Chapter 8: Validating and Cleaning Data

Objectives

- Define data errors in a raw data file.
- Identify procedures for validating data.
- Identify techniques for cleaning data.
- Define the business scenario that will be used with validating and cleaning data.

Business Scenario

A delimited raw data file containing information on Orion Star non-sales employees from Australia and the United States needs to be read to create a data set.

Requirements of non-sales employee data:

- **`Employee_ID`**, **`Salary`**, **`Birth_Date`**, and **`Hire_Date`** must be numeric variables.
- **`First`**, **`Last`**, **`Gender`**, **`Job_Title`**, and **`Country`** must be character variables.

8.01 Quiz

What problems will SAS have reading the numeric data **`Salary`** and **`Hire_Date`**?

Partial **`nonsales.csv`**

```
120101,Patrick,Lu,M,163040,Director,AU,18AUG1976,01JUL2003
120104,Kareen,Billington,F,46230,Administration Manager,au,11MAY1954,01JAN1981
120105,Liz,Povey,F,27110,Secretary I,AU,21DEC1974,01MAY1999
120106,John,Hornsey,M,unknown,Office Assistant II,AU,23DEC1944,01JAN1974
120107,Sherie,Sheedy,F,30475,Office Assistant III,AU,01FEB1978,21JAN1953
120108,Gladys,Gromek,F,27660,Warehouse Assistant II,AU,23FEB1984,01AUG2006
120108,Gabriele,Baker,F,26495,Warehouse Assistant I,AU,15DEC1986,01OCT2006
120110,Dennis,Entwisle,M,28615,Warehouse Assistant III,AU,20NOV1949,01NOV1979
120111,Ubaldo,Spillane,M,26895,Security Guard II,AU,23JUL1949,99NOV1978
120112,Ellis,Glattback,F,26550, ,AU,17FEB1969,01JUL1990
120113,Riu,Horsey,F,26870,Security Guard II,AU,10MAY1944,01JAN1974
120114,Jeannette,Buddery,G,31285,Security Manager,AU,08FEB1944,01JAN1974
120115,Hugh,Nichollas,M,2650,Service Assistant I,AU,08MAY1984,01AUG2005
.,Austen,Ralston,M,29250,Service Assistant II,AU,13JUN1959,01FEB1980
120117,Bill,Mccleary,M,31670,Cabinet Maker III,AU,11SEP1964,01APR1986
```

Data Errors

Data errors occur when data values are not appropriate for the SAS statements that are specified in a program.

- SAS detects data errors during program execution.
- When a data error is detected, SAS continues to execute the program.

```
NOTE: Invalid data for Salary in line 4 23-29.
RULE:     ----+----1----+----2----+----3----+----4----+----5----+----6
4         120106,John,Hornsey,M,unknown,Office Assistant II,AU,23DEC19
      61  44,01JAN1974 72
Employee_ID=120106 First=John Last=Hornsey Gender=M Salary=.
Job_Title=Office Assistant II Country=AU Birth_Date=23/12/1944
Hire_Date=01/01/1974 _ERROR_=1 _N_=4
NOTE: Invalid data for Hire_Date in li
9         120111,Ubaldo,Spillane,M,268                                  94
      61  9,99NOV1978 71
Employee_ID=120111 First=Ubaldo Last=S
Job_Title=Security Guard II Country=AU
Hire_Date=. _ERROR_=1 _N_=9
```

A data error example is defining a variable as numeric, but the data value is actually character.

Business Scenario

Additional requirements of non-sales employee data:

- **`Employee_ID`** must be unique and not missing.
- **`Gender`** must have a value of `F` or `M`.
- **`Salary`** must be in the numeric range of 24000 – 500000.
- **`Job_Title`** must not be missing.
- **`Country`** must have a value of `AU` or `US`.
- **`Birth_Date`** value must occur before **`Hire_Date`** value.
- **`Hire_Date`** must have a value of 01/01/1974 or later.

8.02 Quiz

What problems exist with the data in this partial data set?

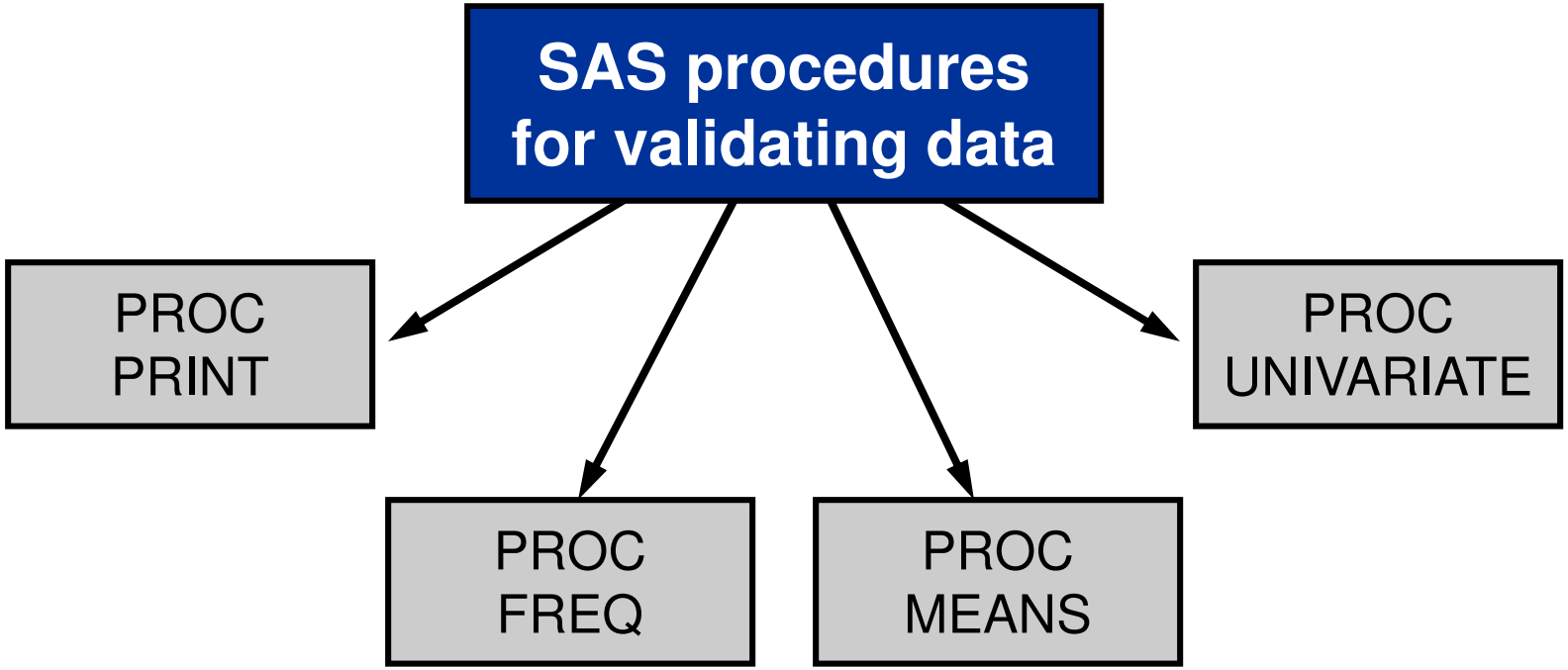
	Employee_ID	First	Last	Gender	Salary	Job_Title	Country	Birth_Date	Hire_Date
1	120101	Patrick	Lu	M	163E3	Director	AU	18/08/1976	01/07/2003
2	120104	Kareen	Billington	F	46230	Administration Manager	au	11/05/1954	01/01/1981
3	120105	Liz	Povey	F	27110	Secretary I	AU	21/12/1974	01/05/1999
4	120106	John	Hornsey	M	.	Office Assistant II	AU	23/12/1944	01/01/1974
5	120107	Sherie	Sheedy	F	30475	Office Assistant III	AU	01/02/1978	21/01/1953
6	120108	Gladys	Gromek	F	27660	Warehouse Assistant II	AU	23/02/1984	01/08/2006
7	120108	Gabriele	Baker	F	26495	Warehouse Assistant I	AU	15/12/1986	01/10/2006
8	120110	Dennis	Entwisle	M	28615	Warehouse Assistant III	AU	20/11/1949	01/11/1979
9	120111	Ubaldo	Spillane	M	26895	Security Guard II	AU	23/07/1949	.
10	120112	Ellis	Glattback	F	26550		AU	17/02/1969	01/07/1990
11	120113	Riu	Horsey	F	26870	Security Guard II	AU	10/05/1944	01/01/1974
12	120114	Jeannette	Buddery	G	31285	Security Manager	AU	08/02/1944	01/01/1974
13	120115	Hugh	Nichollas	M	2650	Service Assistant I	AU	08/05/1984	01/08/2005
14	.	Austen	Ralston	M	29250	Service Assistant II	AU	13/06/1959	01/02/1980
15	120117	Bill	Mccleary	M	31670	Cabinet Maker III	AU	11/09/1964	01/04/1986
16	120118	Darshi	Hartshorn	M	28090	Cabinet Maker II	AU	03/06/1959	01/07/1984

Hint: There are nine data problems.

Validating the Data

In general, SAS procedures analyze data, produce output, or manage SAS files.

In addition, SAS procedures can be used to detect invalid data.



The PRINT Procedure

The PRINT procedure can show the job titles that are missing and the hire dates that occur before the birth dates.

Obs	Employee_ ID	Job_Title	Birth_Date	Hire_Date
5	120107	Office Assistant III	01/02/1978	21/01/1953
9	120111	Security Guard II	23/07/1949	.
10	120112		17/02/1969	01/07/1990

p108d01

The FREQ Procedure

The FREQ procedure can show if any genders are not `F` or `M` and if any countries are not `AU` or `US`.

The FREQ Procedure

Gender	Frequency	Percent	Cumulative Frequency	Cumulative Percent
F	110	47.01	110	47.01
G	1	0.43	111	47.44
M	123	52.56	234	100.00

Frequency Missing = 1

Country	Frequency	Percent	Cumulative Frequency	Cumulative Percent
AU	33	14.04	33	14.04
US	196	83.40	229	97.45
au	3	1.28	232	98.72
us	3	1.28	235	100.00

The MEANS Procedure

The MEANS procedure can show if any salaries are not in the range of 24000 to 500000.

The MEANS Procedure

Analysis Variable : Salary

N	N Miss	Minimum	Maximum
234	1	2401.00	433800.00

p108d01

The UNIVARIATE Procedure

The UNIVARIATE procedure can show if any salaries are not in the range of 24000 to 500000.

Partial PROC UNIVARIATE Output

```
                    The UNIVARIATE Procedure
                       Variable:  Salary

                     Extreme Observations

           -----Lowest----          -----Highest----

            Value      Obs            Value      Obs

             2401       20           163040        1
             2650       13           194885      231
            24025       25           207885       28
            24100       19           268455       29
            24390      228           433800       27
```

p108d01

Cleaning the Data

After the data is validated, the invalid data needs to be cleaned.

Techniques for cleaning data:

- Editing raw data file outside of SAS
- Interactively editing data set using VIEWTABLE
- Programmatically editing data set using the DATA step
- Programmatically editing data set using the SQL procedure
- Using the SAS DataFlux product dfPower Studio

Chapter 8: Validating and Cleaning Data

Objectives

- Identify data errors.
- Demonstrate what happens when a data error is encountered.
- Direct the observations with data errors to a different data set than the observations without data errors. (Self-Study)

Business Scenario

A delimited raw data file containing information on Orion Star non-sales employees from Australia and the United States needs to be read to create a data set.

Requirements of non-sales employee data:

- **`Employee_ID`**, **`Salary`**, **`Birth_Date`**, and **`Hire_Date`** must be numeric variables.
- **`First`**, **`Last`**, **`Gender`**, **`Job_Title`**, and **`Country`** must be character variables.

8.03 Multiple Choice Poll

Which statements are used to read a delimited raw data file and create a SAS data set?

a. DATA and SET only
b. DATA and INFILE only
c. DATA, SET, and INPUT only
d. DATA, INFILE, and INPUT only

Data Errors

One type of data error is when the INPUT statement encounters invalid data in a field.

When SAS encounters a data error, these events occur:

- A note that describes the error is printed in the SAS log.
- The input record (contents of the input buffer) being read is displayed in the SAS log.
- The values in the SAS observation (contents of the PDV) being created are displayed in the SAS log.
- A missing value is assigned to the appropriate SAS variable.
- Execution continues.

Data Errors

A note that describes the error is printed in the SAS log.

Partial SAS Log

```
NOTE: Invalid data for Salary in line 4 23-29.
RULE:     ----+----1----+----2----+----3----+----4----+----5----+----6
4         120106,John,Hornsey,M,unknown,Office Assistant II,AU,23DEC19
      61  44,01JAN1974 72
Employee_ID=120106 First=John Last=Hornsey Gender=M Salary=.
Job_Title=Office Assistant II Country=AU Birth_Date=23/12/1944
Hire_Date=01/01/1974 _ERROR_=1 _N_=4
```

This note indicates that invalid data was found for variable **`Salary`** in line 4 of the raw data file in columns 23-29.

Data Errors

The input record (contents of the input buffer) being read is displayed in the SAS log.

Partial SAS Log

```
NOTE: Invalid data for Salary in line 4 23-29.
RULE:     ----+----1----+----2----+----3----+----4----+----5----+----6
4         120106,John,Hornsey,M,unknown,Office Assistant II,AU,23DEC19
      61  44,01JAN1974 72
Employee_ID=120106 First=John Last=Hornsey Gender=M Salary=.
Job_Title=Office Assistant II Country=AU Birth_Date=23/12/1944
Hire_Date=01/01/1974 _ERROR_=1 _N_=4
```

A ruler is drawn above the raw data record that contains the invalid data.

Data Errors

The values in the SAS observation (contents of the PDV) being created are displayed in the SAS log.

Partial SAS Log

```
NOTE: Invalid data for Salary in line 4 23-29.
RULE:     ----+----1----+----2----+----3----+----4----+----5----+----6
4         120106,John,Hornsey,M,unknown,Office Assistant II,AU,23DEC19
      61  44,01JAN1974 72
Employee_ID=120106 First=John Last=Hornsey Gender=M Salary=.
Job_Title=Office Assistant II Country=AU Birth_Date=23/12/1944
Hire_Date=01/01/1974 _ERROR_=1 _N_=4
```

Data Errors

A missing value is assigned to the appropriate SAS variable.

Partial SAS Log

```
NOTE: Invalid data for Salary in line 4 23-29.
RULE:     ----+----1----+----2----+----3----+----4----+----5----+----6
4         120106,John,Hornsey,M,unknown,Office Assistant II,AU,23DEC19
      61  44,01JAN1974 72
Employee_ID=120106 First=John Last=Hornsey Gender=M Salary=.
Job_Title=Office Assistant II Country=AU Birth_Date=23/12/1944
Hire_Date=01/01/1974 _ERROR_=1 _N_=4
```

Data Errors

During the processing of every DATA step, SAS automatically creates the following temporary variables:

- the _N_ variable, which counts the number of times the DATA step begins to iterate
- the _ERROR_ variable, which signals the occurrence of an error caused by the data during execution

  0 indicates that no errors exist.

  1 indicates that one or more errors occurred.

```
NOTE: Invalid data for Salary in line 4 23-29.
RULE:     ----+----1----+----2----+----3----+----4----+----5----+----6
4         120106,John,Hornsey,M,unknown,Office Assistant II,AU,23DEC19
      61  44,01JAN1974 72
Employee_ID=120106 First=John Last=Hornsey Gender=M Salary=.
Job_Title=Office Assistant II Country=AU Birth_Date=23/12/1944
Hire_Date=01/01/1974 _ERROR_=1 _N_=4
```

Setup for the Poll

- Submit program **p108a01**.
- Determine the reason for the invalid data that appears in the SAS log.

8.04 Multiple Choice Poll

Which statement best describes the invalid data?

a. The data in the raw data file is bad.
b. The programmer incorrectly read the data.

Outputting to Multiple Data Sets (Self-Study)

The DATA statement can specify multiple output data sets.

```
data work.baddata work.gooddata;
   length Employee_ID 8 First $ 12 Last $ 18
          Gender $ 1 Salary 8 Job_Title $ 25
          Country $ 2 Birth_Date Hire_Date 8;
   infile 'nonsales.csv' dlm=',';
   input Employee_ID First $ Last $
         Gender $ Salary Job_Title $ Country $
         Birth_Date :date9.
         Hire_Date :date9.;
   format Birth_Date Hire_Date ddmmyy10.;
   if _error_=1 then output work.baddata;
   else output work.gooddata;
run;
```

p108d03

Outputting to Multiple Data Sets (Self-Study)

An OUTPUT statement can be used in a conditional statement to write the current observation to a specific data set that is listed in the DATA statement.

```
data work.baddata work.gooddata;
   length Employee_ID 8 First $ 12 Last $ 18
          Gender $ 1 Salary 8 Job_Title $ 25
          Country $ 2 Birth_Date Hire_Date 8;
   infile 'nonsales.csv' dlm=',';
   input Employee_ID First $ Last $
         Gender $ Salary Job_Title $ Country $
         Birth_Date :date9.
         Hire_Date :date9.;
   format Birth_Date Hire_Date ddmmyy10.;
   if _error_=1 then output work.baddata;
   else output work.gooddata;
run;
```

p108d03

Outputting to Multiple Data Sets (Self-Study)

Partial SAS Log

```
NOTE: Invalid data for Salary in line 4 23-29.
RULE:     ----+----1----+----2----+----3----+----4----+----5----+----6
4         120106,John,Hornsey,M,unknown,Office Assistant II,AU,23DEC19
      61  44,01JAN1974 72
Employee_ID=120106 First=John Last=Hornsey Gender=M Salary=.
Job_Title=Office Assistant II Country=AU Birth_Date=23/12/1944
Hire_Date=01/01/1974 _ERROR_=1 _N_=4
NOTE: Invalid data for Hire_Date in line 9 63-71.
9         120111,Ubaldo,Spillane,M,26895,Security Guard II,AU,23JUL194
      61  9,99NOV1978 71
Employee_ID=120111 First=Ubaldo Last=Spillane Gender=M Salary=26895
Job_Title=Security Guard II Country=AU Birth_Date=23/07/1949
Hire_Date=. _ERROR_=1 _N_=9
NOTE: 235 records were read from the infile
      's:\workshop\nonsales.csv'.
      The minimum record length was 55.
      The maximum record length was 82.
NOTE: The data set WORK.BADDATA has 2 observations and 9 variables.
NOTE: The data set WORK.GOODDATA has 233 observations and 9 variables.
```

Chapter 8: Validating and Cleaning Data

Objectives

- Validate data by using the PRINT procedure with the WHERE statement.
- Validate data by using the FREQ procedure with the TABLES statement.

Business Scenario

Additional requirements of non-sales employee data:

- **`Employee_ID`** must be unique and not missing.
- **`Gender`** must have a value of `F` or `M`.
- **`Salary`** must be in the numeric range of 24000 – 500000.
- **`Job_Title`** must not be missing.
- **`Country`** must have a value of `AU` or `US`.
- **`Birth_Date`** value must occur before **`Hire_Date`** value.
- **`Hire_Date`** must have a value of 01/01/1974 or later.

SAS Procedures for Validating Data

SAS procedures can be used to detect invalid data.

PROC PRINT step with **VAR** and **WHERE** statements	detects invalid character and numeric values by subsetting observations based on conditions.
PROC FREQ step with **TABLES** statement	detects invalid character and numeric values by looking at distinct values.
PROC MEANS step with **VAR** statement	detects invalid numeric values by using summary statistics.
PROC UNIVARIATE step with **VAR** statement	detects invalid numeric values by looking at extreme values.

The PRINT Procedure

The PRINT procedure produces detail reports based on SAS data sets.

General form of the PRINT procedure:

```
PROC PRINT DATA=SAS-data-set ;
      VAR variable(s) ;
      WHERE where-expression ;
RUN;
```

- The VAR statement selects variables to include in the report and determines their order in the report.
- The WHERE statement is used to obtain a subset of observations.

The WHERE Statement

For validating data, the WHERE statement is used to retrieve the observations that do not meet the data requirements.

General form of the WHERE statement:

WHERE *where-expression* **;**

The *where-expression* is a sequence of operands and operators that form a set of instructions that define a condition for selecting observations.

- Operands include constants and variables.
- Operators are symbols that request a comparison, arithmetic calculation, or logical operation.

The WHERE Statement

The following PROC PRINT step retrieves observations that have missing values for **`Job_Title`**.

```
proc print data=orion.nonsales;
   var Employee_ID Last Job_Title;
   where Job_Title = ' ';
run;
```

Obs	Employee_ ID	Last	Job_ Title
10	120112	Glattback	

p108d04

The WHERE Statement

A WHERE statement might need to reference a SAS date value.

For example, the PRINT procedure needs to retrieve observations that have values of **`Hire_Date`** less than January 1, 1974.

What is the numeric SAS date value for January 1, 1974?

A *SAS date constant* is used to convert a calendar date to a SAS date value.

SAS Date Constant

To write a SAS date constant, enclose a date in quotation marks in the form ***ddMMMyyyy*** and immediately follow the final quotation mark with the letter **d**.

dd	is a one- or two-digit value for the day.
MMM	is a three-letter abbreviation for the month.
yyyy	is a four-digit value for the year.
d	is required to convert the quoted string to a SAS date.

Example:

The date constant for January 1, 1974, is `'01JAN1974'd`.

SAS Date Constant

The following PROC PRINT step retrieves observations that have values of **`Hire_Date`** that are less than January 1, 1974.

```
proc print data=orion.nonsales;
   var Employee_ID Birth_Date Hire_Date;
   where Hire_Date < '01JAN1974'd;
run;
```

Obs	Employee_ ID	Birth_Date	Hire_Date
5	120107	01/02/1978	21/01/1953
9	120111	23/07/1949	.
214	121011	11/03/1944	01/01/1968

p108d04

8.05 Multiple Choice Poll

Which data requirement cannot be achieved with the PRINT procedure using a WHERE statement?

a. **`Employee_ID`** must be unique and not missing.
b. **`Gender`** must have a value of `F` or `M`.
c. **`Salary`** must be in the numeric range of 24000 – 500000.
d. **`Job_Title`** must not be missing.
e. **`Country`** must have a value of `AU` or `US`.
f. **`Birth_Date`** value must occur before **`Hire_Date`** value.
g. **`Hire_Date`** must have a value of 01/01/1974 or later.

Data Requirements

Data Requirement	*where-expression* to obtain invalid data
`Employee_ID` must be unique and not missing.	**`Employee_ID = .`** **Does not account for uniqueness.**
`Gender` must have a value of `F` or `M`.	**`Gender not in ('F','M')`**
`Salary` must be in the range of 24000 – 500000.	**`Salary not between 24000 and 500000`**
`Job_Title` must not be missing.	**`Job_Title = ' '`**
`Country` must have a value of `AU` or `US`.	**`Country not in ('AU','US')`**
`Birth_Date` must occur before **`Hire_Date`**.	**`Birth_Date > Hire_Date`**
`Hire_Date` must have a value of 01/01/1974 or later.	**`Hire_Date < '01JAN1974'd`**

Data Requirements

The following PROC PRINT step accounts for all of the data requirements except the **Employee_ID** being unique.

```
proc print data=orion.nonsales;
   var Employee_ID Gender Salary Job_Title
       Country Birth_Date Hire_Date;
   where Employee_ID = . or
         Gender not in ('F','M') or
         Salary not between 24000 and 500000 or
         Job_Title = ' ' or
         Country not in ('AU','US') or
         Birth_Date > Hire_Date or
         Hire_Date < '01JAN1974'd;
run;
```

The OR operator is used between expressions. Only one expression needs to be true to account for an observation with invalid data.

p108d04

Data Requirements

Sixteen observations need the data cleaned.

Obs	Employee_ID	Gender	Salary	Job_Title	Country	Birth_Date	Hire_Date
2	120104	F	46230	Administration Manager	au	11/05/1954	01/01/1981
4	120106	M	.	Office Assistant II	AU	23/12/1944	01/01/1974
5	120107	F	30475	Office Assistant III	AU	01/02/1978	21/01/1953
9	120111	M	26895	Security Guard II	AU	23/07/1949	.
10	120112	F	26550		AU	17/02/1969	01/07/1990
12	120114	G	31285	Security Manager	AU	08/02/1944	01/01/1974
13	120115	M	2650	Service Assistant I	AU	08/05/1984	01/08/2005
14	.	M	29250	Service Assistant II	AU	13/06/1959	01/02/1980
20	120191	F	2401	Trainee	AU	17/01/1959	01/01/2003
84	120695	M	28180	Warehouse Assistant II	au	13/07/1964	01/07/1989
87	120698	M	26160	Warehouse Assistant I	au	17/05/1954	01/08/1976
101	120723		33950	Corp. Comm. Specialist II	US	10/08/1949	01/01/1974
125	120747	F	43590	Financial Controller I	us	20/06/1974	01/08/1995
197	120994	F	31645	Office Administrator I	us	16/06/1974	01/11/1994
200	120997	F	27420	Shipping Administrator I	us	21/11/1974	01/09/1996
214	121011	M	25735	Service Assistant I	US	11/03/1944	01/01/1968

The FREQ Procedure

The FREQ procedure produces one-way to *n*-way frequency tables.

General form of the FREQ procedure:

```
PROC FREQ DATA=SAS-data-set <NLEVELS>;
      TABLES variable(s);
RUN;
```

- The TABLES statement specifies the frequency tables to produce.
- The NLEVELS option displays a table that provides the number of distinct values for each variable named in the TABLES statement.

The FREQ Procedure

The following PROC FREQ step will show whether there are any invalid values for **`Gender`** and **`Country`**.

```
proc freq data=orion.nonsales;
   tables Gender Country;
run;
```

✎ Without the TABLES statement, PROC FREQ produces a frequency table for each variable.

The FREQ Procedure

Two observations need the data cleaned for **`Gender`** and six observations need the data cleaned for **`Country`**.

The FREQ Procedure

Gender	Frequency	Percent	Cumulative Frequency	Cumulative Percent
F	110	47.01	110	47.01
G	1	0.43	111	47.44
M	123	52.56	234	100.00

Frequency Missing = 1

Country	Frequency	Percent	Cumulative Frequency	Cumulative Percent
AU	33	14.04	33	14.04
US	196	83.40	229	97.45
au	3	1.28	232	98.72
us	3	1.28	235	100.00

The FREQ Procedure

This PROC FREQ step will show whether there are any duplicates for **`Employee_ID`**.

```
proc freq data=orion.nonsales;
   tables Employee_ID;
run;
```

The FREQ Procedure

Partial PROC FREQ Output

The FREQ Procedure

Employee_ID	Frequency	Percent	Cumulative Frequency	Cumulative Percent
120101	1	0.43	1	0.43
120104	1	0.43	2	0.86
120105	1	0.43	3	1.29
120106	1	0.43	4	1.72
120107	1	0.43	5	2.15
120108	2	0.85	7	2.99
120110	1	0.43	8	3.43
120111	1	0.43	9	3.86
120112	1	0.43	10	4.29
120113	1	0.43	11	4.72
121146	1	0.43	232	99.14
121147	1	0.43	233	99.57
121148	1	0.43	234	100.00

Frequency Missing = 1

The NLEVELS Option

If the number of desired distinct values is known, the NLEVELS option can help to determine whether there are any duplicates.

```
proc freq data=orion.nonsales nlevels;
   tables Gender Country Employee_ID;
run;
```

The *NLEVELS option* displays a table that provides the number of distinct values for each variable named in the TABLES statement.

The NLEVELS Option

The Number of Variable Levels table appears before the individual frequency tables.

Partial PROC FREQ Output

The FREQ Procedure

Number of Variable Levels

Variable	Levels	Missing Levels	Nonmissing Levels
Gender	4	1	3
Country	4	0	4
Employee_ID	234	1	233

There are 235 employees but there are only 234 distinct **`Employee_ID`** values. Therefore, there is one duplicate value for **`Employee_ID`**.

Chapter 8: Validating and Cleaning Data

Objectives

- Validate data by using the MEANS procedure with the VAR statement.
- Validate data by using the UNIVARIATE procedure with the VAR statement.

The MEANS Procedure

The MEANS procedure produces summary reports that display descriptive statistics.

General form of the MEANS procedure:

```
PROC MEANS DATA=SAS-data-set <statistics>;
      VAR variable(s);
RUN;
```

- The VAR statement specifies the analysis variables and their order in the results.
- The statistics to display can be specified in the PROC MEANS statement.

The MEANS Procedure

This PROC MEANS step shows default descriptive statistics for **`Salary`**.

```
proc means data=orion.nonsales;
   var Salary;
run;
```

The MEANS Procedure

Analysis Variable : Salary

N	Mean	Std Dev	Minimum	Maximum
234	43954.60	38354.77	2401.00	433800.00

✎ Without the VAR statement, PROC MEANS analyzes all numeric variables in the data set.

p108d06

The MEANS Procedure

By default, the MEANS procedure creates a report with N (number of nonmissing values), MEAN, STDDEV, MIN, and MAX.

For validating data, the following descriptive statistics are beneficial:

- N, number of nonmissing values
- NMISS, number of missing values
- MIN
- MAX

The MEANS Procedure

The following PROC MEANS step shows whether there are any **Salary** values not in the range of 24000 through 500000.

```
proc means data=orion.nonsales n nmiss min max;
   var Salary;
run;
```

The MEANS Procedure

Analysis Variable : Salary

N	N Miss	Minimum	Maximum
234	1	2401.00	433800.00

p108d06

The UNIVARIATE Procedure

The UNIVARIATE procedure produces summary reports that display descriptive statistics.

General form of the UNIVARIATE procedure:

```
PROC UNIVARIATE DATA=SAS-data-set;
      VAR variable(s);
RUN;
```

The VAR statement specifies the analysis variables and their order in the results.

The UNIVARIATE Procedure

The following PROC UNIVARIATE step shows default descriptive statistics for **`Salary`**.

```
proc univariate data=orion.nonsales;
   var Salary;
run;
```

✎ Without the VAR statement, SAS will analyze all numeric variables.

The UNIVARIATE Procedure

The UNIVARIATE procedure can produce the following sections of output:

- Moments
- Basic Statistical Measures
- Tests for Locations
- Quantiles
- Extreme Observations
- Missing Values

For validating data, the Extreme Observations and Missing Values sections are beneficial.

The UNIVARIATE Procedure

Partial PROC UNIVARIATE Output

```
                    Extreme Observations

          -----Lowest----        -----Highest----

           Value      Obs           Value      Obs

            2401       20          163040        1
            2650       13          194885      231
           24025       25          207885       28
           24100       19          268455       29
           24390      228          433800       27


                     Missing Values

                              -----Percent Of-----
      Missing                              Missing
        Value       Count     All Obs         Obs

            .           1        0.43      100.00
```

Chapter 8: Validating and Cleaning Data

Objectives

- Clean data by using the Viewtable window. (Self-Study)
- Clean data by using assignment statements in the DATA step.
- Clean data by using IF-THEN/ELSE statements in the DATA step.

Invalid Data to Clean

The **`orion.nonsales`** data set contains invalid data that needs to be cleaned.

	Employee_ID	First	Last	Gender	Salary	Job_Title	Country	Birth_Date	Hire_Date
1	120101	Patrick	Lu	M	163E3	Director	AU	18/08/1976	01/07/2003
2	120104	Kareen	Billington	F	46230	Administration Manager	au	11/05/1954	01/01/1981
3	120105	Liz	Povey	F	27110	Secretary I	AU	21/12/1974	01/05/1999
4	120106	John	Hornsey	M	.	Office Assistant II	AU	23/12/1944	01/01/1974
5	120107	Sherie	Sheedy	F	30475	Office Assistant III	AU	01/02/1978	21/01/1953
6	120108	Gladys	Gromek	F	27660	Warehouse Assistant II	AU	23/02/1984	01/08/2006
7	120108	Gabriele	Baker	F	26495	Warehouse Assistant I	AU	15/12/1986	01/10/2006
8	120110	Dennis	Entwisle	M	28615	Warehouse Assistant III	AU	20/11/1949	01/11/1979
9	120111	Ubaldo	Spillane	M	26895	Security Guard II	AU	23/07/1949	.
10	120112	Ellis	Glattback	F	26550		AU	17/02/1969	01/07/1990
11	120113	Riu	Horsey	F	26870	Security Guard II	AU	10/05/1944	01/01/1974
12	120114	Jeannette	Buddery	G	31285	Security Manager	AU	08/02/1944	01/01/1974
13	120115	Hugh	Nichollas	M	2650	Service Assistant I	AU	08/05/1984	01/08/2005
14	.	Austen	Ralston	M	29250	Service Assistant II	AU	13/06/1959	01/02/1980
15	120117	Bill	Mccleary	M	31670	Cabinet Maker III	AU	11/09/1964	01/04/1986
16	120118	Darshi	Hartshorn	M	28090	Cabinet Maker II	AU	03/06/1959	01/07/1984

After you validate the data and find the invalid data, the correct data values are needed.

Variable	Obs	Invalid Value	Correct Value
Employee_ID	7	120108	120109
	14	.	120116
Gender	12	G	F
	101		F
Job_Title	10		Security Guard I
Country	2, 84, 87, 125, 197, and 200	au or us	AU or US
Salary	4	.	26960
	13	2650	26500
	20	2401	24015
Hire_Date	5	21/01/1953	21/01/1995
	9	.	01/11/1978
	214	01/01/1968	01/01/1998

Interactively Cleaning Data (Self-Study)

If you are using the SAS windowing environment, the Viewtable window can be used to interactively clean data.

Use the Viewtable window to interactively clean the following five observations:

Variable	Obs	Invalid Value	Correct Value
`Employee_ID`	7	120108	120109
	14	.	120116
`Gender`	12	G	F
	101		F
`Job_Title`	10		Security Guard I

Interactively Cleaning Data (Self-Study)

The Viewtable window enables you to browse, edit, or create SAS data sets.

VIEWTABLE: Orion.Nonsales

	Employee_ID	First	Last	Gender	Salary	Job_Title	Country	Birth_Date	Hire
1	120101	Patrick	Lu	M	163040	Director	AU	18/08/1976	*****
2	120104	Kareen	Billington	F	46230	Administration Manager	au	11/05/1954	*****
3	120105	Liz	Povey	F	27110	Secretary I	AU	21/12/1974	*****
4	120106	John	Hornsey	M	.	Office Assistant II	AU	23/12/1944	*****
5	120107	Sherie	Sheedy	F	30475	Office Assistant III	AU	01/02/1978	*****
6	120108	Gladys	Gromek	F	27660	Warehouse Assistant II	AU	23/02/1984	*****
7	120108	Gabriele	Baker	F	26495	Warehouse Assistant I	AU	15/12/1986	*****
8	120110	Dennis	Entwisle	M	28615	Warehouse Assistant III	AU	20/11/1949	*****
9	120111	Ubaldo	Spillane	M	26895	Security Guard II	AU	23/07/1949	.
10	120112	Ellis	Glattback	F	26550		AU	17/02/1969	*****
11	120113	Riu	Horsey	F	26870	Security Guard II	AU	10/05/1944	*****
12	120114	Jeannette	Buddery	G	31285	Security Manager	AU	08/02/1944	*****
13	120115	Hugh	Nichollas	M	2650	Service Assistant I	AU	08/05/1984	*****
14	.	Austen	Ralston	M	29250	Service Assistant II	AU	13/06/1959	*****
15	120117	Bill	Mccleary	M	31670	Cabinet Maker III	AU	11/09/1964	*****
16	120118	Darshi	Hartshorn	M	28090	Cabinet Maker II	AU	03/06/1959	*****
17	120119	Lal	Elleman	M	30255	Electrician IV	AU	21/12/1969	*****
18	120120	Krishna	Peiris	F	27645	Electrician II	AU	05/05/1944	*****
19	120190	Ivor	Czernezkyi	M	24100	Trainee	AU	05/12/1984	*****
20	120191	Jannene	Graham-Rowe	F	2401	Trainee	AU	17/01/1959	*****
21	120192	Anthony	Nichollas	M	26185	Trainee	AU	08/05/1984	*****
22	120193	Russell	Streit	M	24515	Trainee	AU	06/12/1984	*****
23	120194	Reece	Harwood	M	25985	Trainee	AU	23/09/1984	*****

8.06 Quiz (Self-Study)

- Open the VIEWTABLE window for **`orion.nonsales`**.
- Use the VIEWTABLE window to interactively clean the following observation:

Variable	Obs	Invalid Value	Correct Value
Job_Title	**10**		**Security Guard I**

Programmatically Cleaning Data

The DATA step can be used to programmatically clean the invalid data.

Use the DATA step to clean the following observations:

Variable	Obs	Invalid Value	Correct Value
Country	2, 84, 87, 125, 197, and 200	au or us	AU or US
Salary	4	.	26960
	13	2650	26500
	20	2401	24015
Hire_Date	5	21/01/1953	21/01/1995
	9	.	01/11/1978
	214	01/01/1968	01/01/1998

The Assignment Statement

The *assignment statement* evaluates an expression and assigns the resulting value to a variable.

General form of the assignment statement:

variable = *expression*;

- *variable* names an existing or new variable.
- *expression* is a sequence of operands and operators that form a set of instructions that produce a value.

The Assignment Statement Expression

Operands are

- character constants
- numeric constants
- date constants
- character variables
- numeric variables.

Operators are

- symbols that represent an arithmetic calculation
- SAS functions.

The Assignment Statement Expression

Examples:

```
Salary = 26960;
```

numeric constant

```
Gender = 'F';
```

character constant

```
Hire_Date = '21JAN1995'd;
```

date constant

```
Country = upcase(Country);
```

function

variable

SAS Functions

A SAS *function* is a routine that returns a value that is determined from specified arguments.

The *UPCASE function* converts all letters in an argument to uppercase.

General form of the UPCASE function:

UPCASE(*argument***)**

The *argument* specifies any SAS character expression.

The Assignment Statement

All the values of **`Country`** in the data set **`orion.nonsales`** need to be uppercase.

```
data work.clean;
   set orion.nonsales;
   Country=upcase(Country);
run;
```

PDV

Employee_ID		Job_Title	Country	
120101	...	Director	AU	...

The Assignment Statement

All the values of **`Country`** in the data set **`orion.nonsales`** need to be uppercase.

```
data work.clean;
   set orion.nonsales;
   Country=upcase(Country);
run;
```

PDV

Employee_ID	...	Job_Title	Country	...
120101	...	Director	AU	...

`upcase(au)`

...

The Assignment Statement

All the values of **`Country`** in the data set **`orion.nonsales`** need to be uppercase.

```
data work.clean;
   set orion.nonsales;
   Country=upcase(Country);
run;
```

PDV

Employee_ID		Job_Title	Country	
120104	...	Administration Manager	au	...

...

The Assignment Statement

All the values of **`Country`** in the data set **`orion.nonsales`** need to be uppercase.

```
data work.clean;
   set orion.nonsales;
   Country=upcase(Country);
run;
```

PDV

Employee_ID		Job_Title	Country	
120104	...	Administration Manager	AU	...

`upcase(au)`

The Assignment Statement

```
proc print data=work.clean;
   var Employee_ID Job_Title Country;
run;
```

Partial PROC PRINT Output

Obs	Employee_ ID	Job_Title	Country
84	120695	Warehouse Assistant II	AU
85	120696	Warehouse Assistant I	AU
86	120697	Warehouse Assistant IV	AU
87	120698	Warehouse Assistant I	AU
88	120710	Business Analyst II	US
89	120711	Business Analyst III	US
90	120712	Marketing Manager	US
91	120713	Marketing Assistant III	US

- The assignment statement executed for every observation regardless of whether the value needed to be uppercased or not.

Programmatically Cleaning Data

The DATA step can be used to programmatically clean the invalid data.

Use the DATA step to clean the following observations:

Variable	Obs	Invalid Value	Correct Value
Country	The assignment statement was applied to all observations.		
Salary	4	.	26960
	13	2650	26500
	20		
Hire_Date	5		
	9	.	01/11/1978
	214	01/01/1968	01/01/1998

The assignment statement needs to be applied to specific observations.

8.07 Quiz

Which variable can be used to specifically identify the observations with invalid salary values?

Obs	Employee_ID	Gender	Salary	Job_Title	Country	Birth_Date	Hire_Date
2	120104	F	46230	Administration Manager	au	11/05/1954	01/01/1981
4	120106	M	.	Office Assistant II	AU	23/12/1944	01/01/1974
5	120107	F	30475	Office Assistant III	AU	01/02/1978	21/01/1953
9	120111	M	26895	Security Guard II	AU	23/07/1949	.
10	120112	F	26550		AU	17/02/1969	01/07/1990
12	120114	G	31285	Security Manager	AU	08/02/1944	01/01/1974
13	120115	M	2650	Service Assistant I	AU	08/05/1984	01/08/2005
14	.	M	29250	Service Assistant II	AU	13/06/1959	01/02/1980
20	120191	F	2401	Trainee	AU	17/01/1959	01/01/2003
84	120695	M	28180	Warehouse Assistant II	au	13/07/1964	01/07/1989
87	120698	M	26160	Warehouse Assistant I	au	17/05/1954	01/08/1976
101	120723		33950	Corp. Comm. Specialist II	US	10/08/1949	01/01/1974
125	120747	F	43590	Financial Controller I	us	20/06/1974	01/08/1995
197	120994	F	31645	Office Administrator I	us	16/06/1974	01/11/1994
200	120997	F	27420	Shipping Administrator I	us	21/11/1974	01/09/1996
214	121011	M	25735	Service Assistant I	US	11/03/1944	01/01/1968

Programmatically Cleaning Data

The DATA step can be used to programmatically clean the invalid data.

Use the DATA step to clean the following observations:

Variable	Obs	Invalid Value	Correct Value
Country	2, 84, 87, 125, 197, and 200	au or us	AU or US
Salary	4	.	26960
	13	2650	26500
	20	2401	24015
Hire_Date	5	21/01/1953	21/01/1995
	9	.	01/11/1978
	214	01/01/1968	01/01/1998

IF-THEN Statements

The *IF-THEN statement* executes a SAS statement for observations that meet specific conditions.

General form of the IF-THEN statement:

IF *expression* **THEN** *statement* **;**

- *expression* is a sequence of operands and operators that form a set of instructions that define a condition for selecting observations.
- *statement* is any executable statement such as the assignment statement.

IF-THEN Statements

All the values of **`Salary`** must be in the range of 24000 – 500000.

```
data work.clean;
   set orion.nonsales;
   if Employee_ID=120106 then Salary=26960;
   if Employee_ID=120115 then Salary=26500;
   if Employee_ID=120191 then Salary=24015;
run;
```

PDV

Employee_ID		Salary	Job_Title	
120105	...	27110	Secretary I	...

IF-THEN Statements

All the values of **`Salary`** must be in the range of 24000 – 500000.

FALSE

```
data work.clean;
   set orion.nonsales;
   if Employee_ID=120106 then Salary=26960;
   if Employee_ID=120115 then Salary=26500;
   if Employee_ID=120191 then Salary=24015;
run;
```

PDV

Employee_ID		Salary	Job_Title	
120105	...	27110	Secretary I	...

...

IF-THEN Statements

All the values of **`Salary`** must be in the range of 24000 – 500000.

FALSE

```
data work.clean;
   set orion.nonsales;
   if Employee_ID=120106 then Salary=26960;
   if Employee_ID=120115 then Salary=26500;
   if Employee_ID=120191 then Salary=24015;
run;
```

PDV

Employee_ID		Salary	Job_Title	
120105	...	27110	Secretary I	...

...

IF-THEN Statements

All the values of **`Salary`** must be in the range of 24000 – 500000.

FALSE

```
data work.clean;
   set orion.nonsales;
   if Employee_ID=120106 then Salary=26960;
   if Employee_ID=120115 then Salary=26500;
   if Employee_ID=120191 then Salary=24015;
run;
```

PDV

Employee_ID		Salary	Job_Title	
120105	...	27110	Secretary I	...

...

IF-THEN Statements

All the values of **`Salary`** must be in the range of 24000 – 500000.

```
data work.clean;
   set orion.nonsales;
   if Employee_ID=120106 then Salary=26960;
   if Employee_ID=120115 then Salary=26500;
   if Employee_ID=120191 then Salary=24015;
run;
```

PDV

Employee_ID		Salary	Job_Title	
120106	...	.	Office Assistant II	...

IF-THEN Statements

All the values of **`Salary`** must be in the range of 24000 – 500000.

TRUE

```
data work.clean;
   set orion.nonsales;
   if Employee_ID=120106 then Salary=26960;
   if Employee_ID=120115 then Salary=26500;
   if Employee_ID=120191 then Salary=24015;
run;
```

PDV

Employee_ID		Salary	Job_Title	
120106	...	26960	Office Assistant II	...

...

IF-THEN Statements

All the values of **`Salary`** must be in the range of 24000 – 500000.

FALSE

```
data work.clean;
   set orion.nonsales;
   if Employee_ID=120106 then Salary=26960;
   if Employee_ID=120115 then Salary=26500;
   if Employee_ID=120191 then Salary=24015;
run;
```

PDV

Employee_ID		Salary	Job_Title	
120106	...	26960	Office Assistant II	...

...

IF-THEN Statements

All the values of **`Salary`** must be in the range of 24000 – 500000.

FALSE

```
data work.clean;
   set orion.nonsales;
   if Employee_ID=120106 then Salary=26960;
   if Employee_ID=120115 then Salary=26500;
   if Employee_ID=120191 then Salary=24015;
run;
```

PDV

Employee_ID	...	Salary	Job_Title	...
120106	...	26960	Office Assistant II	...

IF-THEN Statements

When an IF expression is TRUE in this IF-THEN statement series, there is no reason to check the remaining IF-THEN statements when checking **Employee_ID**.

TRUE

```
data work.clean;
   set orion.nonsales;
   if Employee_ID=120106 then Salary=26960;
   if Employee_ID=120115 then Salary=26500;
   if Employee_ID=120191 then Salary=24015;
run;
```

The word ELSE can be placed before the word IF, causing SAS to execute conditional statements until it encounters the first true statement.

IF-THEN/ELSE Statements

All the values of **`Salary`** must be in the range of 24000 – 500000.

```
data work.clean;
   set orion.nonsales;
   if Employee_ID=120106 then Salary=26960;
   else if Employee_ID=120115 then Salary=26500;
   else if Employee_ID=120191 then Salary=24015;
run;
```

PDV

Employee_ID		Salary	Job_Title	
120106	...	.	Office Assistant II	...

p108d07

IF-THEN/ELSE Statements

All the values of **`Salary`** must be in the range of 24000 – 500000.

TRUE

```
data work.clean;
   set orion.nonsales;
   if Employee_ID=120106 then Salary=26960;
   else if Employee_ID=120115 then Salary=26500;
   else if Employee_ID=120191 then Salary=24015;
run;
```

SKIP

PDV

Employee_ID		Salary	Job_Title	
120106	...	26960	Office Assistant II	...

Programmatically Cleaning Data

The DATA step can be used to programmatically clean the invalid data.

Use the DATA step to clean the following observations:

Variable	Obs	Invalid Value	Correct Value
Country	2, 84, 87, 125, 197, and 200	au or us	AU or US
Salary	4	.	26960
	13	2650	26500
	20	2401	24015
Hire_Date	5	21/01/1953	21/01/1995
	9	.	01/11/1978
	214	01/01/1968	01/01/1998

IF-THEN/ELSE Statements

All the values of **`Hire_Date`** must have a value of 01/01/1974 or later.

```
data work.clean;
  set orion.nonsales;
  Country=upcase(Country);
  if Employee_ID=120106 then Salary=26960;
  else if Employee_ID=120115 then Salary=26500;
  else if Employee_ID=120191 then Salary=24015;
  else if Employee_ID=120107 then
          Hire_Date='21JAN1995'd;
  else if Employee_ID=120111 then
          Hire_Date='01NOV1978'd;
  else if Employee_ID=121011 then
          Hire_Date='01JAN1998'd;
run;
```

p108d07

Chapter Review

1. What procedures can be used to detect invalid data?

2. What happens when SAS encounters a data error?

3. Why would you need a SAS date constant?

4. How can you clean invalid data?

5. What symbol is required in an assignment statement?

6. Why would you use IF-THEN/ELSE statements instead of IF-THEN statements?