Chapter 3.2:

Level 1

4. Accessing a Permanent Data Set

a. Use an interactive facility to explore the orion library and answer the following questions:

How many observations are in the orion.country data set? ___7____

How many variables are in the orion.country data set? ___6___

What is the name of the last country in the data set? ________South Africa________

b. Submit a PROC CONTENTS step to generate a list of all members in the orion library.

```
proc contents data=orion._all_ nods;
run;
```

What is the name of the last member listed? US_SUPPLIERS

Level 2

5. Viewing General Data Set Properties

a. Examine the general data set properties of orion.staff.

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

b. What sort information is stored for this data set? The General Information section indicates that the data set is sorted. The Variable section indicates that it is sorted by Employee_ID using the ANSI character set, and has been validated.

Challenge

6. SAS Autoexec File

Use the Help facility or product documentation to investigate the SAS autoexec file and answer the following questions.

What is the name of the file? autoexec.sas

What is its purpose? It contains SAS statements that are executed immediately after SAS initializes. These SAS statements can be used to invoke SAS programs automatically, set up certain variables for use during your SAS session, or set system options.

How is it created? With any text editor, but the recommended method is to use a SAS text editor (such as the Enhanced Editor window) and save it using the Save As dialog box.

How could this be useful in a SAS session? It can be used to set the path macro variable and to automatically submit a LIBNAME statement.

Chapter 4.1, 4.2, 4.3:

CH 4.1

Level 1

1. Displaying orion.order_fact with the PRINT Procedure

```
proc print data=orion.order_fact noobs;
  where Total_Retail_Price>500;
  id Customer_ID;
  var Order_ID Order_Type Quantity Total_Retail_Price;
  sum Total_Retail_Price;
run;
```

a. Retrieve the starter program p104e01. Run the program and view the output. Observe that there are 617 observations. Observations might be displayed over two lines, depending on output settings.

b. Add a SUM statement to display the sum of Total_Retail_Price. The last several lines of the report are shown below.

c. Add a WHERE statement to select only the observations with Total_Retail_Price more than 500. Submit the program. Verify that 35 observations were displayed.

What do you notice about the Obs column? The numbers are not sequential. The original observation numbers are displayed.

Did the sum of Total_Retail_Price change to reflect only the subset? Yes

d. Add an option to suppress the Obs column. Verify that there are 35 observations in the results.

How can you verify the number of observations in the results? Check the log.

e. Add an ID statement to use Customer_ID as the identifying variable. Submit the program. The results contain 35 observations.

How did the output change? Customer_ID is the leftmost column and is displayed on each line for an observation.

f. Add a VAR statement to display Customer_ID, Order_ID, Order_Type, Quantity, and Total_Retail_Price.

What do you notice about Customer_ID? There are two Customer_ID columns. The first column is the ID field, and a second one is included because Customer_ID is listed in the VAR statement.

g. Modify the VAR statement to address the issue with Customer_ID.

Level 2

2. Displaying orion.customer_dim with the PRINT Procedure a.

a. Write a PRINT step to display orion.customer_dim.

b. Modify the program to display a subset of orion.customer_dim by selecting only the observations for customers between the ages of 30 and 40. Also, suppress the Obs column. The resulting report should contain 17 observations.

c. Add a statement to use Customer_ID instead of Obs as the identifying column. Submit the program and verify the results.

d. Add a statement to limit the variables to those shown in the report below.

```
proc print data=orion.customer_dim noobs;
   where Customer_Age between 30 and 40;
   id Customer_ID;
   var Customer_Name Customer_Age Customer_Type;
run;
```

CH 4.2
Level 1

5. Sorting orion.employee_payroll and Displaying the New Data Set

a. Open p104e05. Add a PROC SORT step before the PROC PRINT step to sort orion.employee_payroll by Salary, placing the sorted observations into a temporary data set named sort_salary.

b. Modify the PROC PRINT step to display the new data set. Verify that your output matches the report below.

```
proc sort data=orion.employee_payroll out=work.sort_salary;
   by Salary;
run;

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

6. Sorting orion.employee_payroll and Displaying Grouped Observations

g. Open p104e06. Add a PROC SORT step before the PROC PRINT step to sort orion.employee_payroll by Employee_Gender, and within gender by Salary in descending order. Place the sorted observations into a temporary data set named sort_salary2.

h. Modify the PROC PRINT step to display the new data set with the observations grouped by Employee_Gender.

```
proc sort data=orion.employee_payroll out=work.sort_salary2;
   by Employee_Gender descending Salary;
run;

proc print data=work.sort_salary2;
   by Employee_Gender;
run;
```

Level 2

7. Sorting orion.employee_payroll and Displaying a Subset of the New Data Set

a. Sort orion.employee_payroll by Employee_Gender, and by descending Salary within gender.
Place the sorted observations into a temporary data set named sort_sal.

b. Print a subset of the sort_sal data set. Select only the observations for active employees (those without a value for Employee_Term_Date) who earn more than $65,000. Group the report by Employee_Gender, and include a total and subtotals for Salary. Suppress the Obs column. Display only Employee_ID, Salary, and Marital_Status. The results contain 18 observations.

```
proc sort data=orion.employee_payroll out=work.sort_sal;
  by Employee_Gender descending Salary;
run;

proc print data=work.sort_sal noobs;
  by Employee_Gender;
  sum Salary;
  where Employee_Term_Date is missing and Salary>65000;
  var Employee_ID Salary Marital_Status;
run;
```

CH 4.3
Level 1

9. Displaying Titles and Footnotes in a Detail Report

a. Open and submit p104e09 to display all observations for Australian Sales Rep IVs.

b. Add a VAR statement to display only the variables shown in the report below.

c. Add TITLE and FOOTNOTE statements to include the titles and footnotes shown in the report below.

d. Submit the program and verify the output. The results contain five observations as shown below.

e. Submit a null TITLE and null FOOTNOTE statement to clear all titles and footnotes.

```
title1 'Australian Sales Employees';
title2 'Senior Sales Representatives';
footnote1 'Job_Title: Sales Rep. IV';
proc print data=orion.sales noobs;
   where Country='AU' and Job_Title contains 'Rep. IV';
   var Employee_ID First_Name Last_Name Gender Salary;
run;
title;
footnote;
```

10. Displaying Column Headings in a Detail Report

a. Open and submit p104e10. Modify the program to define and use the following labels:

```
title 'Entry-level Sales Representatives';
footnote 'Job_Title: Sales Rep. I';
proc print data=orion.sales noobs label;
   where Country='US' and Job_Title='Sales Rep. I';
   var Employee_ID First_Name Last_Name Gender Salary;
   label Employee_ID="Employee ID"
         First_Name="First Name"
         Last_Name="Last Name"
         Salary="Annual Salary";
run;

title;
footnote;
```

b. Modify the program to use a blank space as the SPLIT= character to generate two-line column headings. Submit the modified program and verify that two-line column labels are displayed.

```
title 'Entry-level Sales Representatives';
footnote 'Job_Title: Sales Rep. I';
proc print data=orion.sales noobs split=' ';
   where Country='US' and Job_Title='Sales Rep. I';
   var Employee_ID First_Name Last_Name Gender Salary;
   label Employee_ID="Employee ID"
         First_Name="First Name"
         Last_Name="Last Name"
         Salary="Annual Salary";
run;

title;
footnote;
```

Level 2

11. Writing an Enhanced Detail Report

a. Write a program to display a subset of orion.employee_addresses as shown below. The program should sort the observations by State, City, and Employee_Name and then display the sorted observations grouped by State. The resulting report should contain 311 observations.

```
proc sort data=orion.employee_addresses out=work.address;
   where Country='US';
   by State City Employee_Name;
run;

title "US Employees by State";
proc print data=work.address noobs split=' ';
   var Employee_ID Employee_Name City Postal_Code;
   label Employee_ID='Employee ID'
         Employee_Name='Name'
         Postal_Code='Zip Code';
   by State;
run;
```

Chapter 5.1, 5.2:

CH 5.1

Level 1

1. Displaying Formatted Values in a Detail Report

a. Open p105e01 and submit. Review the output.

b. Modify the PROC PRINT step to display only Employee_ID, Salary, Birth_Date, and Employee_Hire_Date.

c. Add a FORMAT statement to display Salary in a dollar format, Birth_Date in 01/31/2012 date style, and Employee_Hire_Date in the 01JAN2012 date style, as shown in the report below.

```
proc print data=orion.employee_payroll;
   var Employee_ID Salary Birth_Date Employee_Hire_Date;
   format Salary dollar11.2 Birth_Date mmddyy10.
          Employee_Hire_Date date9.;
run;
```

Level 2

2. Displaying Formatted Values in a Detail Report

a. Write a PROC PRINT step to display the report below using orion.sales as input. Subset the observations and variables to produce the report shown below. Include titles, labels, and formats. The results contain 13 observations.

```
title1 'US Sales Employees';
title2 'Earning Under $26,000';
proc print data=orion.sales label noobs;
   where Country='US' and Salary<26000;
   var Employee_ID First_Name Last_Name Job_Title Salary Hire_Date;
   label First_Name='First Name'
         Last_Name='Last Name'
         Job_Title='Title'
         Hire_Date='Date Hired';
   format Salary dollar10. Hire_Date monyy7.;
run;
title;
footnote;
```

CH 5.2

Level 1

4. Creating User-Defined Formats

a. Retrieve the starter program p105e04.

b. Create a character format named $GENDER that displays gender codes as follows:

c. Create a numeric format named MNAME that displays month numbers as follows:

d. Add a PROC PRINT step to display the data set, applying these two user-defined formats to the Employee_Gender and BirthMonth variables, respectively.

e. Submit the program to produce the following report. The results contain 113 observations.

```
data Q1Birthdays;
   set orion.employee_payroll;
   BirthMonth=month(Birth_Date);
   if BirthMonth le 3;
run;

proc format;
   value $gender
      'F'='Female'
      'M'='Male';
   value mname
       1='January'
       2='February'
       3='March';
run;

title 'Employees with Birthdays in Q1';
proc print data=Q1Birthdays;
   var Employee_ID Employee_Gender BirthMonth;
   format Employee_Gender $gender.
          BirthMonth mname.;
run;
title;
```

Level 2

5. Defining Ranges in User-Defined Formats

a. Retrieve the starter program p105e05.

b. Create a character format named $GENDER that displays gender codes as follows:

c. Create a numeric format named SALRANGE that displays salary ranges as follows:

d. In the PROC PRINT step, apply these two user-defined formats to the Gender and Salary variables, respectively. Submit the program to produce the following report:

Partial PROC PRINT Output

```
proc format;
   value $gender
        'F'='Female'
        'M'='Male'
      other='Invalid code';
   value salrange  .='Missing salary'
       20000-<100000='Below $100,000'
       100000-500000='$100,000 or more'
              other='Invalid salary';
run;

title1 'Salary and Gender Values';
title2 'for Non-Sales Employees';
proc print data=orion.nonsales;
   var Employee_ID Job_Title Salary Gender;
   format Salary salrange. Gender $gender.;
run; title;
```

Chapter 6.2:

Level 2

5. Subsetting Observations Based on Three Conditions

a. Write a DATA step to create work.delays using orion.orders as input.

b. Create a new variable, Order_Month, and set it to the month of Order_Date. Hint: Use the MONTH function.

c. Use a WHERE statement and a subsetting IF statement to select only the observations that meet all of the following conditions:

- Delivery_Date values that are more than four days beyond Order_Date

- Employee_ID values that are equal to 99999999
- Order_Month values occurring in August

d. The new data set should include only Employee_ID, Customer_ID, Order_Date, Delivery_Date, and Order_Month.

e. Add permanent labels for Order_Date, Delivery_Date, and Order_Month as shown below.

f. Add permanent formats to display Order_Date and Delivery_Date as MM/DD/YYYY.

g. Add a PROC CONTENTS step to verify that the labels and formats were stored permanently.

h. Write a PROC PRINT step to create the report below. Results should contain nine observations.

```
data work.delays;
  set orion.orders;
  where Order_Date+4<Delivery_Date
      and Employee_ID=99999999;
  Order_Month=month(Order_Date);
  if Order_Month=8;
  label Order_Date='Date Ordered';
      Delivery_Date='Date Delivered'
      Order_Month='Month Ordered';
      format Order_Date Delivery_Date mmddyy10.;
  keep Employee_ID Customer_ID Order_Date Delivery_Date
     Order_Month;
run;

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

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

Chapter 9.1:

Level 2
2. Creating New Variables

a. Write a DATA step that reads orion.customer to create work.birthday.

b. In the DATA step, create three new variables: Bday2012, BdayDOW2012, and Age2012.

- Bday2012 is the combination of the month of Birth_Date, the day of Birth_Date, and the
constant of 2012 in the MDY function.
- BdayDOW2012 is the day of the week of Bday2012.
- Age2012 is the age of the customer in 2012. Subtract Birth_Date from Bday2012 and divide the result by 365.25.

c. Include only the following variables in the new data set: Customer_Name, Birth_Date, Bday2012, BdayDOW2012, and Age2012.

d. Format Bday2012 to display in the form 01Jan2012. Age2012 should be formatted to display with no decimal places.

e. Write a PROC PRINT step to create the report below. The results should contain 77 observations.

```
data work.birthday;
  set orion.customer;
  Bday2012=mdy(month(Birth_Date),day(Birth_Date),2012);
  BdayDOW2012=weekday(Bday2012);
  Age2012=(Bday2012-Birth_Date)/365.25;
  keep Customer_Name Birth_Date Bday2012 BdayDOW2012 Age2012;
  format Bday2012 date9. Age2012 3.;
run;

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

Chapter 9.2:

Level 2

6. Creating Multiple Variables in Conditional Processing

a. Write a DATA step that reads orion.customer_dim to create work.season.

b. Create two new variables: Promo and Promo2.
The value of Promo is based on the quarter in which the customer was born.
• If the customer was born in the first quarter, then Promo is equal to Winter.
• If the customer was born in the second quarter, then Promo is equal to Spring.

• If the customer was born in the third quarter, then Promo is equal to Summer. • If the customer was born in the fourth quarter, then Promo is equal to Fall.

The value of Promo2 is based on the customer's age:
• For young adults, whose age is between 18 and 25, set Promo2 equal to YA. • For seniors, aged 65 or older, set Promo2 equal to Senior.
• Promo2 should have a missing value for all other customers.

c. The new data set should include only Customer_FirstName, Customer_LastName, Customer_BirthDate, Customer_Age, Promo, and Promo2.

d. Create the report below. The results should include 77 observations. Partial PROC PRINT Output

```
data work.season;
   set orion.customer_dim;
   length Promo2 $ 6;
   Quarter=qtr(Customer_BirthDate);
   if Quarter=1 then Promo='Winter';
   else if Quarter=2 then Promo='Spring';
   else if Quarter=3 then Promo='Summer';
   else if Quarter=4 then Promo='Fall';
   if Customer_Age>=18 and Customer_Age<=25 then  Promo2='YA';
   else if Customer_Age>=65 then  Promo2='Senior';
   keep Customer_FirstName Customer_LastName Customer_BirthDate
        Customer_Age Promo Promo2;
run;

proc print data=work.season;
   var Customer_FirstName Customer_LastName Customer_BirthDate Promo
       Customer_Age Promo2;
run;
```

7. Creating Variables Unconditionally and Conditionally

a. Write a DATA step that reads orion.orders to create work.ordertype.
b. Create a new variable, DayOfWeek, that is equal to the weekday of Order_Date.

c. Create the new variable Type, which is equal to

Retail Sale if Order_Type is equal to 1

Catalog Sale if Order_Type is equal to 2

Internet Sale if Order_Type is equal to 3.

d.Create the new variable SaleAds, which is equal to

Mail if Order_Type is equal to 2
Email if Order_Type is equal to 3.

e.Do not include Order_Type, Employee_ID, and Customer_ID in the new data set. Create the report below. The results should contain 490 observations.

```
data work.ordertype;
  set orion.orders;
  length Type $ 13 SaleAds $ 5;
  DayOfWeek=weekday(Order_Date);
  if Order_Type=1 then
    Type='Retail Sale';
  else if Order_Type=2 then do;
    Type='Catalog Sale';
   SaleAds='Mail';
  end;
  else if Order_Type=3 then do;
    Type='Internet Sale';
   SaleAds='Email';
end;
  drop Order_Type Employee_ID Customer_ID;
run;
proc print data=work.ordertype;
run;
```

Chapter 10.1:

Level 2

3. Concatenating Data Sets with Variables of Different Lengths and Types

a. Open p110e03. Submit the PROC CONTENTS steps or explore the data sets interactively to complete the table below by filling in attribute information for each variable in each data set.

b. Write a DATA step to concatenate orion.charities and orion.us_suppliers, creating a temporary data set, contacts.

```
proc contents data=orion.charities;
run;
proc contents data=orion.us_suppliers;
run;
proc contents data=orion.consultants;
run;
data work.contacts;
  set orion.charities orion.us_suppliers;
run;
proc contents data=work.contacts;
run;
data work.contacts2;
  set orion.us_suppliers orion.charities;
run;
proc contents data=work.contacts2;
run;
data work.contacts3;
  set  orion.us_suppliers orion.consultants;
run;
```

c. Submit a PROC CONTENTS step to examine work.contacts. From which input data set were the variable attributes assigned? The first data set in the set statement, orion.charities

d. Write a DATA step to concatenate orion.us_suppliers and orion.charities, creating a temporary data set, contacts2. Note that these are the same data sets as the previous program, but they are in reverse order.

e. Submit a PROC CONTENTS step to examine work.contacts2. From which input data set were the variable attributes assigned? The first data set in the set statement, orion.us_suppliers

f. Write a DATA step to concatenate orion.us_suppliers and orion.consultants, creating a temporary data set, contacts3.

Why did the DATA step fail? ContactType has been defined as both character and numeric.

Chapter 10.3:

Level 2

5. Merging a Sorted Data Set and an Unsorted Data Set in a One-to-Many Merge

a. Sort orion.product_list by Product_Level to create a new data set, work.product_list.

b. Merge orion.product_level with the sorted data set. Create a new data set, work.listlevel, which includes only Product_ID, Product_Name, Product_Level, and Product_Level_Name.

c. Create the report below, including only observations with Product Level equal to 3. The results should contain 13 observations.

```
proc sort data=orion.product_list
        out=work.product_list;
   by Product_Level;
run;
data work.listlevel;
   merge orion.product_level work.product_list ;
   by Product_Level;
   keep Product_ID Product_Name Product_Level Product_Level_Name;
run;

proc print data=work.listlevel noobs;
   where Product_Level=3;
run;
```

Chapter 10.4:

Level 2

8. Merging Using the IN= and RENAME= Options

a. Write a PROC SORT step to sort orion.customer by Country to create a new data set, work.customer.

b. Write a DATA step to merge the resulting data set with orion.lookup_country by Country to create a new data set, work.allcustomer.
In the orion.lookup_country data set, rename Start to Country and rename Label to Country_Name.
Include only four variables: Customer_ID, Country, Customer_Name, and Country_Name.

c. Create the report below. The results should contain 308 observations.

d. Modify the DATA step to store only the observations that contain both customer information and country information. A subsetting IF statement that references IN= variables in the MERGE statement must be added.

e. Submit the program to create the report below. The results should contain 77 observations.

```
proc sort data=orion.customer
       out=work.customer;
  by Country;
run;

data work.allcustomer;
  merge work.customer(in=Cust)
      orion.lookup_country(rename=(Start=Country
                              Label=Country_Name) in=Ctry);
  by Country;
  keep Customer_ID Country Customer_Name Country_Name;
  if Cust=1 and Ctry=1;
run;
proc print data=work.allcustomer;
run;
```