

STAT604

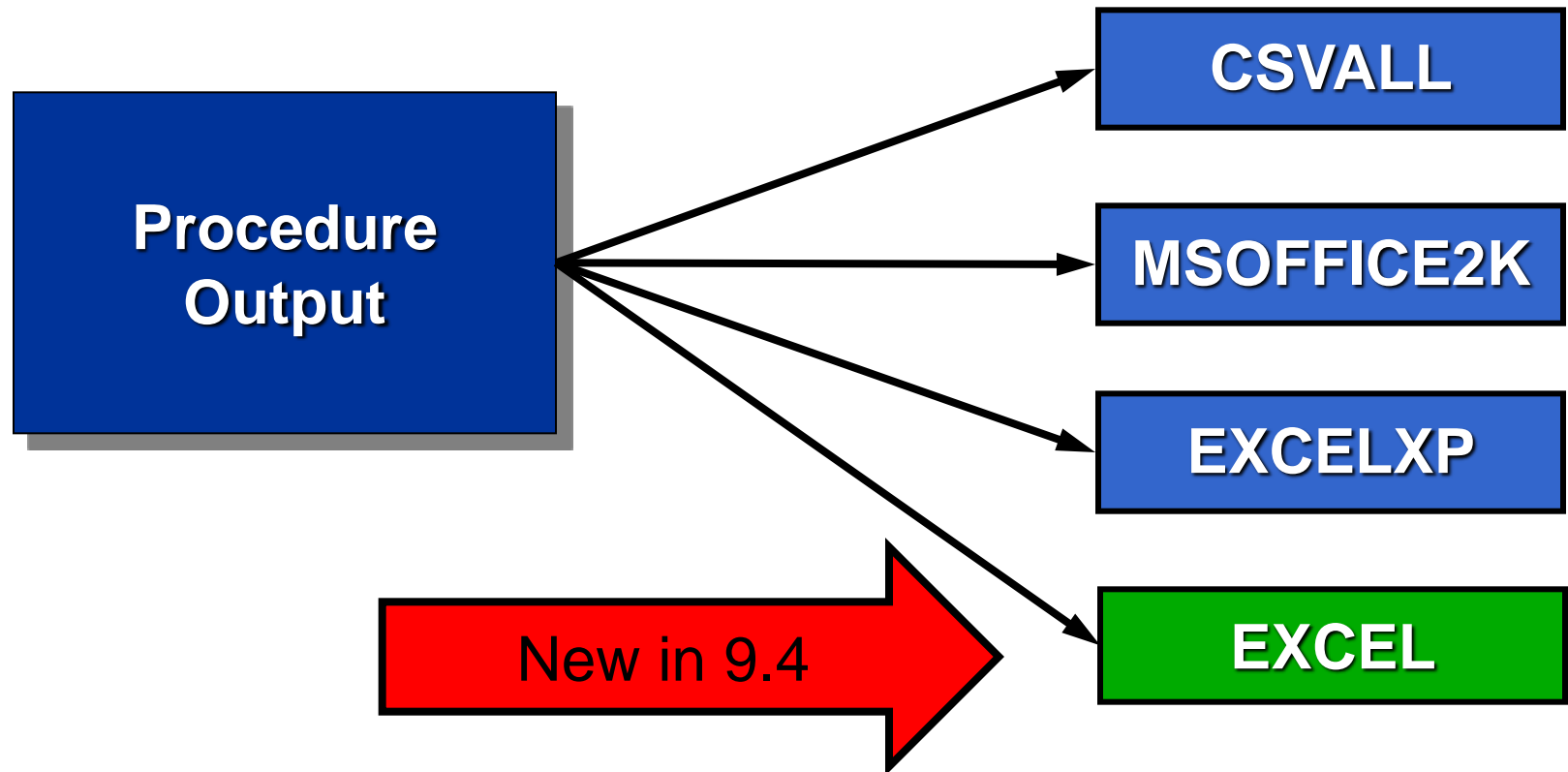
Lesson SAS 03

Portions Copyright © 2009 SAS Institute Inc., Cary, NC, USA. All rights reserved. Reproduced with permission of SAS Institute Inc., Cary, NC, USA. SAS Institute Inc. makes no warranties with respect to these materials and disclaims all liability therefor.

**THE
POWER
TO KNOW®**

Destinations Used with Excel

The following destinations create files that can be opened in Excel.



Destinations Used with Excel

Destination	Type of File	Viewed In
CSVALL	Comma-Separated Value	Editor or Microsoft Excel
MSOFFICE2K	Hypertext Markup Language	Web Browser or Microsoft Word or Microsoft Excel
EXCELXP	Extensible Markup Language	Microsoft Excel
EXCEL	True Excel File	Microsoft Excel

CSVALL Destination

```
ods csvall file='myexcel.csv';
```

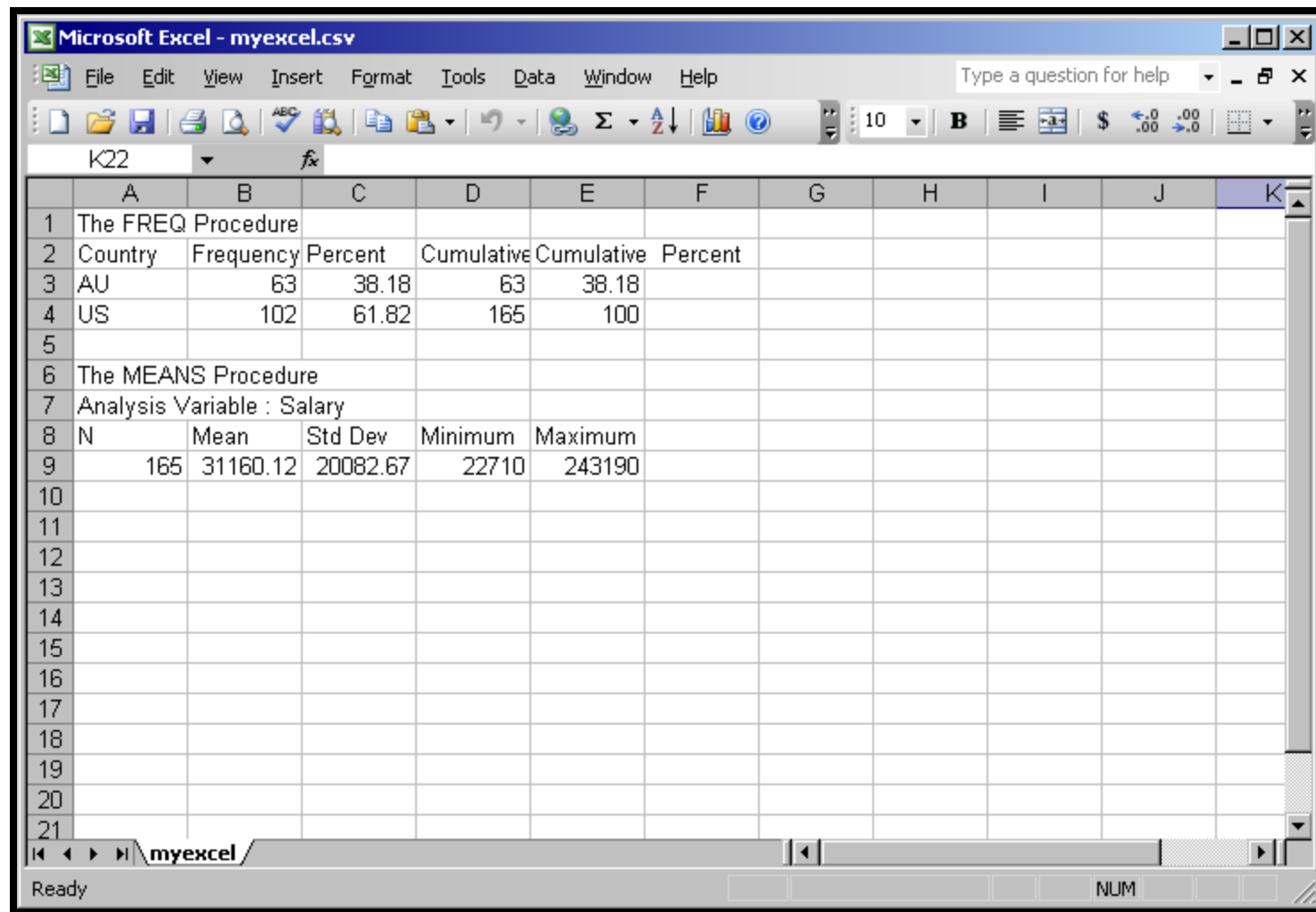
```
proc freq data=orion.sales;  
    tables Country;  
run;
```

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

```
ods csvall close;
```

CSVALL Destination

CSVALL does not include any style information.



Microsoft Excel - myexcel.csv

Type a question for help

File Edit View Insert Format Tools Data Window Help

10 B \$.00 .00

K22 fx

	A	B	C	D	E	F	G	H	I	J	K
1	The FREQ Procedure										
2	Country	Frequency	Percent	Cumulative	Cumulative	Percent					
3	AU	63	38.18	63	38.18						
4	US	102	61.82	165	100						
5											
6	The MEANS Procedure										
7	Analysis Variable : Salary										
8	N	Mean	Std Dev	Minimum	Maximum						
9	165	31160.12	20082.67	22710	243190						
10											
11											
12											
13											
14											
15											
16											
17											
18											
19											
20											
21											

myexcel

Ready NUM

MSOFFICE2K Destination

```
ods msoffice2k file='myexcel.html';
```

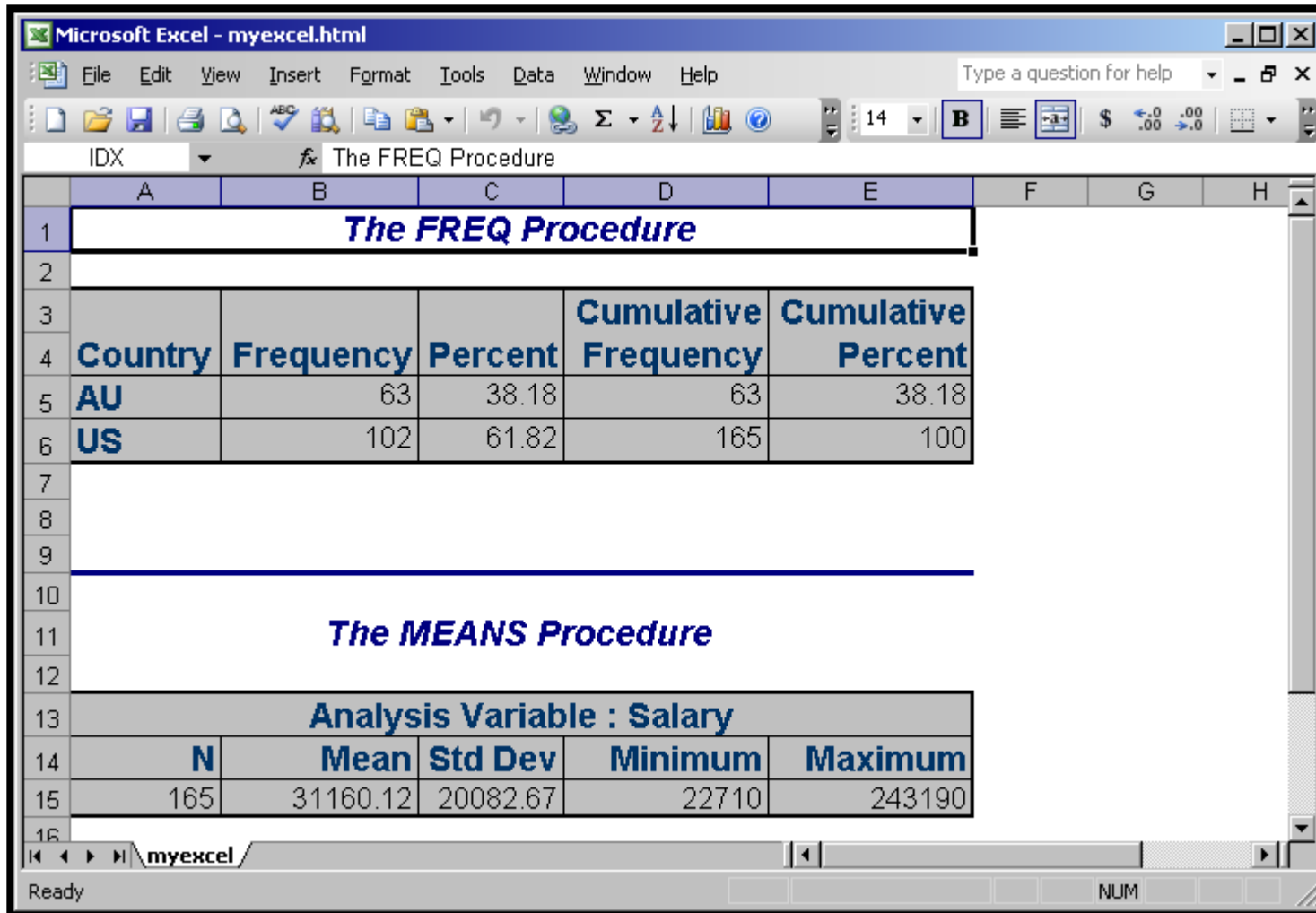
```
proc freq data=orion.sales;  
    tables Country;  
run;
```

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

```
ods msoffice2k close;
```

MSOFFICE2K Destination

MSOFFICE2K keeps the style information including spanning headers.



Microsoft Excel - myexcel.html

File Edit View Insert Format Tools Data Window Help

Type a question for help

14 B

IDX The FREQ Procedure

	A	B	C	D	E	F	G	H
1	The FREQ Procedure							
2								
3				Cumulative	Cumulative			
4	Country	Frequency	Percent	Frequency	Percent			
5	AU	63	38.18	63	38.18			
6	US	102	61.82	165	100			
7								
8								
9								
10								
11								
12								
13								
14								
15								
16								

myexcel

Ready

NUM

EXCELXP Destination

```
ods tagsets.excelxp file='myexcel.xml';
```

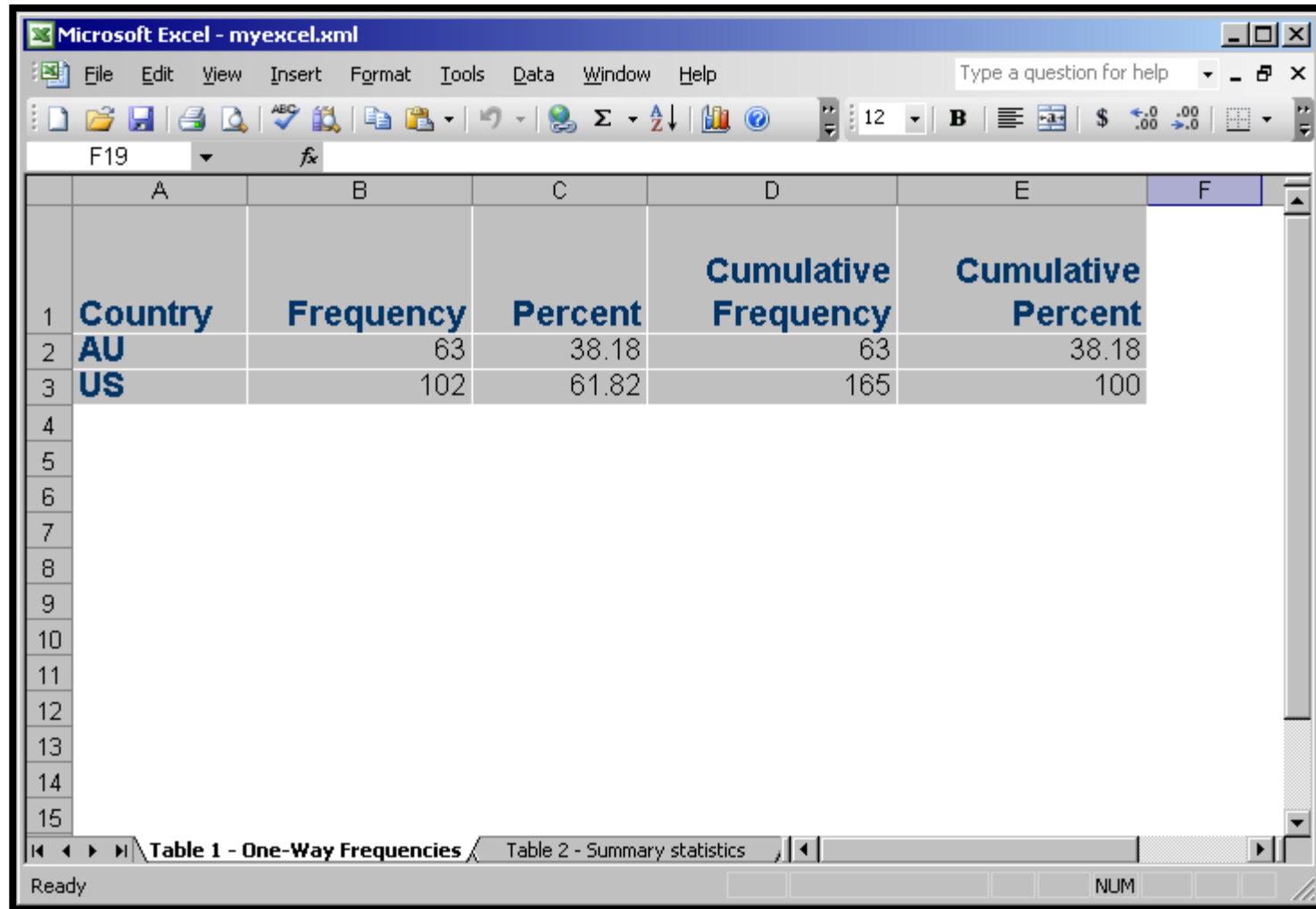
```
proc freq data=orion.sales;  
    tables Country;  
run;
```

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

```
ods tagsets.excelxp close;
```


EXCELXP Destination

EXCELXP keeps the style information and each procedure is a separate sheet.



Microsoft Excel - myexcel.xml

Type a question for help

File Edit View Insert Format Tools Data Window Help

12 B \$.00 .00

F19 fx

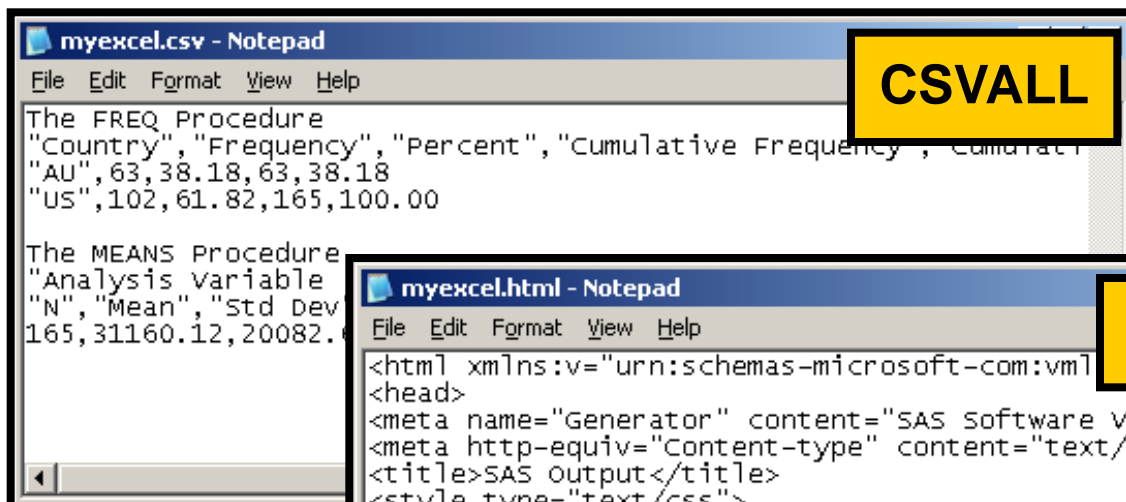
	A	B	C	D	E	F
1	Country	Frequency	Percent	Cumulative Frequency	Cumulative Percent	
2	AU	63	38.18	63	38.18	
3	US	102	61.82	165	100	
4						
5						
6						
7						
8						
9						
10						
11						
12						
13						
14						
15						

Table 1 - One-Way Frequencies Table 2 - Summary statistics

Ready NUM

Keep in Mind

The file you are creating is not an Excel file.

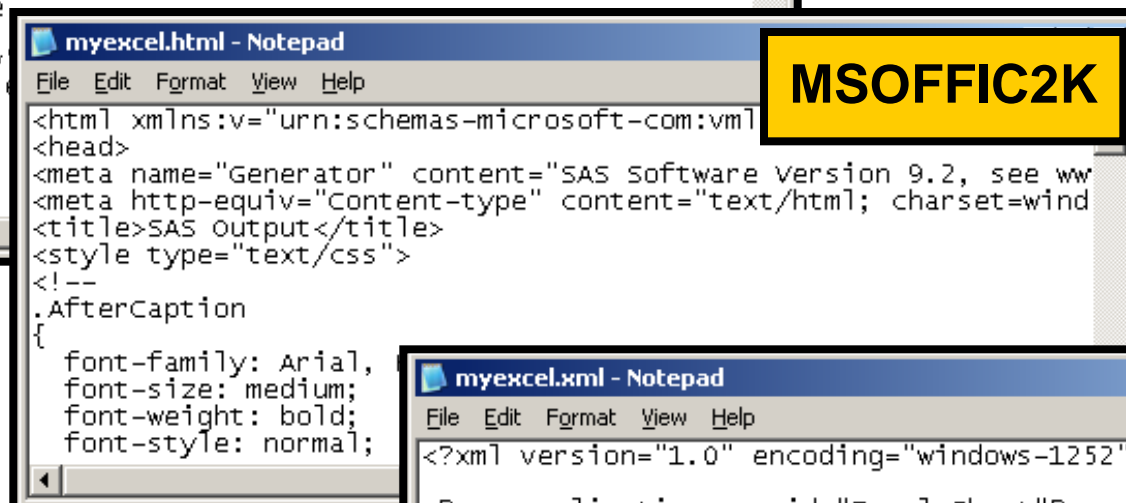


myexcel.csv - Notepad

```
File Edit Format View Help
The FREQ Procedure
"Country", "Frequency", "Percent", "Cumulative Frequency", "Cumulative Percent"
"AU", 63, 38.18, 63, 38.18
"US", 102, 61.82, 165, 100.00

The MEANS Procedure
"Analysis Variable"
"N", "Mean", "Std Dev"
165, 31160.12, 20082.12
```

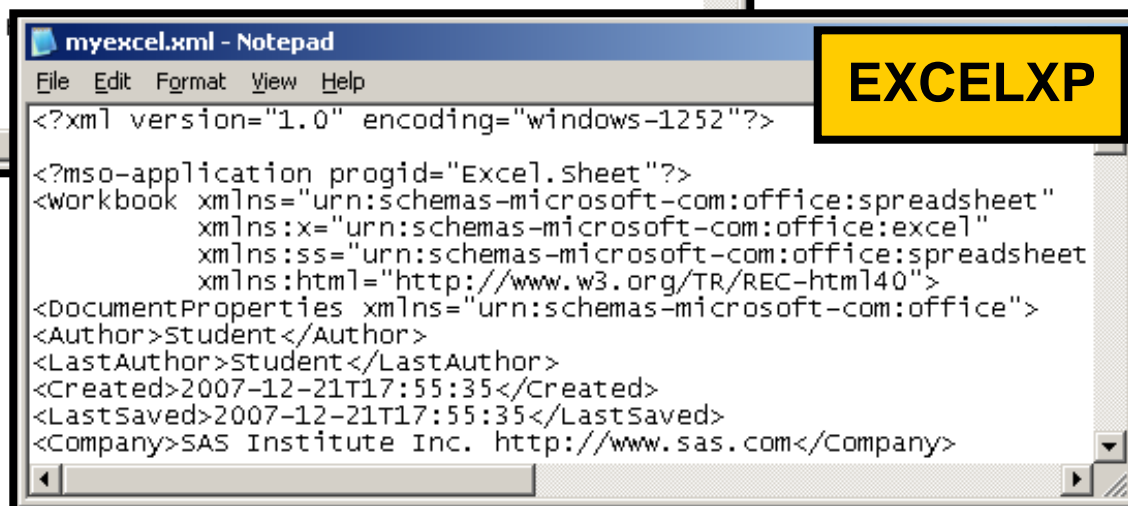
CSVALL



myexcel.html - Notepad

```
File Edit Format View Help
<html xmlns:v="urn:schemas-microsoft-com:xml"
<head>
<meta name="Generator" content="SAS Software Version 9.2, see www.sas.com"
<meta http-equiv="Content-type" content="text/html; charset=windows-1252"
<title>SAS Output</title>
<style type="text/css">
<!--
.AfterCaption
{
font-family: Arial,
font-size: medium;
font-weight: bold;
font-style: normal;
}
```

MSOFFIC2K



myexcel.xml - Notepad

```
File Edit Format View Help
<?xml version="1.0" encoding="windows-1252"?>

<?mso-application progid="Excel.Sheet"?>
<workbook xmlns="urn:schemas-microsoft-com:office:spreadsheet"
xmlns:x="urn:schemas-microsoft-com:office:excel"
xmlns:ss="urn:schemas-microsoft-com:office:spreadsheet"
xmlns:html="http://www.w3.org/TR/REC-html40">
<DocumentProperties xmlns="urn:schemas-microsoft-com:office:office">
<Author>Student</Author>
<LastAuthor>Student</LastAuthor>
<Created>2007-12-21T17:55:35</Created>
<LastSaved>2007-12-21T17:55:35</LastSaved>
<Company>SAS Institute Inc. http://www.sas.com</Company>
```

EXCELXP

EXCEL Destination

```
ods excel file='myexcel.xlsx';
```

```
proc freq data=orion.sales;  
  tables Country;  
run;
```

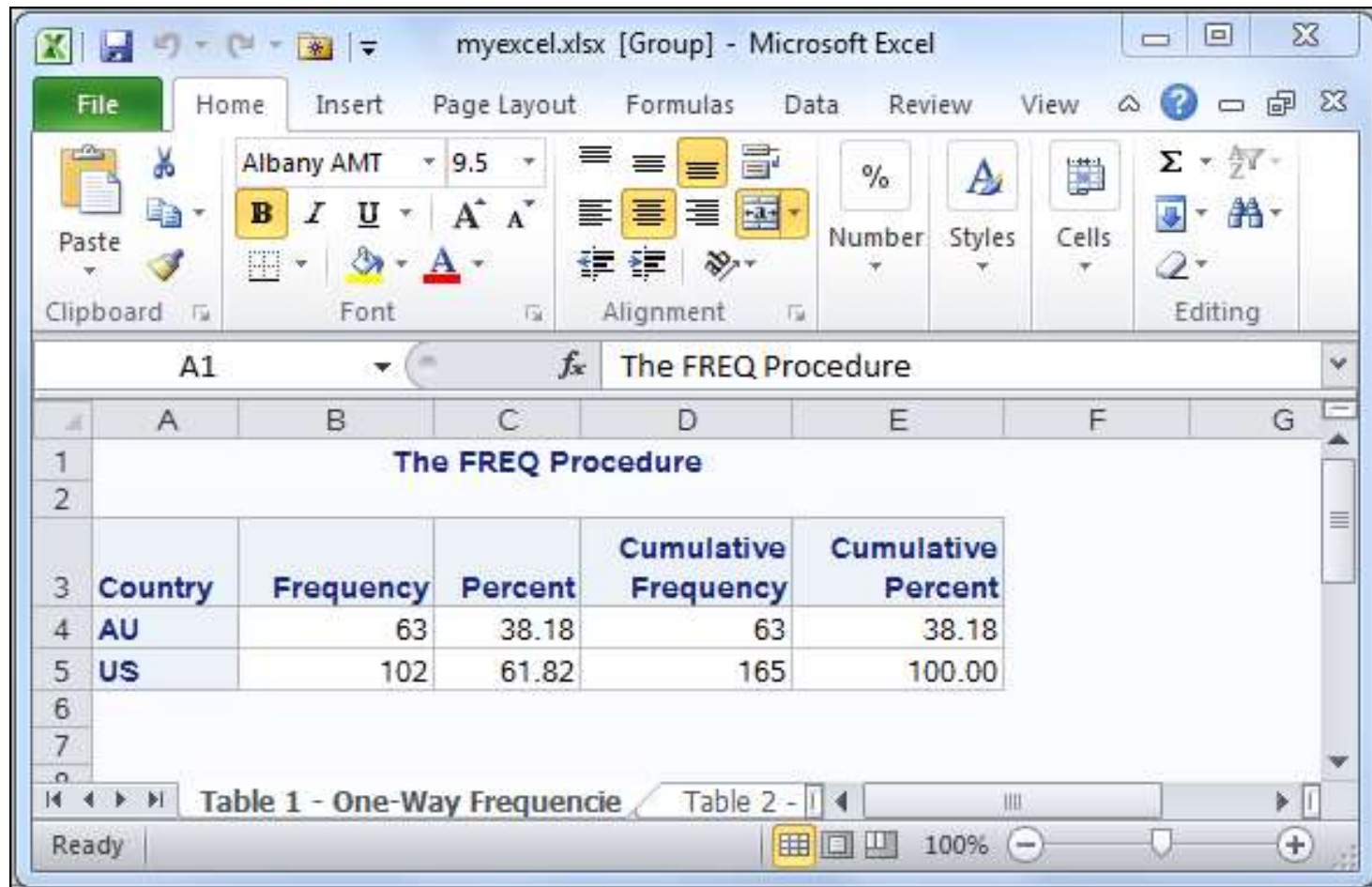
```
proc means data=orion.sales;  
  var Salary;  
run;
```

```
ods excel close;
```

**ODS EXCEL produces a native XLSX file.
ODS EXCEL supports SAS graphics in the
output.**

EXCEL Destination

EXCEL functions very much like the EXCELP destination.



The screenshot shows the Microsoft Excel 2010 interface. The title bar reads "myexcel.xlsx [Group] - Microsoft Excel". The ribbon is set to the "Formulas" tab. The active cell is A1, which contains the text "The FREQ Procedure". Below this, a table is displayed with the following data:

	A	B	C	D	E	F	G
1	The FREQ Procedure						
2							
3	Country	Frequency	Percent	Cumulative Frequency	Cumulative Percent		
4	AU	63	38.18	63	38.18		
5	US	102	61.82	165	100.00		
6							
7							

The status bar at the bottom indicates "Ready" and shows the zoom level at 100%.

Assigning a Fileref

You can use the *FILENAME* statement to assign a file reference name (fileref) to an external file.

General form of the FILENAME statement:

FILENAME *fileref* 'external-file' <options>;

Rules for naming a fileref:

- The name must be 8 characters or less.
- The name must begin with a letter or underscore.
- The remaining characters must be letters, numerals, or underscores.

Assigning and Using a Fileref

Windows Example:

Assigning

```
filename pdfrep1 's:\workshop\Assign6.pdf';
```

```
FILENAME OUT FTP '/home/ftpas/dialog.txt'  
    host='ftpsrv.tamu.edu'  
    user='ftpas' pass='ftppass' lrecl=2437  
    rcmd='site umask 022'  
    /* Set permissions to -rw-r--r-- */  
;
```

Using

```
ods pdf file = pdfrep1 notoc;
```

Reviewing Concepts

- Libref (libname) = alias to a collection of tables
- Fileref (filename) = alias to a single file
- Raw text files are not considered tables and cannot be accessed through a libref
- Both librefs and filerefs can be read from and written to

Chapter 5: Reading SAS Data Sets



5.1 Introduction to Reading Data

5.2 Using SAS Data as Input

5.3 Subsetting Observations and Variables

5.4 Adding Permanent Attributes

Chapter 5: Reading SAS Data Sets

5.1 Introduction to Reading Data

5.2 Using SAS Data as Input

5.3 Subsetting Observations and Variables

5.4 Adding Permanent Attributes

Objectives

- Define the concept of reading from a data source to create a SAS data set.
- Define the business scenario that will be used when reading from a SAS data set, an Excel worksheet, and a raw data file.

Business Scenario

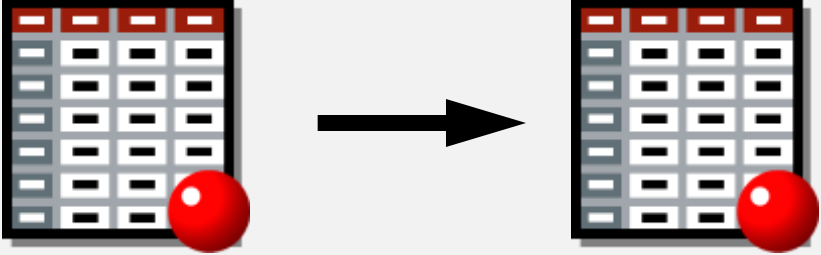
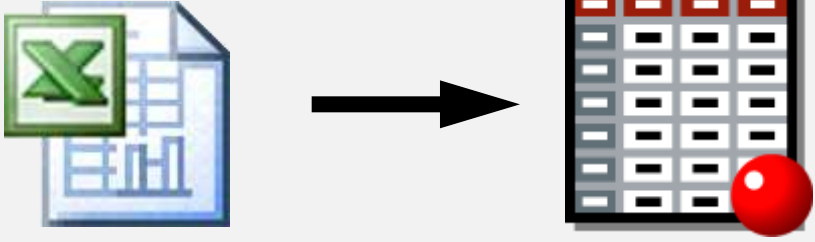
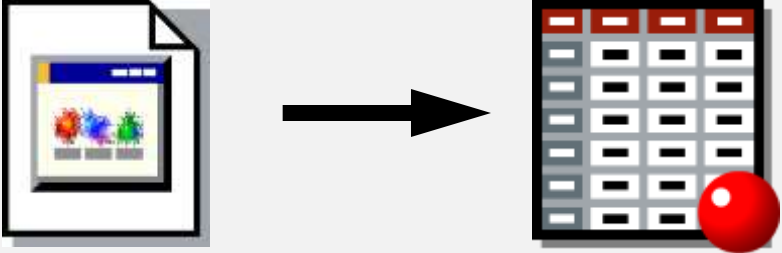
An existing data source contains information on Orion Star sales employees from Australia and the United States.

A new SAS data set needs to be created that contains a subset of this existing data source.

This new SAS data set must contain the following:

- only the employees from Australia who are Sales Representatives
- the employee's first name, last name, salary, job title, and hired date
- labels and formats in the descriptor portion

Business Scenario

<p>Reading SAS Data Sets</p>	
<p>Reading Excel Worksheets</p>	
<p>Reading Delimited Raw Data Files</p>	

Business Scenario

Reading SAS Data Sets	<pre>libname _____; data _____; set _____; ... run;</pre>
Reading Excel Worksheets	<pre>libname _____; data _____; set _____; ... run;</pre>
Reading Delimited Raw Data Files	<pre>data _____; infile _____; input _____; ... run;</pre>

Chapter 5: Reading SAS Data Sets



5.1 Introduction to Reading Data

5.2 Using SAS Data as Input

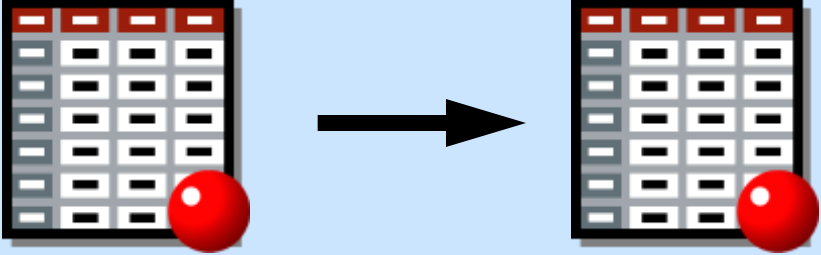
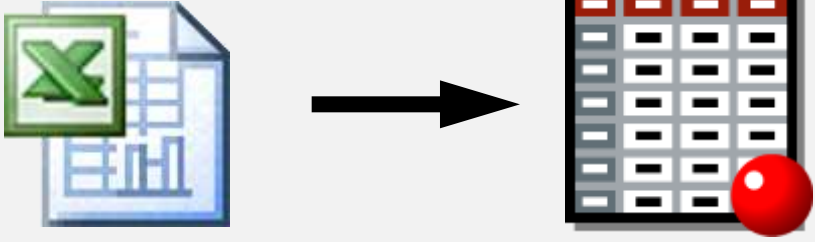
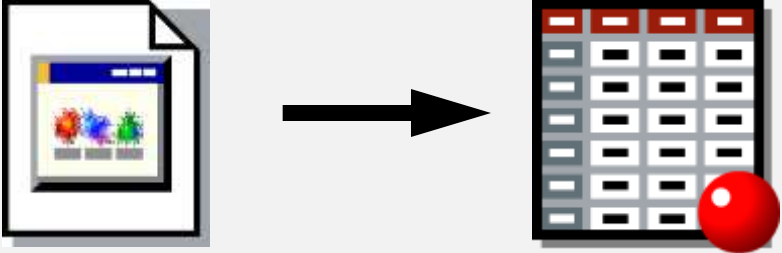
5.3 Subsetting Observations and Variables

5.4 Adding Permanent Attributes

Objectives

- Use the DATA step to create a SAS data set from an existing SAS data set.

Business Scenario

<p>Reading SAS Data Sets</p>	
<p>Reading Excel Worksheets</p>	
<p>Reading Delimited Raw Data Files</p>	

Business Scenario

Reading SAS Data Sets	<pre>libname _____; data _____; set _____; ... run;</pre>
Reading Excel Worksheets	<pre>libname _____; data _____; set _____; ... run;</pre>
Reading Delimited Raw Data Files	<pre>data _____; infile _____; input _____; ... run;</pre>

Business Scenario Syntax

Use the following statements to complete the scenario:

```
LIBNAME libref 'SAS-data-library';
```

```
DATA output-SAS-data-set;
```

```
    SET input-SAS-data-set;
```

```
    WHERE where-expression;
```

```
    KEEP variable-list;
```

```
    LABEL variable = 'label'
```

```
           variable = 'label'
```

```
           variable = 'label';
```

```
    FORMAT variable(s) format;
```

```
RUN;
```

Business Scenario Syntax

Use the following statements to complete the scenario:

```
LIBNAME libref 'SAS-data-library';
```

```
DATA output-SAS-data-set;
```

```
SET input-SAS-data-set;
```

```
WHERE where-expression;
```

```
KEEP variable-list;
```

```
LABEL variable = 'label'
```

```
variable = 'label'
```

```
variable = 'label';
```

```
FORMAT variable(s) format;
```

```
RUN;
```

Part 1

Part 2

Part 3

The LIBNAME Statement (Review)

A library reference name (libref) is needed if a permanent data set is being read or created.

```
LIBNAME libref 'SAS-data-library';
```

```
DATA output-SAS-data-set;  
    SET input-SAS-data-set;  
    <additional SAS statements>  
RUN;
```

The *LIBNAME statement* assigns a libref to a SAS data library.

The DATA Statement

The *DATA statement* begins a DATA step and provides the name of the SAS data set being created.

```
LIBNAME libref 'SAS-data-library';
```

```
DATA output-SAS-data-set;
```

```
    SET input-SAS-data-set;
```

```
    <additional SAS statements>
```

```
RUN;
```

The DATA statement can create temporary or permanent data sets.

The SET Statement

The *SET statement* reads observations from a SAS data set for further processing in the DATA step.

```
LIBNAME libref 'SAS-data-library';  
  
DATA output-SAS-data-set;  
    SET input-SAS-data-set;  
    <additional SAS statements>  
RUN;
```

- By default, the SET statement reads all observations and all variables from the input data set.
- The SET statement can read temporary or permanent data sets.

Business Scenario Part 1

Create a temporary SAS data set named **Work.subset1** from the permanent SAS data set named **orion.sales**.

```
libname orion 's:\workshop';  
  
data work.subset1;  
    set orion.sales;  
run;
```

Partial SAS Log

```
9    data work.subset1;  
10       set orion.sales;  
11    run;
```

Both data sets contain 165 observations and 9 variables

NOTE: There were 165 observations read from the data set ORION.SALES.
NOTE: The data set WORK.SUBSET1 has 165 observations and 9 variables.

Business Scenario Part 1

```
proc print data=work.subset1;  
run;
```

Partial PROC PRINT Output

Obs	Employee_ID	First_ Name	Last_Name	Gender	Salary	Job_Title	Country	Birth_ Date	Hire_ Date
1	120102	Tom	Zhou	M	108255	Sales Manager	AU	3510	10744
2	120103	Wilson	Dawes	M	87975	Sales Manager	AU	-3996	5114
3	120121	Irenie	Elvish	F	26600	Sales Rep. II	AU	-5630	5114
4	120122	Christina	Ngan	F	27475	Sales Rep. II	AU	-1984	6756
5	120123	Kimiko	Hotstone	F	26190	Sales Rep. I	AU	1732	9405
6	120124	Lucian	Daymond	M	26480	Sales Rep. I	AU	-233	6999
7	120125	Fong	Hofmeister	M	32040	Sales Rep. IV	AU	-1852	6999
8	120126	Satyakam	Denny	M	26780	Sales Rep. II	AU	10490	17014
9	120127	Sharryn	Clarkson	F	28100	Sales Rep. II	AU	6943	14184
10	120128	Monica	Kletschkus	F	30890	Sales Rep. IV	AU	9691	17106
11	120129	Alvin	Roebuck	M	30070	Sales Rep. III	AU	1787	9405
12	120130	Kevin	Lyon	M	26955	Sales Rep. I	AU	9114	16922
13	120131	Marinus	Surawski	M	26910	Sales Rep. I	AU	7207	15706
14	120132	Fancine	Kaiser	F	28525	Sales Rep. III	AU	-3923	6848
15	120133	Petrea	Soltau	F	27440	Sales Rep. II	AU	9608	17075

Poll 

Quiz

5.02 Poll

The DATA step reads a temporary SAS data set to create a permanent SAS data set.

- ☐ True
- ☐ False

```
data work.mycustomers;  
    set orion.customer;  
run;  
  
proc print data=work.mycustomers;  
    var Customer_ID Customer_Name  
        Customer_Address;  
run;
```

5.02 Poll – Correct Answer

The DATA step reads a temporary SAS data set to create a permanent SAS data set.

- ☐ True
- ☒ False

```
data work.mycustomers;  
    set orion.customer;  
run;  
  
proc print data=work.mycustomers;  
    var Customer_ID Customer_Name  
        Customer_Address;  
run;
```

A large, stylized graphic featuring the words "Question & Answer" in a bold, sans-serif font. The word "Question" is in blue, and the ampersand "&" is in yellow. The word "Answer" is also in blue. The text is centered within a large, horizontal oval shape. The oval has a thick yellow border with a blue outline. The background is white.

Question & Answer

Chapter 5: Reading SAS Data Sets



5.1 Introduction to Reading Data

5.2 Using SAS Data as Input

5.3 Subsetting Observations and Variables

5.4 Adding Permanent Attributes

Objectives

- Subset observations by using the WHERE statement.
- Subset variables by using the DROP and KEEP statements.

Business Scenario Syntax

Use the following statements to complete the scenario:

```
LIBNAME libref 'SAS-data-library';
```

```
DATA output-SAS-data-set;  
  SET input-SAS-data-set;
```

```
  WHERE where-expression;
```

```
  KEEP variable-list;
```

```
  LABEL variable = 'label'  
         variable = 'label'  
         variable = 'label';
```

```
  FORMAT variable(s) format;
```

```
RUN;
```

Part 1

Part 2

Part 3

Subsetting Observations and Variables

By default, the SET statement reads **all observations** and **all variables** from the input data set.

```
9    data work.subset1;  
10       set orion.sales;  
11    run;
```

NOTE: There were 165 observations read from the data set ORION.SALES.

NOTE: The data set WORK.SUBSET1 has 165 observations and 9 variables.

By adding statements to the DATA step, the number of observations and variables can be reduced.

NOTE: The data set WORK.SUBSET1 has 61 observations and 5 variables.

The WHERE Statement

The *WHERE* statement subsets observations that meet a particular condition.

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.

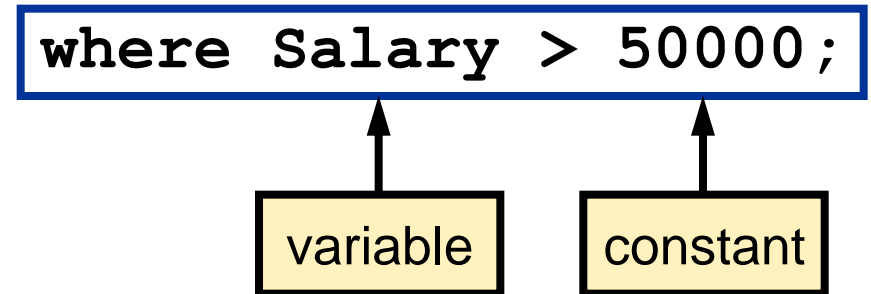
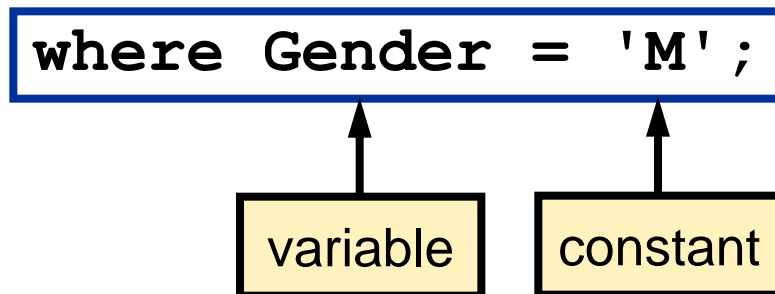
Operands

A *constant operand* is a fixed value.

- Character values must be enclosed in quotation marks and are case sensitive.
- Numeric values do not use quotation marks.

A *variable operand* must be a variable coming from an input data set.

Examples:



Comparison Operators

Comparison operators compare a variable with a value or with another variable.

Symbol	Mnemonic	Definition
=	EQ	equal to
\neq $\neg=$ $\sim=$	NE	not equal to
>	GT	greater than
<	LT	less than
\geq	GE	greater than or equal
\leq	LE	less than or equal
	IN	equal to one of a list

Comparison Operators

Examples:

```
where Gender = 'M' ;
```

```
where Gender eq ' ' ;
```

```
where Salary ne . ;
```

```
where Salary >= 50000 ;
```

```
where Country in ('AU', 'US') ;
```

```
where Country in ('AU' 'US') ;
```

Values must be separated by commas or blanks.

Arithmetic Operators

Arithmetic operators indicate that an arithmetic calculation is performed.

Symbol	Definition
**	exponentiation
*	multiplication
/	division
+	addition
-	subtraction

Arithmetic Operators

Examples:

```
where Salary / 12 < 6000;
```

```
where Salary / 12 * 1.10 >= 7500;
```

```
where (Salary / 12 ) * 1.10 >= 7500;
```

```
where Salary + Bonus <= 10000;
```

Logical Operators

Logical operators combine or modify expressions.

Symbol	Mnemonic	Definition
&	AND	logical and
	OR	logical or
\wedge \neg \sim	NOT	logical not

Logical Operators

Examples:

```
where Gender ne 'M' and Salary >=50000;
```

```
where Gender ne 'M' or Salary >= 50000;
```

```
where Country = 'AU' or Country = 'US' ;
```

```
where Country not in ('AU' 'US') ;
```


Poll

Quiz



5.03 Quiz

Which WHERE statement correctly subsets the numeric values for May, June, or July and missing character names?

a. `where Months in (5-7)
and Names = .;`

b. `where Months in (5,6,7)
and Names = ' ';`

c. `where Months in ('5','6','7')
and Names = ' .';`

5.03 Quiz – Correct Answer

Which WHERE statement correctly subsets the numeric values for May, June, or July and missing character names?

a. `where Months in (5-7)
and Names = .;`

b. `where Months in (5,6,7)
and Names = ' ';`

c. `where Months in ('5','6','7')
and Names = '. ';`

Special WHERE Operators

Special WHERE operators are operators that can only be used in a where-expression.

Symbol	Mnemonic	Definition
	BETWEEN-AND	an inclusive range
	IS NULL	missing value
	IS MISSING	missing value
?	CONTAINS	a character string
	LIKE	a character pattern

BETWEEN-AND Operator

The *BETWEEN-AND operator* selects observations in which the value of a variable falls within an inclusive range of values.

Examples:

```
where salary between 50000 and 100000;
```

```
where salary not between 50000 and 100000;
```

Equivalent Expressions:

```
where salary between 50000 and 100000;
```

```
where 50000 <= salary <= 100000;
```

IS NULL and IS MISSING Operators

The *IS NULL* and *IS MISSING* operators select observations in which the value of a variable is missing.

- The operator can be used for both character and numeric variables. (Use NULL with databases.)
- You can combine the NOT logical operator with IS NULL or IS MISSING to select nonmissing values.
- The MISSING function can be used with both subsetting where and if statements.

Examples:

```
where Employee_ID is null;
```

```
where Employee_ID is missing;
```

```
where missing(Employee_ID) ;
```

CONTAINS Operator

The *CONTAINS (?) operator* selects observations that include the specified substring.

- The position of the substring within the variable's values is not important.
- The operator is case sensitive when you make comparisons.

Example:

```
where Job_Title contains 'Rep' ;
```

Poll

Quiz



5.04 Quiz

Which value will not be returned based on the WHERE statement?

- a. Office Rep
- b. Sales Rep. IV
- c. service rep III
- d. Representative

```
where Job_Title contains 'Rep' ;
```

5.04 Quiz – Correct Answer

Which value will not be returned based on the WHERE statement?

- a. Office Rep
- b. Sales Rep. IV
- ☒ c. service rep III
- d. Representative

```
where Job_Title contains 'Rep' ;
```

LIKE Operator

The *LIKE* operator selects observations by comparing character values to specified patterns.

There are two special characters available for specifying a pattern:

- A percent sign (%) replaces any number of characters.
- An underscore (_) replaces one character.

Consecutive underscores can be specified.

A percent sign and an underscore can be specified in the same pattern.

The operator is case sensitive.

LIKE Operator

Examples:

```
where Name like '%N';
```

This WHERE statement selects observations that begin with any number of characters and end with an N.

```
where Name like 'T_M%';
```

This WHERE statement selects observations that begin with a T, followed by a single character, followed by an M, followed by any number of characters.

LIKE Operator

How do you search for % and _ in your data?

- Beginning with 9.2, an escape character forces literal search for % and _.

Example:

```
where Name like 'A/_C' escape '/';
```

Poll

Quiz

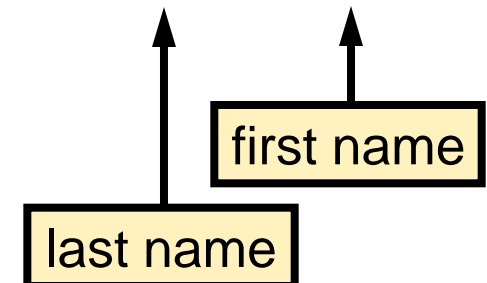


5.05 Quiz

Which WHERE statement will return all the observations that have a first name starting with the letter M for the given values?

- a. `where Name like '__, M_';`
- b. `where Name like '%, M%';`
- c. `where Name like '__, M%';`
- d. `where Name like '%, M_';`

Name
Elvish, Irenie
Ngan, Christina
Hotstone, Kimiko
Daymond, Lucian
Hofmeister, Fong
Denny, Satyakam
Clarkson, Sharryn
Kletschkus, Monica



5.05 Quiz – Correct Answer

Which WHERE statement will return all the observations that have a first name starting with the letter M for the given values?

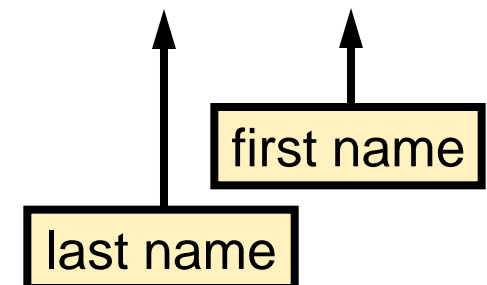
a. `where Name like '__, M_';`

b. `where Name like '%, M%';`

c. `where Name like '__, M%';`

d. `where Name like '%, M_';`

Name
Elvish, Irenie
Ngan, Christina
Hotstone, Kimiko
Daymond, Lucian
Hofmeister, Fong
Denny, Satyakam
Clarkson, Sharryn
Kletschkus, Monica



Business Scenario Part 2

Include only the employees from Australia who have the word `Rep` in their job title.

```
data work.subset1;  
    set orion.sales;  
    where Country='AU' and  
           Job_Title contains 'Rep';  
run;
```

Partial SAS Log

```
NOTE: There were 61 observations read from the data set ORION.SALES.  
      WHERE (Country='AU') and Job_Title contains 'Rep';  
NOTE: The data set WORK.SUBSET1 has 61 observations and 9 variables.
```

Business Scenario Part 2

```
proc print data=work.subset1;  
run;
```

Partial PROC PRINT Output

Obs	Employee_ID	First_ Name	Last_Name	Gender	Salary	Job_Title	Country	Birth_ Date	Hire_ Date
1	120121	Irenie	Elvish	F	26600	Sales Rep. II	AU	-5630	5114
2	120122	Christina	Ngan	F	27475	Sales Rep. II	AU	-1984	6756
3	120123	Kimiko	Hotstone	F	26190	Sales Rep. I	AU	1732	9405
4	120124	Lucian	Daymond	M	26480	Sales Rep. I	AU	-233	6999
5	120125	Fong	Hofmeister	M	32040	Sales Rep. IV	AU	-1852	6999
6	120126	Satyakam	Denny	M	26780	Sales Rep. II	AU	10490	17014
7	120127	Sharryn	Clarkson	F	28100	Sales Rep. II	AU	6943	14184
8	120128	Monica	Kletschkus	F	30890	Sales Rep. IV	AU	9691	17106
9	120129	Alvin	Roebuck	M	30070	Sales Rep. III	AU	1787	9405
10	120130	Kevin	Lyon	M	26955	Sales Rep. I	AU	9114	16922
11	120131	Marinus	Surawski	M	26910	Sales Rep. I	AU	7207	15706
12	120132	Fancine	Kaiser	F	28525	Sales Rep. III	AU	-3923	6848
13	120133	Petrea	Soltau	F	27440	Sales Rep. II	AU	9608	17075
14	120134	Sian	Shannan	M	28015	Sales Rep. II	AU	-3861	5114
15	120135	Alexei	Platts	M	32490	Sales Rep. IV	AU	3313	13788

A large, stylized graphic featuring the words "Question & Answer" in a bold, sans-serif font. The word "Question" is in blue, and the ampersand "&" is in yellow. The word "Answer" is also in blue. The text is centered within a large, horizontal oval shape. The oval has a thick yellow border with a blue outline. The background is white.

Question & Answer

The DROP and KEEP Statements

The *DROP statement* specifies the names of the variables to omit from the output data set(s).

DROP *variable-list*;

The *KEEP statement* specifies the names of the variables to write to the output data set(s).

KEEP *variable-list*;

The *variable-list* specifies the variables to drop or keep, respectively, in the output data set.

The DROP and KEEP Statements

Examples:

```
drop Employee_ID Gender  
Country Birth_Date;
```

```
keep First_Name Last_Name  
Salary Job_Title  
Hire_Date;
```

Business Scenario Part 2

Include only the employee's first name, last name, salary, job title, and hired date in the data set **Work.subset1**.

```
data work.subset1;  
    set orion.sales;  
    where Country='AU' and  
           Job_Title contains 'Rep';  
    keep First_Name Last_Name Salary  
        Job_Title Hire_Date;  
run;
```

Partial SAS Log

```
NOTE: There were 61 observations read from the data set ORION.SALES.  
      WHERE (Country='AU') and Job_Title contains 'Rep';  
NOTE: The data set WORK.SUBSET1 has 61 observations and 5 variables.
```

Business Scenario Part 2

```
proc print data=work.subset1;  
run;
```

Partial PROC PRINT Output

Obs	First_ Name	Last_Name	Salary	Job_Title	Hire_ Date
1	Irenie	Elvish	26600	Sales Rep. II	5114
2	Christina	Ngan	27475	Sales Rep. II	6756
3	Kimiko	Hotstone	26190	Sales Rep. I	9405
4	Lucian	Daymond	26480	Sales Rep. I	6999
5	Fong	Hofmeister	32040	Sales Rep. IV	6999
6	Satyakam	Denny	26780	Sales Rep. II	17014
7	Sharryn	Clarkson	28100	Sales Rep. II	14184
8	Monica	Kletschkus	30890	Sales Rep. IV	17106
9	Alvin	Roebuck	30070	Sales Rep. III	9405
10	Kevin	Lyon	26955	Sales Rep. I	16922
11	Marinus	Surawski	26910	Sales Rep. I	15706
12	Fancine	Kaiser	28525	Sales Rep. III	6848