

Summary

Main Points

Understanding SAS Libraries

- A SAS library is a collection of one or more SAS files that are recognized by SAS and that are referenced and stored as a unit. You reference a SAS library by a logical name called a libref.
- At the beginning of each SAS session, SAS automatically creates at least two libraries that you can access: **Work**, which is the temporary library, and **Sasuser**, which is a permanent library.
- All SAS data sets have a two-level name that consists of the libref and the data set name. When a data set is stored in the **Work** library, you do not need to include the libref when you reference it.

Defining SAS Libraries

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

- You use a LIBNAME statement to define a SAS library and assign a libref to it. The LIBNAME statement is a global statement.
- In an interactive SAS session, a libref that you assign with a LIBNAME statement remains assigned until you cancel or change the libref or until you end your SAS session. After you end your SAS session, the contents of a permanent library still exist in their physical location. However, each time you start a new SAS session, you must resubmit a LIBNAME statement to reassign the libref to that permanent library.
- When you assign a libref, you must follow the rules for valid librefs. Also, the LIBNAME statement does not create a new directory. Instead, the LIBNAME statement points to an existing directory.

Viewing the Contents of SAS Libraries

```
PROC CONTENTS DATA=libref._ALL_ NODS;  
RUN;
```

- You can use a PROC CONTENTS step to view the contents of a SAS library. When you specify the keyword `_ALL_` in the PROC CONTENTS statement, the step displays a list of all the SAS files that are in the specified SAS library.

Lesson 4: Working with SAS Libraries and SAS Data Sets

- By default, a PROC CONTENTS report includes the descriptor portion of each data set in the SAS library. You can use the NODS option to suppress the descriptor portions in the report.

Understanding the Structure of SAS Data Sets

- A SAS data set is a table with columns and rows. In SAS, the table is called a data set, a column is called a variable, and a row is called an observation. In each observation, each variable has a specific value.
- The data portion of a SAS data set contains the data values. The descriptor portion of a SAS data set contains information about the attributes of the data set and information about each variable. The information in the descriptor portion is called metadata.
- You can see the data portion of a data set in the VIEWTABLE window. You can view some of the information from descriptor portion in the Properties window for the data set.

Viewing the Descriptor Portion of SAS Data Sets

```
PROC CONTENTS DATA=SAS-data-set;  
RUN;
```

- You can use a PROC CONTENTS step to view the descriptor portion of a data set. When you specify a single data set in the PROC CONTENTS step and do not include the NODS option, the step creates a report of the descriptor portion for the specified data set.
- A variable in a SAS data set has a type attribute that can be either character or numeric.
- Character variables are stored with a length of 1 to 32,767 bytes. All numeric variables have a default length of 8 bytes. Numeric values, no matter how many digits they contain, are stored as floating-point numbers in 8 bytes of storage, unless you specify a different length.
- SAS variables have additional attributes such as format, informat, and label, which you learn about in later lessons.

Viewing the Data Portion of SAS Data Sets

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

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

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

- You can use the PRINT procedure to display the data portion of a SAS data set.
- Missing values are represented by blanks for character variables and by periods for numeric variables.
- By default, the PROC PRINT step includes all variables from the data set in the report. You can control which variables SAS includes in the report by using a VAR statement in the PROC PRINT step. The VAR statement also controls the order in which SAS displays the variables in the report.
- By default, SAS includes the observation numbers in the report. You can suppress the **Obs** column by using the NOOBS option in the PROC PRINT statement.

Sorting Observations in SAS Data Sets

```
PROC SORT DATA=input-SAS-data-set  
    <OUT=output-SAS-data-set>;  
    BY <DESCENDING> BY-variable(s);  
RUN;
```

- You can use the SORT procedure to sort the observations in a data set.
- By default, the SORT procedure replaces the original data set with the sorted data set. You can use the OUT= option to create a new output data set instead of replacing the input data set.
- PROC SORT does not generate printed output.
- By default, the SORT procedure sorts observations in ascending order of the values for the variable or variables listed in the BY statement. You can use the DESCENDING option to sort on the variable or variables in descending order instead.
- SAS treats missing values as the smallest possible values.

Lesson 4: Working with SAS Libraries and SAS Data Sets

Defining SAS Libraries for Relational Databases

```
LIBNAME libref engine-name <SAS/ACCESS-options>;
```

```
LIBNAME libref CLEAR;
```

- You can use the SAS/ACCESS LIBNAME statement to define a library so that your SAS programs can access tables that are stored in another vendor's relational database.
- SAS/ACCESS uses SAS/ACCESS engines to read from or write to files. An engine is named for a specific database management file, such as Oracle or DB2. Your SAS installation must include the appropriate SAS/ACCESS interfaces for the types of files that you want to access.
- After a database is associated with a libref, you can use a SAS two-level name to specify any table in the database and then work with that table as you would work with a SAS data set.
- You use the CLEAR option in the LIBNAME statement to disassociate a libref that was previously assigned. Disassociating a libref disconnects the database engine from the database and closes any resources that are associated with that libref's connection.

Sample Code

Windows: Replace *my-file-path* with the location where you stored the practice files.

UNIX and z/OS: Specify the fully qualified path in your operating environment.

Assigning a Libref and Viewing the Contents of the Library

```
libname orion 'my-file-path';

proc contents data=orion._all_nods;
run;
```

Creating Work.NewSalesemps

```
data work.newsalesemps;
    length First_Name $ 12 Last_Name $ 18 Job_Title $ 25;
    infile 'my-file-path\newemployees.csv' dlm=',';
    input First_Name $ Last_Name $ Job_Title $ Salary;
run;
```

Lesson 4: Working with SAS Libraries and SAS Data Sets

Displaying the Descriptor Portion of a Data Set

```
proc contents data=work.newsalesemps;  
run;
```

Displaying the Data Portion of a Data Set

```
proc print data=work.newsalesemps noobs;  
  var Last_Name First_Name Salary;  
run;
```

Sorting a Data Set by Two Variables and Displaying the Output Data Set

```
proc sort data=work.newsalesemps  
  out=work.newsalesemps3;  
  by Job_Title descending Salary;  
run;  
  
proc print data=work.newsalesemps3;  
run;
```