PSTAT 130

SAS BASE PROGRAMMING

- Lecture 2 -

Dr. Julie Swenson

Objectives

- The PRINT Procedure
  - Syntax
  - Obs. column
  - Variable selection
  - Observation selection/Subset data
  - Column headers
  - Formats
  - Misc. options

The PRINT Procedure

- What is the purpose of the PRINT procedure?

The PRINT Procedure

- Simplest form

```
proc print;
run;
```

- Which data set does this procedure use?

- When do we use this?

The PRINT Procedure

- For a particular SAS data set

```
proc print data=stresstest;
run;
```

- Which data set does this procedure use?
- Does this produce any output?
- What if we modify the file name to `data1.stresstest`?

Default Properties

- Displays observation numbers
- Displays all variables contained in the data set
- Displays all observations contained in the data set
- Uses variables as column headers
- Displays variable values in their “native” format

Suppress the Obs. Column

- By default, `PROC PRINT` displays an observation column – i.e. row numbers on the left side of the report
- The `NOOBS` option suppresses this column
- General form of the `NOOBS` option:

```
PROC PRINT DATA=SAS-data-set NOOBS;
RUN;
```

Suppress the Obs. Column

- How do we suppress the observation column in the following report?

```
proc print data=data1.stresstest;
run;
```

Suppress the Obs. Column

- Specify the NOOBS option in the PROC PRINT statement, as seen below:

```
proc print data=data1.stresstest noobs;
run;
```

Variable Selection

- By default, `PROC PRINT` displays all variables in the data set
- The `VAR` statement gives you the power to change that – you can
  - Select the variables to include in the report
  - Define the order of the variables in the report
- General form of the `VAR` statement

```
VAR variable(s);
```

Variable Selection

- Let's take a look at the `stresstest` data set again
  - All Variables: `ID, Name, RestHR, MaxHR, RecHR, TimeMin, TimeSec, Tolerance, Date`
- Suppose we only want to display
  - `Name, MaxHR, RestHR`
- We could modify the code as follows

```
proc print data=data1.stresstest;
   var Name MaxHR RestHR;
run;
```

Variable Selection Example

- Example:
  - For the data set `empdata` (found in the library `data1`), print only the salary and last names of the employees (in that order).

Variable Selection Example

- Example:
  - For the data set `empdata` (found in the library `data1`), print only the salary and last names of the employees (in that order).

```
proc print data=data1.empdata;
  var Salary LastName;
run;
```

Observation Selection

- By default, `PROC PRINT` displays all the observations in the data set

- What if we only want a **subset** of data?
  - Ex: In the `stresstest` data set, suppose we are interested in finding individuals with a max heart rate of 170 or more

Observation Selection

- The `WHERE` statement
  - enables you to select observations that meet a certain condition
  - can be used with most SAS procedures

Observation Selection

- General form

```
WHERE where-expression;
```

  - `where-expression` is a sequence of operands and operators

Operands include

  - variables or constants

Operators include

  - comparison operators
  - logical operators
  - special operators
  - functions

Observation Selection

- Thus we can use the `WHERE` statement to control which observations are processed

- Previous example:
  - In the `stresstest` data set, we were interested in finding individuals with a max heart rate of 170 or more

Observation Selection

- Thus we can use the `WHERE` statement to control which observations are processed

- Previous example:
  - In the `stresstest` data set, we were interested in finding individuals with a max heart rate of 170 or more

```
proc print data=data1.stresstest noobs;
  where MaxHR>=170;
run;
```

Observation Selection Example

- Example:
  - For the data set `empdata` (in the library `data1`), print only employees who are Flight Attendants

Observation Selection Example

- Example:
  - For the data set `empdata` (in the library `data1`), print only employees who are Flight Attendants

```
proc print data=data1.empdata;
  var JobCode EmpID Salary;
  WHERE JobCode = 'FLTAT';
run;
```

Comparison Operators

Mnemonic Equivalent	Symbol	Definition	Example
EQ	=	equal to	where empnum eq 3374;
NE	^= ~= <>	not equal to	where status ne fulltime;
GT	>	greater than	where hiredate gt '01jun1982'd;
LT	<	less than	where empnum < 2000;
GE	>=	greater than or equal to	where empnum >= 3374;
LE	<=	less than or equal to	where empnum <= 3374;
IN		equal to one from a list of values	where state in ('NC','TX');

- Character comparisons are case-sensitive
- The IN operator allows commas or spaces to separate arguments

Observation Selection Examples

- Print only people under 40
  - `WHERE Age __ 40;`
  - `WHERE Age __ 40;`
  - `WHERE Age __ 39;`
  - `WHERE Age __ 39;`

Observation Selection Examples

- Print only adults
  - `WHERE Age __ 18;`
  - `WHERE Age __ 18;`
  - `WHERE Age __ 17;`
  - `WHERE Age __ 17;`

Observation Selection Examples

- Print only women
  - `WHERE sex __ 'F';`
  - `WHERE sex __ 'F';`
  - `WHERE sex __ 'M';`
  - `WHERE sex __ 'M';`

Logical Operators

- AND (&)
  - If both expressions are true, then the compound expression is true
  - Examples:

```
where JobCode='FLTAT' and Salary>50000;

where JobCode='FLTAT' & Salary>50000;
```

Logical Operators

- OR (|)
  - If either expression is true, then the compound expression is true
  - Examples:

```
where JobCode='PILOT' or JobCode='FLTAT';

where JobCode='PILOT' | JobCode='FLTAT';
```

Logical Operators

- NOT (^ or ~)
  - Can be combined with other operators to reverse the logic of a comparison
  - Examples:

```
where JobCode not in('PILOT','FLTAT');

where JobCode ^ in('PILOT','FLTAT');
```

Special Operators

- Additional special operators supported by the `WHERE` statement are
  - `BETWEEN - AND`
  - `CONTAINS (?)`
  - `LIKE`
  - `SOUNDS LIKE`
  - `IS MISSING` (or `IS NULL`)

Special Operators

- BETWEEN – AND
  - selects observations in which the value of the variable falls within a range of values, inclusively.

```
where Salary between 50000 and 70000;
```

⇔

```
where 50000 <= Salary <= 70000;
```

- CONTAINS (?)
  - selects observations that include the specified substring

```
where LastName ? 'LAM';
```

(LAMBERT, BELLAMY, and ELAM are selected)

  - Contrast **In** and **?**

Special Operators

- `LIKE` selects observations by comparing character values to specified patterns
- `LIKE` uses two "wildcard" symbols
  - a percent sign (`%`) replaces any number of characters
  - an underscore (`_`) replaces one character
- Example

```
where Code like 'E_U%';
```

  - Selects observations where the value of Code begins with an E, followed by a single character, followed by a U, followed by any number of characters

Special Operators

- The `SOUNDS LIKE (=*)` operator selects observations that contain spelling variations of the word or words specified.

```
where Name=*'SMITH';
```

  - Selects names like SYMTHE or SMITT

- `IS NULL` or `IS MISSING` selects observations in which the value of the variable is missing

```
where Flight is missing;
where Flight is null;
```

- You can use the `NOT` logical operator to select non-missing values.

Column Headers

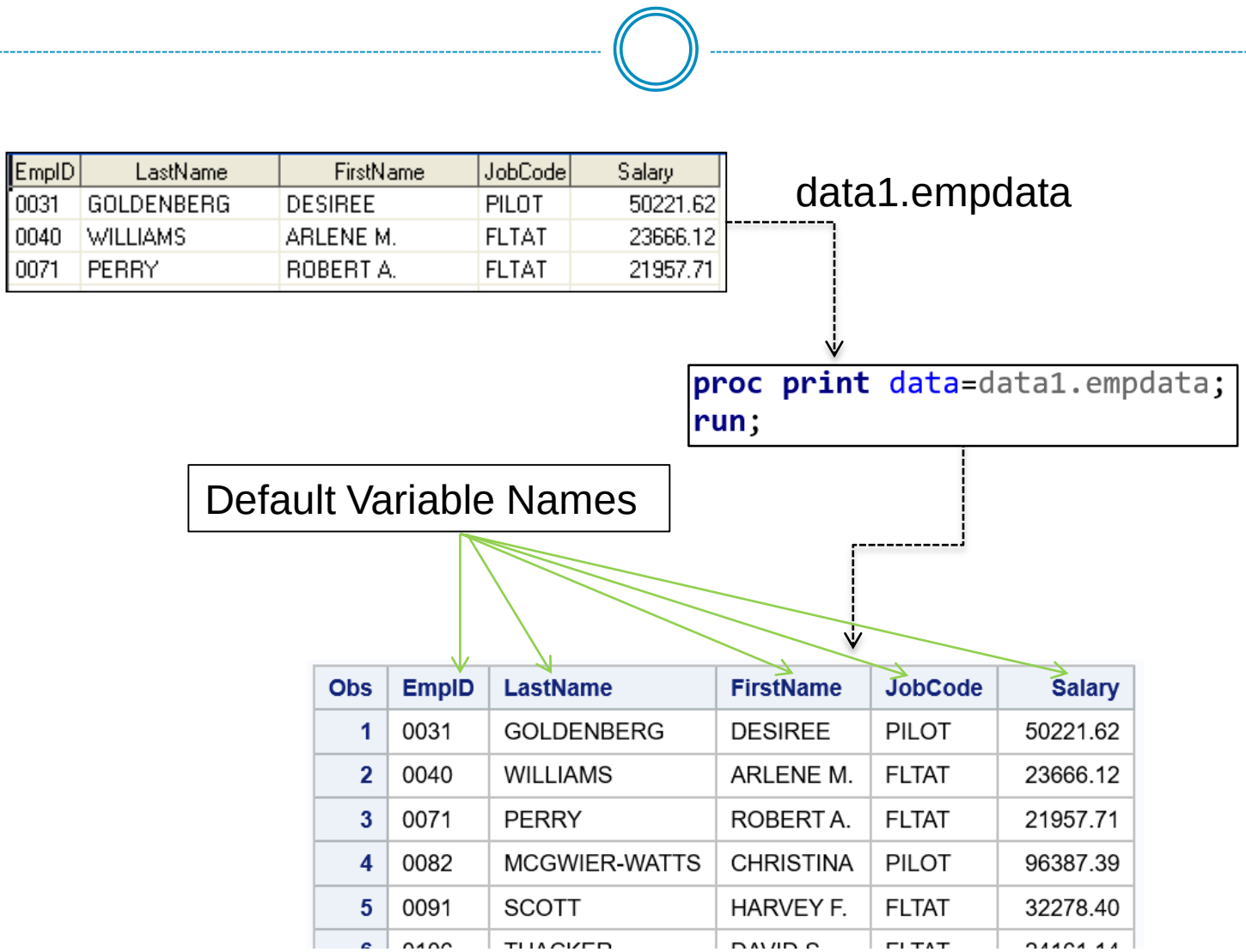
- By default, `PROC PRINT` uses variable names as column headers
- Assigning labels to variables replaces these variables as column headers
- General form of the `LABEL` statement

```
LABEL   variable1='label1'
        variable2='label2';
```

- Example

```
LABEL  Units='Number of Units'
       DOB='Date of Birth';
```

Assign Column Labels



Assign Column Labels

EmpID	LastName	FirstName	JobCode	Salary
0031	GOLDENBERG	DESIREE	PILOT	50221.62
0040	WILLIAMS	ARLENE M.	FLTAT	23666.12
0071	PERRY	ROBERT A.	FLTAT	21957.71

data1.empdata

```
proc print data=data1.empdata label;
        label LastName='Last Name'
              FirstName='First Name'
              Salary='Annual Salary';
run;
```

Obs	EmpID	Last Name	First Name	JobCode	Annual Salary
1	0031	GOLDENBERG	DESIREE	PILOT	50221.62
2	0040	WILLIAMS	ARLENE M.	FLTAT	23666.12
3	0071	PERRY	ROBERT A.	FLTAT	21957.71
4	0082	MCGWIER-WATTS	CHRISTINA	PILOT	96387.39
5	0091	SCOTT	HARVEY F.	FLTAT	32278.40
6	0106	THACKER	DAVID S.	FLTAT	24161.14

Assign Column Labels

EmpID	LastName	FirstName	JobCode	Salary
0031	GOLDENBERG	DESIREE	PILOT	50221.62
0040	WILLIAMS	ARLENE M.	FLTAT	23666.12
0071	PERRY	ROBERT A.	FLTAT	21957.71

data1.empdata

```
proc print data=data1.empdata split=' ';
        label LastName='Last Name'
              FirstName='First Name'
              Salary='Annual Salary';
run;
```

Obs	EmpID	Last Name	First Name	JobCode	Annual Salary
1	0031	GOLDENBERG	DESIREE	PILOT	50221.62
2	0040	WILLIAMS	ARLENE M.	FLTAT	23666.12
3	0071	PERRY	ROBERT A.	FLTAT	21957.71
4	0082	MCGWIER-WATTS	CHRISTINA	PILOT	96387.39
5	0091	SCOTT	HARVEY F.	FLTAT	32278.40
6	0106	THACKER	DAVID S	FLTAT	24161.14

Non-native Formats

- By default, `PROC PRINT` displays variable values in their "native" formats

- What does this mean?
  - In the data set `stresstest`, the variable `date` had the mm/dd/yy format specified in the data set. Hence all the dates were displayed in that "native" format.

- How can we change this?
  - We will find out in the next lecture!

Request Column Totals

- The SUM statement produces column totals for numeric variables
- General form of the SUM statement

```
SUM variable(s);
```

- Example

```
proc print data=data1.empdata noobs;
    var JobCode EmpID Salary;
    sum Salary;
run;
```

(Note: the SUM statement also produces subtotals if you print the data in groups.)

Display Number of Observations

- The N option in PROC print allows you to
  - Print the number of observations in the data set
  - Specify explanatory text to print with the number

- General form of the N option

```
PROC PRINT … N<='explanatory text'>;
```

- Example

```
proc print data=data1.empdata N;
run;
```

Class Exercise 1

- Create a SAS data set called Stocks that contains the variables: Ticker, Open, Close, Volume (in millions)
- Use the following data:

  BAC 29.95 29.58 68.19
  FB 174.25 176.07 28.34
  GOOG 1028.10 1024.38 1.35
  HABT 10.20 10.25 0.64
  TSLA 298.57 301.15 8.82
  TWTR 30.00 30.55 22.48

- Output the data set – suppress the observation numbers

Class Exercise 2

- Create a report using the **`credit`** data set in the data1 folder.
  - Select the following variables
    - Name Transaction
  - Change the column headers as follows
    - `Name = 'Client Name'`
    - `Transaction = 'Transaction Amount'`
  - Suppress the observation numbers
- Aside: can we display the total transaction amount?