customers, Inventory, Manufacturers, Products, Invoice, and Purchases represent a relational database that will be illustrated in the examples in this book. These tables are small enough to follow easily, but complex enough to illustrate the power of SQL. The data contained in each table appears below.
Table 1.13: CUSTOMERS Table

<table>
<thead>
<tr>
<th>Obs</th>
<th>custnum</th>
<th>custname</th>
<th>custcity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>101</td>
<td>La Mesa Computer Land</td>
<td>La Mesa</td>
</tr>
<tr>
<td>2</td>
<td>201</td>
<td>Vista Tech Center</td>
<td>Vista</td>
</tr>
<tr>
<td>3</td>
<td>301</td>
<td>Coronado Internet Zone</td>
<td>Coronado</td>
</tr>
<tr>
<td>4</td>
<td>401</td>
<td>La Jolla Computing</td>
<td>La Jolla</td>
</tr>
<tr>
<td>5</td>
<td>501</td>
<td>Alpine Technical Center</td>
<td>Alpine</td>
</tr>
<tr>
<td>6</td>
<td>601</td>
<td>Oceanside Computer Land</td>
<td>Oceanside</td>
</tr>
<tr>
<td>7</td>
<td>701</td>
<td>San Diego Byte Store</td>
<td>San Diego</td>
</tr>
<tr>
<td>8</td>
<td>801</td>
<td>Jamul Hardware &amp; Software</td>
<td>Jamul</td>
</tr>
<tr>
<td>9</td>
<td>901</td>
<td>Del Mar Tech Center</td>
<td>Del Mar</td>
</tr>
<tr>
<td>10</td>
<td>1001</td>
<td>Lakeside Software Center</td>
<td>Lakeside</td>
</tr>
<tr>
<td>11</td>
<td>1101</td>
<td>Bonsall Network Store</td>
<td>Bonsall</td>
</tr>
<tr>
<td>12</td>
<td>1201</td>
<td>Rancho Santa Fe Tech</td>
<td>Rancho Santa Fe</td>
</tr>
<tr>
<td>13</td>
<td>1301</td>
<td>Spring Valley Byte Center</td>
<td>Spring Valley</td>
</tr>
<tr>
<td>14</td>
<td>1401</td>
<td>Poway Central</td>
<td>Poway</td>
</tr>
<tr>
<td>15</td>
<td>1501</td>
<td>Valley Center Tech Center</td>
<td>Valley Center</td>
</tr>
<tr>
<td>16</td>
<td>1601</td>
<td>Fairbanks Tech USA</td>
<td>Fairbanks Ranch</td>
</tr>
<tr>
<td>17</td>
<td>1701</td>
<td>Blossom Valley Tech</td>
<td>Blossom Valley</td>
</tr>
<tr>
<td>18</td>
<td>1801</td>
<td>Chula Vista Networks</td>
<td></td>
</tr>
</tbody>
</table>

N = 18
Table 1.14: INVENTORY Table

<table>
<thead>
<tr>
<th>Obs</th>
<th>prodmum</th>
<th>invenqty</th>
<th>orddate</th>
<th>invencst</th>
<th>manunum</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1110</td>
<td>20</td>
<td>09/01/2000</td>
<td>$45,000.00</td>
<td>111</td>
</tr>
<tr>
<td>2</td>
<td>1700</td>
<td>10</td>
<td>08/15/2000</td>
<td>$28,000.00</td>
<td>170</td>
</tr>
<tr>
<td>3</td>
<td>5001</td>
<td>5</td>
<td>08/15/2000</td>
<td>$1,000.00</td>
<td>500</td>
</tr>
<tr>
<td>4</td>
<td>5002</td>
<td>3</td>
<td>08/15/2000</td>
<td>$900.00</td>
<td>500</td>
</tr>
<tr>
<td>5</td>
<td>5003</td>
<td>10</td>
<td>08/15/2000</td>
<td>$2,000.00</td>
<td>500</td>
</tr>
<tr>
<td>6</td>
<td>5004</td>
<td>20</td>
<td>09/01/2000</td>
<td>$1,400.00</td>
<td>500</td>
</tr>
<tr>
<td>7</td>
<td>5001</td>
<td>2</td>
<td>09/01/2000</td>
<td>$1,200.00</td>
<td>600</td>
</tr>
</tbody>
</table>

N = 7

Table 1.15: INVOICE Table

<table>
<thead>
<tr>
<th>Obs</th>
<th>invnum</th>
<th>manunum</th>
<th>custnum</th>
<th>invqty</th>
<th>invprice</th>
<th>prodmum</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1001</td>
<td>500</td>
<td>201</td>
<td>5</td>
<td>$1,495.00</td>
<td>5001</td>
</tr>
<tr>
<td>2</td>
<td>1002</td>
<td>600</td>
<td>1301</td>
<td>2</td>
<td>$1,598.00</td>
<td>6001</td>
</tr>
<tr>
<td>3</td>
<td>1003</td>
<td>210</td>
<td>101</td>
<td>7</td>
<td>$245.00</td>
<td>2101</td>
</tr>
<tr>
<td>4</td>
<td>1004</td>
<td>111</td>
<td>501</td>
<td>3</td>
<td>$9,600.00</td>
<td>1110</td>
</tr>
<tr>
<td>5</td>
<td>1005</td>
<td>500</td>
<td>801</td>
<td>2</td>
<td>$798.00</td>
<td>5002</td>
</tr>
<tr>
<td>6</td>
<td>1006</td>
<td>500</td>
<td>901</td>
<td>4</td>
<td>$396.00</td>
<td>6000</td>
</tr>
<tr>
<td>7</td>
<td>1007</td>
<td>500</td>
<td>401</td>
<td>7</td>
<td>$23,100.00</td>
<td>1200</td>
</tr>
</tbody>
</table>

N = 7
### Table 1.16: MANUFACTURERS Table

<table>
<thead>
<tr>
<th>Obs</th>
<th>manunum</th>
<th>manuname</th>
<th>manucity</th>
<th>manustat</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>111</td>
<td>Cupid Computer</td>
<td>Houston</td>
<td>TX</td>
</tr>
<tr>
<td>2</td>
<td>210</td>
<td>Global Comm Corp</td>
<td>San Diego</td>
<td>CA</td>
</tr>
<tr>
<td>3</td>
<td>600</td>
<td>World Internet Corp</td>
<td>Miami</td>
<td>FL</td>
</tr>
<tr>
<td>4</td>
<td>120</td>
<td>Storage Devices Inc</td>
<td>San Mateo</td>
<td>CA</td>
</tr>
<tr>
<td>5</td>
<td>500</td>
<td>KPL Enterprises</td>
<td>San Diego</td>
<td>CA</td>
</tr>
<tr>
<td>6</td>
<td>700</td>
<td>San Diego PC Planet</td>
<td>San Diego</td>
<td>CA</td>
</tr>
</tbody>
</table>

N = 6

### Table 1.17: PRODUCTS Table

<table>
<thead>
<tr>
<th>Obs</th>
<th>prodnum</th>
<th>prodname</th>
<th>manunum</th>
<th>prodtype</th>
<th>prodcost</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1110</td>
<td>Dream Machine</td>
<td>111</td>
<td>Workstation</td>
<td>$3,200.00</td>
</tr>
<tr>
<td>2</td>
<td>1200</td>
<td>Business Machine</td>
<td>120</td>
<td>Workstation</td>
<td>$3,300.00</td>
</tr>
<tr>
<td>3</td>
<td>1700</td>
<td>Travel Laptop</td>
<td>170</td>
<td>Laptop</td>
<td>$3,400.00</td>
</tr>
<tr>
<td>4</td>
<td>2101</td>
<td>Analog Cell Phone</td>
<td>210</td>
<td>Phone</td>
<td>$35.00</td>
</tr>
<tr>
<td>5</td>
<td>2102</td>
<td>Digital Cell Phone</td>
<td>210</td>
<td>Phone</td>
<td>$175.00</td>
</tr>
<tr>
<td>6</td>
<td>2200</td>
<td>Office Phone</td>
<td>220</td>
<td>Phone</td>
<td>$130.00</td>
</tr>
<tr>
<td>7</td>
<td>5001</td>
<td>Spreadsheet Software</td>
<td>500</td>
<td>Software</td>
<td>$299.00</td>
</tr>
<tr>
<td>8</td>
<td>5002</td>
<td>Database Software</td>
<td>500</td>
<td>Software</td>
<td>$399.00</td>
</tr>
<tr>
<td>9</td>
<td>5003</td>
<td>Wordprocessor Software</td>
<td>500</td>
<td>Software</td>
<td>$299.00</td>
</tr>
<tr>
<td>11</td>
<td>5004</td>
<td>Graphics Software</td>
<td>500</td>
<td>Software</td>
<td>$299.00</td>
</tr>
</tbody>
</table>

N = 10
Table 1.18: PURCHASES Table

<table>
<thead>
<tr>
<th>Obs</th>
<th>custnum</th>
<th>prodnm</th>
<th>units</th>
<th>unitcost</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1701</td>
<td>1110</td>
<td>1</td>
<td>$3,200.00</td>
</tr>
<tr>
<td>2</td>
<td>101</td>
<td>5001</td>
<td>7</td>
<td>$299.00</td>
</tr>
<tr>
<td>3</td>
<td>701</td>
<td>5001</td>
<td>11</td>
<td>$299.00</td>
</tr>
<tr>
<td>4</td>
<td>701</td>
<td>5003</td>
<td>8</td>
<td>$299.00</td>
</tr>
<tr>
<td>5</td>
<td>701</td>
<td>5002</td>
<td>4</td>
<td>$399.00</td>
</tr>
<tr>
<td>6</td>
<td>701</td>
<td>5004</td>
<td>3</td>
<td>$299.00</td>
</tr>
<tr>
<td>7</td>
<td>701</td>
<td>1700</td>
<td>2</td>
<td>$3,400.00</td>
</tr>
<tr>
<td>8</td>
<td>701</td>
<td>1200</td>
<td>3</td>
<td>$3,300.00</td>
</tr>
<tr>
<td>9</td>
<td>701</td>
<td>1110</td>
<td>2</td>
<td>$3,200.00</td>
</tr>
<tr>
<td>10</td>
<td>1301</td>
<td>5001</td>
<td>3</td>
<td>$299.00</td>
</tr>
<tr>
<td>11</td>
<td>1301</td>
<td>5003</td>
<td>5</td>
<td>$299.00</td>
</tr>
<tr>
<td>12</td>
<td>1301</td>
<td>5002</td>
<td>2</td>
<td>$399.00</td>
</tr>
<tr>
<td>13</td>
<td>901</td>
<td>1700</td>
<td>2</td>
<td>$3,400.00</td>
</tr>
<tr>
<td>14</td>
<td>901</td>
<td>1200</td>
<td>3</td>
<td>$3,300.00</td>
</tr>
<tr>
<td>15</td>
<td>901</td>
<td>1110</td>
<td>5</td>
<td>$3,200.00</td>
</tr>
<tr>
<td>16</td>
<td>901</td>
<td>5001</td>
<td>9</td>
<td>$299.00</td>
</tr>
<tr>
<td>17</td>
<td>901</td>
<td>5002</td>
<td>5</td>
<td>$399.00</td>
</tr>
<tr>
<td>18</td>
<td>901</td>
<td>5003</td>
<td>8</td>
<td>$299.00</td>
</tr>
<tr>
<td>19</td>
<td>901</td>
<td>5004</td>
<td>2</td>
<td>$299.00</td>
</tr>
<tr>
<td>20</td>
<td>401</td>
<td>5001</td>
<td>11</td>
<td>$299.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>21</td>
<td>401</td>
<td>5002</td>
<td>5</td>
<td>$399.00</td>
</tr>
<tr>
<td>22</td>
<td>401</td>
<td>5003</td>
<td>7</td>
<td>$299.00</td>
</tr>
<tr>
<td>23</td>
<td>401</td>
<td>5004</td>
<td>3</td>
<td>$299.00</td>
</tr>
<tr>
<td>24</td>
<td>401</td>
<td>1700</td>
<td>3</td>
<td>$3,400.00</td>
</tr>
<tr>
<td>25</td>
<td>401</td>
<td>1200</td>
<td>6</td>
<td>$3,300.00</td>
</tr>
<tr>
<td>26</td>
<td>201</td>
<td>5001</td>
<td>6</td>
<td>$299.00</td>
</tr>
<tr>
<td>27</td>
<td>201</td>
<td>5001</td>
<td>6</td>
<td>$299.00</td>
</tr>
<tr>
<td>28</td>
<td>201</td>
<td>5003</td>
<td>9</td>
<td>$299.00</td>
</tr>
<tr>
<td>29</td>
<td>201</td>
<td>5002</td>
<td>4</td>
<td>$399.00</td>
</tr>
<tr>
<td>30</td>
<td>201</td>
<td>1700</td>
<td>3</td>
<td>$3,400.00</td>
</tr>
<tr>
<td>31</td>
<td>901</td>
<td>5001</td>
<td>2</td>
<td>$299.00</td>
</tr>
<tr>
<td>32</td>
<td>201</td>
<td>5001</td>
<td>2</td>
<td>$299.00</td>
</tr>
<tr>
<td>33</td>
<td>201</td>
<td>2102</td>
<td>5</td>
<td>$175.00</td>
</tr>
<tr>
<td>34</td>
<td>1101</td>
<td>2102</td>
<td>9</td>
<td>$175.00</td>
</tr>
<tr>
<td>35</td>
<td>1301</td>
<td>2102</td>
<td>11</td>
<td>$175.00</td>
</tr>
<tr>
<td>36</td>
<td>1401</td>
<td>2102</td>
<td>7</td>
<td>$175.00</td>
</tr>
<tr>
<td>37</td>
<td>801</td>
<td>2102</td>
<td>5</td>
<td>$175.00</td>
</tr>
<tr>
<td>38</td>
<td>501</td>
<td>2102</td>
<td>12</td>
<td>$175.00</td>
</tr>
<tr>
<td>39</td>
<td>301</td>
<td>2102</td>
<td>8</td>
<td>$175.00</td>
</tr>
<tr>
<td>40</td>
<td>1101</td>
<td>2200</td>
<td>3</td>
<td>$130.00</td>
</tr>
<tr>
<td>41</td>
<td>101</td>
<td>2102</td>
<td>9</td>
<td>$175.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>42</td>
<td>101</td>
<td>5003</td>
<td>3</td>
<td>$299.00</td>
</tr>
<tr>
<td>43</td>
<td>101</td>
<td>5004</td>
<td>2</td>
<td>$299.00</td>
</tr>
<tr>
<td>44</td>
<td>101</td>
<td>1200</td>
<td>3</td>
<td>$3,300.00</td>
</tr>
<tr>
<td>45</td>
<td>101</td>
<td>1700</td>
<td>5</td>
<td>$3,400.00</td>
</tr>
<tr>
<td>46</td>
<td>1301</td>
<td>1700</td>
<td>3</td>
<td>$3,400.00</td>
</tr>
<tr>
<td>47</td>
<td>1601</td>
<td>1700</td>
<td>7</td>
<td>$3,400.00</td>
</tr>
<tr>
<td>48</td>
<td>1801</td>
<td>1700</td>
<td>4</td>
<td>$3,400.00</td>
</tr>
<tr>
<td>49</td>
<td>1001</td>
<td>1700</td>
<td>5</td>
<td>$3,400.00</td>
</tr>
<tr>
<td>50</td>
<td>1101</td>
<td>1700</td>
<td>2</td>
<td>$3,400.00</td>
</tr>
<tr>
<td>51</td>
<td>1201</td>
<td>1200</td>
<td>8</td>
<td>$3,300.00</td>
</tr>
<tr>
<td>52</td>
<td>501</td>
<td>5001</td>
<td>3</td>
<td>$299.00</td>
</tr>
<tr>
<td>53</td>
<td>501</td>
<td>5003</td>
<td>5</td>
<td>$299.00</td>
</tr>
<tr>
<td>54</td>
<td>501</td>
<td>5004</td>
<td>1</td>
<td>$299.00</td>
</tr>
<tr>
<td>55</td>
<td>501</td>
<td>1700</td>
<td>4</td>
<td>$3,400.00</td>
</tr>
<tr>
<td>56</td>
<td>301</td>
<td>5001</td>
<td>6</td>
<td>$299.00</td>
</tr>
<tr>
<td>57</td>
<td>501</td>
<td>2102</td>
<td>9</td>
<td>$175.00</td>
</tr>
</tbody>
</table>

N = 57
Summary

1. Good database design often improves the relative ease by which tables can be created and populated in a relational database and can be implemented into any database (see the “Conceptual View” section).

2. SQL was designed to work with sets of data and accesses a data structure known as a table or a “virtual” table, known as a view (see the “Table Definitions” section).

3. Achieving optimal design of a database means that the database contains little or no redundant information in two or more of its tables. This means that good database design calls for little or no replication of data (see the “Redundant Information” section).

4. Good database design avoids data redundancy, update anomalies, costly or inefficient processing, coding complexities, complex logical relationships, long application development times, and/or excessive storage requirements (see the “Normalization” section).

5. Design decisions made in one phase may involve making one or more tradeoffs in another phase (see the “Normalization” section).

6. A database in third normal form (3NF) is defined as a column that is dependent on the key, the whole key, and nothing but the key (see the “Normalization” section).
Index

A
ADD clause 210–211
addition (+) operator 38–40
ad-hoc queries 280
_AGGR option 342
aggregate functions
creating macro variables with 152–153
specifying 257–258
ALL keyword 268
Ambiguous reference, column error 347
AND operator 36–38
ANSI (American National Standards Institute) 7–8
APPEND procedure 276
arithmetic data 29–32
arithmetic operators 38–40
AS keyword 32–34
_ASGN option 342
asterisk (*) wildcard 59
automatic macro variables, controlling
processing with 156–158
automatic sorting 353
AVG function 59

B
BEST option 353
BETWEEN predicate 63–65
B-tree 223
bugs 324
bulk loading data from Microsoft Excel 186–191
Burlew, Michele M.
Output Delivery System: The Basics and Beyond 114

BY statement 144

C
calculated column view 298
calculated columns 299
cardinality 225
Carpenter, Art
Carpenter's Complete Guide to the SAS Macro Language, Second Edition (Carpenter) 148
Cartesian product joins 242
Cartesian Product query 237
CASE expressions
about 123–124
assigning labels and grouping data 143–146
case logic versus COALESCE expression 142–143
logic and nulls 146–148
searched 137–142
simple 124–136
case logic, COALESCE expression versus 142–143
CAT function 41
CATALOGS dictionary table 73, 79
CATS function 41
CATX function 100
Celko, Joe
SQL for Smarties: Advanced SQL Programming 4, 47, 48, 298
change control, preserving 202
character data 28
character strings
concatenating 97–101
operators and functions 40–58
characters, aligning 43–44
CHECK constraint 196–197
CHECK_CONSTRAINTS dictionary table 73
COALESCE expression, case logic versus 142–143
COALESCE function 49, 142–143
coding
  See complex queries, coding
coding logic
    about 123
    CASE expressions 123–148
    conditional logic 122–123
    interfacing PROC SQL with Macro language 148–161
column aliases, creating 32–33
column constraints 192
column names 7–8, 75–78
column-definition lists, creating tables using 165–169
columns
  adding data to in rows 176–185
  adding to tables 210–211
  calculated 299
  changing format of 218–219
  changing label 219
  changing length of 214–218
  controlling position of in tables 212–214
  cross-referencing 158–159
  derived 299
  inserting text and constants between 99–101
  modifying columns containing indexes 229
  ordering output by 104–107
  renaming 219–221
  using scalar expressions with selected 101–104
COLUMNS dictionary table 73, 79–80
Comma Separated Value (CSV) file 116
comparison operators 35–36
The Complete Guide to Using SAS Indexes (Raithel) 224
complex comparisons, with searched CASE expressions 139–140
complex queries, coding
  about 234–235
  Cartesian product joins 242
  complex query applications 280–292
  inner joins 237, 242–255
  joins 235–255
  outer joins 238, 255–260
  set operations 238, 271–280
  subqueries 238, 261–271
  types of 236–239
complex query applications
  about 280
  determining number of rows in input tables 290–291
  identifying tables with most indexes 291–292
  many-to-many relationships 237, 280–285
  many-to-one relationships 237, 280–285
  one-to-many relationships 237, 280–285
  one-to-one relationships 237, 280–285
  processing first, last, and between rows for BY-and groups 285–290
composite indexes 228–229
combinations, performing in joins 250–251
concatenating
  character strings 97–101
  rows from two queries 276–278
concatenating strings 40–41
concatenation character string operator (||) 97–101
conceptual database design 2–3
conditional logic 122–123
constants, inserting between columns 99–101
CONSTRANT_COLUMN_USAGE dictionary table 73
CONSTRANT_TABLE_USAGE dictionary table 73
CONTENTS procedure 211–216, 218, 300–301, 355–358
correlated subqueries 238, 261, 269–271
COUNT function 59, 60, 160–161
CREATE VIEW statement  213–214, 297, 299–300
Cross Join query  237
cross-referencing columns  158–159
CSS function  59
CSV (Comma Separated Value) file  116
custom queries  280
CUSTOMERS table  9, 12, 15
customized lists
   creating with searched CASE expressions 140–142
   creating with simple CASE expressions 129–130
CV function  59

D
DASD (Direct Access Storage Device)  352
data
   about  24
   accessing dictionary table's contents  78–88
   adding to columns in rows  176–185
   adding to tables with SET clause  173–175
   adding with SELECT query  185–186
   arithmetic  29–32
   arithmetic operators  38–40
   bulk loading from Microsoft Excel  186–191
   character  28
   character string operators and functions  40–58
   comparison operators  35–36
   date and time column definitions  27–28
   deleting rows of  320
dictionary table column names  75–78
dictionary tables  72–88
displaying dictionary table definitions  74–75
exporting to Microsoft Excel  116–118
grouping  107–110, 143–146
   grouping with summary functions  107–109
logical operators  36–38
metadata and dictionary tables  72–74
missing  29–32
missing values  28–29, 66–68
NULL values  3, 28–29, 66–68
numeric  24–27
predicates  62–71
sorting  109–110
SQL keywords  32–34
SQL operators and functions  35–71
summarizing  58–62
types  24
   updating in tables  230–231
   updating rows of  313–320
Data Definition Language (DDL) statements  210
data integrity  8–9
data problems  326
data security  307–308
database design
   about  2
   column names  7–8
   conceptual view of  2–3
   data integrity  8–9
database tables used in this book  9–20
   normalization  4–7
   redundant information  3–4
   reserved words  7–8
table definitions  3
database structure  14
database tables
   See tables
database-enforced constraints
   See integrity constraints
DATAITEMS dictionary table  73
DATASETS procedure  221–222
date column definitions  26–27
DDL (Data Definition Language) statements  210
debugging

See also troubleshooting
about 323
bugs 324
examples 345–349
with macro variables 343–345
process of 324–325
techniques for 327–342
defining indexes 227–228
DELETE statement 203–205
deleting
indexes 229–230
rows in tables 203–205
rows of data 320
tables 205–208
views 321
derived columns 299
DESCRIBE TABLE statement 74–75, 202–203
DESCRIBE VIEW statement 301–302
DESTINATIONS dictionary table 73
_DFRT option 342
DICTIONARIES dictionary table 73, 80–82
dictionary tables
about 72
accessing content 78–88
column names 75–78
displaying definitions 74–75
metadata and 72–74
Direct Access Storage Device (DASD) 352
DISTINCT keyword 32–34
division by zero, preventing with simple CASE
expressions 132–133
division (/) operator 38–40
DROP INDEX statement 229–230
DROP TABLE statement 205–208, 222
DROP VIEW statement 321
dropping indexes 229–230
duplicate values, finding 33–34
DUPS 160–161

E

Effective Methods for Software Testing (Perry) 324
ENGINEES dictionary table 73
equals ( = ) operator 38–40, 243, 245, 262–266
equijoins 237, 243–245
ERRORSTOP/NOERRORSTOP option 340–341
EXCEPT operator 238, 278–280
EXEC/NOEXEC option 338–340
EXISTS predicate 71
exponent (** ) operator 38–40
exporting data and output to Microsoft Excel 116–118
EXTFILES dictionary table 73, 82

F

feature creep problems 327
FEEDBACK option 328–331
fifth normal form (5NF) 7
FILENAME statement 82
FILTERS dictionary table 73
first normal form (1NF) 5
foreign key 9, 199–202
FORMAT= column modifier 96–97
FORMAT procedure 144–145
FORMATS dictionary table 73
formatting output
about 92
concatenating character strings 97–99
converting output to rich text format 115–116
delivering results to Web 118–119
displaying row numbers 93–96
exporting data and output to Excel 116–118
FORMAT= column modifier 96–97
grouping data and sorting 109–110
grouping data with summary functions 107–109
inserting text and constants between columns 99–101
ordering output by columns 104–107
with Output Delivery System (ODS) 112–119
scalar expressions 101–104
sending output to SAS data sets 114–115
subsetting groups with HAVING clause 110–112
writing blank lines between rows 92–93
fourth normal form (4NF) 7
FREQ function 59
FROM clause 78, 104–107
full outer joins 259–260
functions
  See SQL operators and functions
FUNCTIONS dictionary table 73

G

GOPTIONS dictionary table 73
greater than operator (>) 35–36, 246
GROUP BY clause 58, 107–112, 257–258, 354
grouped view 298
grouping 107–110, 143–146, 354
groups, subsetting with HAVING clause 110–112
Gupta, Sunil Kumar
  Quick Results with the Output Delivery System 114

H

hash join algorithm 239
Haworth, Lauren
  Output Delivery System: The Basics and Beyond 114
HOST option 353
Hsieh, Yuan
  The Science of Debugging 324
HTML statement 118–119
hybrid view 298

HyperText Markup Language (HTML) 117

I

IMPORT procedure 186, 190
IN predicate 65–66
INDEX function 45
index join algorithm 239
indexes
  about 210, 222–224
  composite 228–229
  creating 228–229
  defining 227–228
  deleting 229–230
  designing 224–225
  identifying tables with most 291–292
  modifying columns containing 229
  optimizing WHERE clause processing with 358–365
  performance and 354–355
  preventing duplicate values in 229
  selectivity in 225–227
INDEXES dictionary table 73, 83
INFOMAPS dictionary table 73
information collection
  based on relationships 235–236
  as step in debugging process 325
inner joins
  about 237, 242–243
  equijoins 237, 243–245
  with more than three tables 253–255
  non-equijoins 237, 245–246
  performing computations in 250–251
  reflexive joins 237, 247–249
  self joins 237, 247–249
  with three tables 251–253
  using table aliases in 249–250
INOBS= option 333–334
INSERT INTO statement 172–186, 313–317, 343–344
integrity 9
integrity constraints
  about 192
Index

defining 192
deleting tables containing 206–208
displaying 202–203
preventing null values with NOT NULL constraint 192–195
referential 197–198
types of 192
INTERSECT operator 238, 273–274
INTO clause 151–152, 153–154, 154–155, 155–156
INVENTORY table 10, 12, 16
INVOICE table 10, 12, 16
IS MISSING predicate 67–68
IS NOT NULL predicate 67
IS NULL predicate 66–68

JOIN construct 131–132
joined view 298
joins
about 235
algorithms 239
importance of 235–242
influencing 239–242
with more than three tables 253–255
performing computations in 250–251
with three tables 251–253
using table aliases in 249–250

LABEL= option 103–104
labels, assigning 143–146
LEFT function 43–44
left outer joins 255–258
LENGTH function 42
LENGTH= modifier 25–26
LENGTH statement 25, 166, 168, 216, 217–218
%LET, creating macro variables with 149–151
LIBNAME statement 186
LIBNAMES dictionary table 73
LIKE clause, creating tables using 169–170
LIKE predicate 68–70
logic
conditional 122–123
nulls and 146–148
logic complexities, hiding 308–310
logic problems 326
logical design 2–3
logical operators 36–38
LOOPS= option 334–336
LOWCASE function 45–46

M
macro applications, building 158–161
Macro language, interfacing PROC SQL with 148–161
macro tools, building 158–161
macro variables and values
about 149
controlling processing with 156–158
controlling selection and population of with WHERE clause 154–155
creating from table row columns 151–152
creating lists of values in 155–156
creating multiple 153–154
creating with aggregate functions 152–153
creating with %LET 149–151
troubleshooting and debugging with 343–345
MACROS dictionary table 73, 83–84
MAGIC option 239–242
MANUFACTURERS table 10–11, 13, 17
many-to-many relationships 237, 280–285
many-to-one relationships 237, 280–285
MAX function 59
McConnell, Steve
MEAN function 59
MEMBERS dictionary table 73, 84–85
metadata, dictionary tables and 72–74
_METHOD option 331–333
Microsoft Excel
  bulk loading from 186–191
  exporting data and output to 116–118
MIN function 59, 60
missing data 29–32
missing values 28–29, 66–68
MODIFY clause 214–218, 218–219
MONOTONIC() function 52–58
MSGLEVEL=1 331–333
multiplication (*) operator 38–40

OPTIONS dictionary table 73, 85–86
ORDER BY clause 104–107, 109–110, 154–155
outer joins
  about 238, 255
  full outer joins 259–260
  left outer joins 255–258
  right outer joins 258–259
OUTER UNION operator 238, 276–278
OUTOBS= option 341
output, formatting
  See formatting output
Output Delivery System (ODS) 112–119
Output Delivery System: The Basics and Beyond
  (Haworth, Zender, and Burlew) 114
OUTPUT statement 114–115

patterns
  finding in strings 68–70
  finding occurrences of with INDEX function 45
percent sign (%) 68–70
performance
  See tuning process
period (.) 29
Perry, William E.
  Effective Methods for Software Testing 324
phonetic matching 47–49
physical design 2–3
_PJD option 342
populating tables 172–191
precedence, set operators and 272
predicates
  about 62–63
  finding patterns in strings 68–70
  selecting nonconsecutive values 65–66
  selecting ranges of values 63–65
  testing for existence of values 71
  testing for NULL or MISSING values 66–68
primary key 9, 198–199

N

N function 59
nested loop join algorithm 239
nested view 298
nesting 134–135, 310–312
NMISS function 59
NOBS 159–160
NOFEEDBACK option 328–331
nonconsecutive values, selecting 65–66
non-eqijoins 237, 245–246
normalization 4–7
NOT IS NULL predicate 67
NOT NULL constraint 192–195
NOT operator 36–38, 68
NULL values 3, 28–29, 66–68
nulls, logic and 146–148
NUMBER option 55–58
numeric data 24–27

O

ODS (Output Delivery System) 112–119
ODS statement 114–115
180-322: Statement is not valid or it is used out
  of proper order error 348–349
one-to-many relationships 237, 280–285
one-to-one relationships 237, 280–285
operators, combining with functions 42–43
  See also SQL operators and functions
PRINT procedure  52, 92, 93, 304–305
problem assessment and classification, as step in
debugging process  325
problem identification, as step in debugging
process 325
problem resolution, as step in debugging process
325
PROC step  113
PRODTYPE macro variable  149
production-oriented queries  280
PRODUCTS table  11, 13, 17
PROMPT option  341–342
PROMPTS dictionary table  74
PROMPTXML dictionary table  74
propagation of nulls  29
PRT function  59
PURCHASES table  11, 13, 18–20
%PUT statement  330, 343–344

Q
queries
   See also complex queries, coding
   accessing rows from combination of two
      274–276
   accessing rows from intersection of two
      272–274
   ad-hoc  280
   comparing rows from two  278–280
   concatenating rows from two  276–278
   Cross Join  237
   custom  280
   production-oriented  280
   validating with VALIDATE statement
      327–328
Quick Results with the Output Delivery System
   (Gupta)  114
QUIT statement  93

R
Raithel, Michael
   The Complete Guide to Using SAS Indexes  224
RANGE function  59
read-only view  298
redundancy, eliminating  307
redundant information, in database design  3–4
referential integrity  9, 197–198
REFERENTIAL_CONSTRAINTS dictionary
   table  74
reflexive joins  237, 247–249
relationships, information retrieval based on
   235–236
REMEMBER dictionary table  74
renaming
   columns  219–221
   tables  221–222
requirements problems  327
reserved words  7–8
RESET statement  93, 336–338
rich text format, converting output to  115–116
RIGHT function  43–44
right outer joins  258–259
rows
   accessing from combination of two queries
      274–276
   accessing from intersection of two queries
      272–274
   adding data to columns in  176–185
   comparing from two queries  278–280
   concatenating from two queries  276–278
   deleting in tables  203–205
   deleting rows of data  320
   determining number of in tables  159–160
   displaying numbers  93–96
   identifying duplicates in tables  160–161
   passing more than one row with subqueries
      266–267
   producing numbers  52–58
   updating rows of data  313–317, 317–320
   writing blank lines between  92–93
_RSLV option  343
### Index 375

**S**

samples of database tables 14–20
SAS data sets, sending output to 114–115
*SAS Language Reference: Dictionary* 8, 27
*SAS Macro Language: Reference* (SAS Institute Inc.) 148
*SAS Procedures Guide* 75
SASHELP views 72–75
scalar expressions, using with selected columns 101–104
SCAN function 43
*The Science of Debugging* (Telles and Hsieh) 324
searched CASE expression
  about 137
  complex comparisons with 139–140
  creating customized lists with 140–142
  in SELECT clause 137–138
second normal form (2NF) 5–6
2NF (second normal form) 5–6
security, data 307–308
SELECT clause
  CREATE TABLE statement 215–216, 219–221
  searched CASE expressions in 137–138
  simple CASE expressions in 124–128, 130–131
SELECT query 185–186, 213–214, 222
SELECT statement
  FROM clause 78
  creating macro variables with aggregate functions 152–154
  creating views 299–300, 302–303
  finding first nonmissing value 49
  grouping data with summary functions 107–109
  with joins 235
MONOTONIC() function 52–58
SQLOOPS macro variable 344
SQLRC macro variable 345
summarizing data 58

updating rows of data 313–320
using scalar expressions with selected columns 101–104
validating queries 327–328
wildcard characters in 279
selectivity, of indexes 225–227
self joins 237, 247–249
SET clause 173–175, 230–231
set operation view 298
set operations
  about 238, 271
  accessing rows from combination of two queries 274–276
  accessing rows from intersection of two queries 272–274
  comparing rows from two queries 278–280
  concatenating rows from two queries 276–278
  precedence and 272
  rules for set operators 271–272
73-322: Expecting an AS error 345–346, 349
single-table view 298
solution complexity problems 327
sorting
  automatic 353
  data 109–110
  performance and 352–353
  user-specified 353
sort-merge join algorithm 239
SORTPGM= system option 353
SOUNDEX function 48–49
sounds-like operator (=* ) 47–49
splitting tables 354
*SQL for Smarties: Advanced SQL Programming* (Celko) 4, 47, 48, 298
SQL keywords 32–34
SQL language 7–8
SQL operators and functions
  about 35
  aggregate functions 152–153, 257–258
  arithmetic operators 38–40
  character string 40–58
  combining functions with operators 42–43
comparison operators 35–36
logical operators 36–38
predicates 62–71
summarizing data 58–62
SQL procedure 250–251, 328–349
SQL procedure joins, DATA step merges versus 236
SQLOBS macro variable 156–158, 343–344
SQLOOPS macro variable 156–158, 344
SQLRC macro variable 156–158, 345
statements
   See specific statements
STD function 59
STDERR function 59
strategies, normalization 5–7
strings
   changing case of 45–46
   concatenating 40–41
   extracting information from 46–47
   finding length of 42
   finding patterns in 68–70
structure, database 14
STYLES dictionary table 74
_SUBQ option 343
subqueries
   about 238, 261
   alternate approach to 261–262
   correlated 238, 261, 269–271
   passing more than one row with 266–267
   passing single values with 262–266
SUBSTR function 46–47, 126
subtraction (-) operator 38–40
SUM function 59, 60–61
summarizing data 58–62
summary functions, grouping data with 107–109
SUMWGT function 59
syntax problems 326
system messages, reviewing 355–358
system-related problems 326

T
T function 59
table aliases, using in joins 249–250
table constraints 192
table row columns, creating macro variables
   from 151–152
TABLE_CONSTRAINTS dictionary table 74
tables
   about 164, 210
   adding columns to 210–211
   adding data to with SET clause 173–175
   cardinality of 225
   controlling position of columns in 212–214
   creating 164–172
   creating from existing tables 170–172
   creating using column-definition lists 165–169
   creating using LIKE clause 169–170
   deleting 205–208
   deleting rows in 203–205
   identifying with most indexes 291–292
   integrity constraints 192–203
   joins with more than three 253–255
   joins with three 251–253
   modifying 210–222
   populating 172–191
   renaming 221–222
   samples 14–20
   splitting 354
   updating data in 230–231
   used in this book 9–20
TABLES dictionary table 74, 86–87
Telles, Matthew A.
   The Science of Debugging 324
testing
   environment problems 327
   for existence of values 71
   for missing values 66–68
   for NULL values 66–68
   text, inserting between columns 99–101
third normal form (3NF) 4, 6–7
time column definitions 26–27
TITLE statement 149
INDEX

TITLES dictionary table  74, 87–88
-TREE option  331–333
TRIM function  43, 98–99
troubleshooting
  See also debugging
  about  323
  examples  345–349
  with macro variables  343–345
  techniques for  327–342
  types of problems  326–327
truncated string comparison operators  36
tuning process
  about  351–352
  automatic sorting  353
  avoiding UNIONS  361–365
  constructing efficient logic conditions  359–361
  grouping and performance  354
  indexes and performance  354–355
  optimizing WHERE clause processing with indexes  358–365
  reviewing CONTENTS output and system messages  355–358
  sorting and performance  352–353
  SORTPGM= system option  353
  splitting tables  354
  user-specified sorting  353
  200-322: The symbol is not recognized and will be ignored error  347–348
  202-322: The option or parameter s not recognized and will be ignored error  346

U

underscore (_ )  70
undocumented SQL procedure options  342–349
UNION operator  238, 274–276
UNIONs, avoiding  361–365
UNIQUE keyword  32–34, 80, 195–196, 229
unique values, finding  34
UPCASE function  45–46
updatable views  298, 312–320
UPDATE query  337–338
upgrading
  data in tables  230–231
  rows of data  317–320
  tables conditionally with simple CASE expressions  135–136
usage error  324
user-specified sorting  353
USS function  59
_UTIL option  343

V

validate solution, as step in debugging process  325
VALIDATE statement  327–328
values
  See also macro variables and values
  comparing sets of  267–269
  creating lists of in macro variables  155–156
  finding duplicate  33–34
  finding first nonmissing  49–52
  finding unique  34
  missing  28–29, 66–68
  NULL  3, 28–29, 66–68
  passing single values with subqueries  262–266
  preventing duplicates in indexes  229
  selecting nonconsecutive  65–66
  selecting ranges of  63–65
  testing for existence of  71
VALUES clause  176–181, 316
VAR function  59
views
  about  296–297
  creating  299–303
  data security  307–308
  DATA steps and  305–306
deleting  321
deleting rows of data  320
describing definitions  301–302
displaying contents of  300–301
eliminating redundancy 307
hiding logic complexities 308–310
nesting 310–312
SAS procedures and 303–305
types of 297–298
updatable 298, 312–320
updating existing rows of data 317–320
using in SAS 302–303
VIEWS dictionary table 74, 88

W
Web, delivering results to 118–119
WHEN conditions 124, 137
WHERE clause
  CATALOGS dictionary 79
  combining functions and operators 43
  comparing sets of values 267–269
  conditional logic 122–123
  controlling selection and population of
    macro variables with 154–155
  creating macro variables from table row
    columns 151–152
  for deleting rows in tables 203–205
  greater than operator (>) in 246
  joins and 251–256
  optimizing processing with indexes
    358–365
  passing single values with subqueries
    262–266
  preventing division by zero with simple
    CASE expression 132–133
  selecting ranges of values 63–64
  set operations 272–274
  simple CASE expression in 128–129
  specifying 257
  subsetting groups with HAVING clause
    110–112
  TABLES dictionary 86–87
  updating rows of data 313–317, 317–320
WHERE expression 230–231

Y
YEAR function 63–64

Z
Zender, Cynthia L.
  Output Delivery System: The Basics and
    Beyond 114
zero, division by 132–133

Symbols
Symbols and Numerics
+ (addition) operator 38–40
* (asterisk) wildcard 59
|| (concatenation character string operator)
  97–101
/ (division) operator 38–40
= (equals operator) 38–40, 243, 245, 262–266
** (exponent) operator 38–40
> (greater than operator) 35–36, 246
* (multiplication) operator 38–40
% (percent sign) 68–70
. (period) 29
=*(sounds-like operator) 47–49
- (subtraction) operator 38–40
_ (underscore) 70
1NF (first normal form) 5
2NF (second normal form) 5–6
3NF (third normal form) 4, 6–7
4NF (fourth normal form) 7
5NF (fifth normal form) 7
73-322: Expecting an AS error 345–346, 349
180-322: Statement is not valid or it is used out
  of proper order error 348–349
200-322: The symbol is not recognized and will
  be ignored error 347–348
202-322: The option or parameters not
  recognized and will be ignored error 346
Kirk Paul Lafler is consultant and founder of Software Intelligence Corporation and has been using SAS since 1979. He is a SAS Certified Professional, provider of IT consulting services, trainer to SAS users around the world, and sasCommunity.org Advisory Board emeritus member. The author of 5 books, Kirk has written more than 500 papers and articles, been an invited speaker and trainer at 400-plus SAS users group conferences and meetings, and is the recipient of nearly two dozen “Best” contributed paper, hands-on workshop (HOW), and poster awards. For more than three decades he has supported the SAS users community by chairing the Southern California SAS Users Group (SoCalSUG), starting and chairing the San Diego SAS Users Group (SANDS), chairing and co-chairing academic sections at in-house, local, regional, and SAS Global Forum conferences, mentoring users, and contributing his popular SAS Tips column, “Kirk’s Korner of Quick and Simple Tips,” which appears regularly in several SAS Users Group newsletters and websites.

Learn more about this author by visiting his author page at http://support.sas.com/lafler.html. There you can download free chapters, access example code and data, read the latest reviews, get updates, and more.
Gain Greater Insight into Your SAS® Software with SAS Books.

Discover all that you need on your journey to knowledge and empowerment.

support.sas.com/bookstore
for additional books and resources.