Satoshi Ido
34788706
11/01/2023

STAT506

HW5

1. Debugging the PDV

   Modify the code below to do the following:

   - Ensure that the values of Size won't get truncated.
   - Add PUTLOG statements to provide the following information in the log:
     i. Immediately after the SET statement, write "START DATA STEP ITERATION" to the log as a color- coded NOTE.
     ii. Before the Type= assignment statement, write the value of Type to the log.
     iii. After the Type= assignment statement, write the value of Type to the log.
     iv. At the end of the DATA step, write the contents of the entire PDV to the log.

   Show a screenshot of the log showing all iterations of the DATA step.

```
NOTE: START DATA STEP ITERATION
Type=RIVER
Type=River
Region=Midwest Type=River ParkName=Ozark National Scenic Riverways DayVisits=1,241,480 Campers=87152 OtherLodging=0 Acres=80,785
AvgMonthlyVisitors=110,719 size=Medium _ERROR_=0 _N_=1
NOTE: START DATA STEP ITERATION
Type=RIVER
Type=River
Region=Midwest Type=River ParkName=Buffalo National River DayVisits=1,785,359 Campers=85131 OtherLodging=3,150 Acres=94,293
AvgMonthlyVisitors=156,137 size=Large _ERROR_=0 _N_=2
NOTE: START DATA STEP ITERATION
Type=RIVER
Type=River
Region=Northeast Type=River ParkName=New River Gorge National River DayVisits=1,197,931 Campers=5491 OtherLodging=0 Acres=72,186
AvgMonthlyVisitors=100,285 size=Small _ERROR_=0 _N_=3
NOTE: There were 3 observations read from the data set PG2.NP_FINAL.
      WHERE Type='RIVER';
NOTE: The data set WORK.NP_MONUMENTS has 3 observations and 5 variables.
NOTE: DATA statement used (Total process time):
      real time           0.00 seconds
      user cpu time       0.00 seconds
      system cpu time     0.00 seconds
      memory              981.65k
      OS Memory           20136.00k
      Timestamp           11/01/2023 11:44:14 PM
      Step Count                        41  Switch Count  2
      Page Faults                       0
      Page Reclaims                     185
      Page Swaps                        0
      Voluntary Context Switches        17
      Involuntary Context Switches      0
      Block Input Operations            0
      Block Output Operations           272
```

2. Directing Output to Multiple Tables

   Based on the data in pg2.np_yearlytraffic, create three new tables named river, seashore, and other.

   - Use SELECT and WHEN statements (see the SAS Documentation or elsewhere) to direct the output:
     i. If ParkType is “National River”, then the row should be output to the river table
     ii. If ParkType is “National Seashore”, then the row should be output to the seashore table
     iii. If ParkType is anything else, the row should be output to the other table
   - Only read the columns ParkName, ParkType, and Location into the PDV
   - Drop ParkType from the river and seashore tables

   Show your code and a screenshot of the corresponding log notes.

```
data river(drop = ParkType) seashore(drop = ParkType) other;
    set pg2.np_yearlytraffic(keep = ParkName ParkType Location);
    select (ParkType);
        when ("National River") output river;
        when ("National Seashore") output seashore;
        otherwise output other;
    end;
run;
```

```
NOTE: There were 478 observations read from the data set PG2.NP_YEARLYTRAFFIC.
NOTE: The data set WORK.RIVER has 60 observations and 2 variables.
NOTE: The data set WORK.SEASHORE has 47 observations and 2 variables.
NOTE: The data set WORK.OTHER has 371 observations and 3 variables.
NOTE: DATA statement used (Total process time):
      real time           0.00 seconds
      user cpu time       0.00 seconds
      system cpu time     0.00 seconds
      memory              1572.31k
      OS Memory           21168.00k
      Timestamp           10/31/2023 08:50:29 PM
      Step Count                        35  Switch Count  6
      Page Faults                       0
      Page Reclaims                     366
      Page Swaps                        0
      Voluntary Context Switches        39
      Involuntary Context Switches      0
      Block Input Operations            0
      Block Output Operations           792
```

3. Creating Summed Columns

   Open the pg2.np_yearlytraffic table. Notice the Count column records the number of cars that have passed through a particular Location at each Park.

   Write a PROC SORT step to create a new table named traffic_sort based on the data in pg2.np_yearlytraffic:

   - The table should be sorted by ParkType, and then by ParkName within ParkType (both in ascending order).
   - The table should only include rows where ParkType is “National Preserve”, “National River”, or “National Seashore”.

   Write a DATA step to create a new table named total_traffic based on traffic_sort:

   - Group the data by ParkType, and then by ParkName within ParkType (both in ascending order).
   - Create a new column named TypeCount that is the running total of Count within each ParkType. Reset the count for each new value of ParkType.
   - Create a new column named NameCount that is the running total of Count within each ParkName. Reset the count for each new value of ParkName.
   - Format TypeCount and NameCount so values are displayed with commas.
   - Keep only the ParkType, ParkName, TypeCount, and NameCount columns.
   - Write only the last row for each ParkName to the output table.

   Write a PROC PRINT to display total_traffic.

   Show all your code and a screenshot of the PROC PRINT output.

```
proc sort data=pg2.np_yearlytraffic out=traffic_sort;
    by ParkType ParkName;
    where ParkType in ("National Preserve" "National River" "National Seashore");
run;

data total_traffic;
    set traffic_sort;
    by ParkType ParkName;
    if first.ParkType=1 then TypeCount=0;
    TypeCount+Count;
    if first.ParkName=1 then NameCount=0;
    NameCount+Count;
    format TypeCount NameCount comma12.;
    keep ParkType ParkName TypeCount NameCount;
    if last.ParkName=1 then output;
run;

proc print data=total_traffic;
run;
```

Obs	ParkName	ParkType	TypeCount	NameCount
1	Big Cypress NPRES	National Preserve	239,769	239,769
2	Big Thicket NPRES	National Preserve	263,913	24,144
3	Little River Canyon NPRES	National Preserve	473,044	209,131
4	Mojave NPRES	National Preserve	771,629	298,585
5	Tallgrass Prairie NPRES	National Preserve	773,468	1,839
6	Timucuan EHP	National Preserve	1,067,315	293,848
7	Big South Fork NRRA	National River	292,943	292,943
8	Buffalo NR	National River	774,615	481,672
9	New River Gorge NR	National River	1,139,688	365,073
10	Ozark NSR	National River	1,499,496	359,809
11	Assateague Island NS	National Seashore	775,953	775,953
12	Canaveral NS	National Seashore	1,284,284	508,331
13	Cape Cod NS	National Seashore	2,708,633	1,424,349
14	Cape Hatteras NS	National Seashore	4,008,857	1,300,224
15	Cape Lookout NS	National Seashore	4,057,349	48,493
16	Gulf Islands NS	National Seashore	5,447,175	1,389,826
17	Padre Island NS	National Seashore	5,809,095	361,920
18	Point Reyes NS	National Seashore	6,622,359	813,264

4. Using Numeric Functions

   Modify the following code to do the following:

   - Create a new column named MonthlyRainTotalCM which is MonthlyRainTotal multiplied by 2.54 and rounded to the nearest half centimeter.
   - Create a new column named Date which is the date portion of the DateTime column.
   - Create a new column named MonthEnd which is the date corresponding to the last day of that month.
   - Format Date and MonthEnd as date values.
   - Keep only the StationName, MonthlyRainTotal, MonthlyRainTotalCM, Date, and MonthEnd columns.
   - Only output a row if it is the last row within its Month.
   - Add a PROC PRINT to display rainsummary.

   Show all your code and a screenshot of the PROC PRINT output.

```
data rainsummary;
    set pg2.np_hourlyrain;
    by Month;
    if first.Month=1 then MonthlyRainTotal=0;
    MonthlyRainTotal+Rain;
    MonthlyRainTotalCM = round(MonthlyRainTotal * 2.54, 0.5);
    Date = datepart(DateTime);
    MonthEnd = intnx("month", Date, 0, "E");
    format Date MonthEnd date.;
    keep StationName MonthlyRainTotal MonthlyRainTotalCM Date MonthEnd;
    if last.Month=1 then output;
run;

proc print data=rainsummary;
run;
```

Obs	StationName	MonthlyRainTotal	MonthlyRainTotalCM	Date	MonthEnd
1	PANTHER JUNCTION TX	0.3	1.0	24JAN17	31JAN17
2	PANTHER JUNCTION TX	0.0	0.0	01FEB17	28FEB17
3	PANTHER JUNCTION TX	0.0	0.0	01MAR17	31MAR17
4	PANTHER JUNCTION TX	0.0	0.0	16APR17	30APR17
5	PANTHER JUNCTION TX	2.0	5.0	27MAY17	31MAY17
6	PANTHER JUNCTION TX	1.3	3.5	30JUN17	30JUN17
7	PANTHER JUNCTION TX	4.8	12.0	30JUL17	31JUL17
8	PANTHER JUNCTION TX	2.4	6.0	29AUG17	31AUG17
9	PANTHER JUNCTION TX	3.4	8.5	25SEP17	30SEP17
10	PANTHER JUNCTION TX	0.3	1.0	22OCT17	31OCT17
11	PANTHER JUNCTION TX	0.1	0.5	12NOV17	30NOV17
12	PANTHER JUNCTION TX	0.1	0.5	31DEC17	31DEC17

5. Using Character Functions

   Write a DATA step to create a new table named clean_traffic based on pg2.np_monthlytraffic:

   - Add a LENGTH statement to create a new five-character-long column named Type. Add an assignment
   - statement that uses the SCAN function to extract the last word from the ParkName column and assign the resulting value to Type.
   - Use the SUBSTR function to create a new column named Park that reads each ParkName value and
   - excludes the Type code at the end of the string. Note: Use the FIND function to identify the position number of the Type string (with any trailing blanks removed

from Type). That value can be used in calculating the third argument of the SUBSTR function to specify how many characters to read.

- Add an assignment statement to use the UPCASE and COMPRESS functions to change the case of Region and remove any blanks.
- Add an assignment statement to change the case of Location to proper case. Use the COMPBL function to remove any extra blanks between words.
- Use the TRANWRD function to create a new column named Gate that reads Location (after performing the previous step) and converts the string “Traffic Count At” to a blank.
- Create a new column named GateCode that concatenates ParkCode and Gate together with an underscore (_) between the strings.

Then write a PROC PRINT step to display the first 10 rows of clean_traffic where Month is equal to 5. Only display the columns Type, Park, Region, Location, GateCode, and Count, in that order. Include a descriptive title.

Show all your code and a screenshot of the PROC PRINT output.

```
data clean_traffic;
    set pg2.np_monthlytraffic;
    length Type $ 5;
    Type = scan(ParkName, -1);
    Park = substr(ParkName, 1, find(ParkName, Type, "t")-2);
    Region = compress(upcase(Region), " ");
    Location = compbl(propcase(Location));
    Gate = tranwrd(Location, "Traffic Count At", " ");
    GateCode = catx("_", ParkCode, Gate);
run;

title "Traffic summary";
proc print data=clean_traffic (obs=10);
    var Type Park Region Location GateCode Count;
    where Month=5;
run;
```

Traffic summary

Obs	Type	Park	Region	Location	GateCode	Count
5	NP	Acadia	NORTHEAST	Traffic Count At Sand Beach	ACAD_Sand Beach	25,473
17	NP	Acadia	NORTHEAST	Traffic Count At Schoodic	ACAD_Schoodic	7,889
29	NP	Arches	INTERMOUNTAIN	Total Vehicles Entering Park	ARCH_Total Vehicles Entering Park	67,916
41	NS	Assateague Island	NORTHEAST	Traffic Count At Bayberry Drive	ASIS_Bayberry Drive	29,671
53	NS	Assateague Island	NORTHEAST	Traffic Count At Fws Entrance	ASIS_Fws Entrance	34,432
65	NP	Badlands	MIDWEST	Total Traffic Count At Interior Entrance (2602)	BADL_Total Interior Entrance (2602)	11,169
77	NP	Badlands	MIDWEST	Total Traffic Count At Northeast Entrance (2601)	BADL_Total Northeast Entrance (2601)	15,988
89	NP	Badlands	MIDWEST	Total Traffic Count At Pinnacles Entrance (2603)	BADL_Total Pinnacles Entrance (2603)	12,511
101	NM	Bandelier	INTERMOUNTAIN	Traffic Count At Entrance	BAND_Entrance	0
113	NP	Big Bend	INTERMOUNTAIN	Traffic Count At Route 11-Pers.Gap	BIBE_Route 11-Pers.Gap	5,135

6. Using Functions to Optimize Column Length
    - Run the program below. Notice that the Column1 column contains raw data with values separated by various symbols. The SCAN function is being used to extract the ParkCode and ParkName values. Also examine the PROC CONTENTS report. Notice that ParkCode and ParkType have a length of 200, which is the same as Column1. Note: When the SCAN function creates a new column, the new column will have the same length as the column listed as the first argument.
    - The ParkCode column should include only the first four characters in the string. Add a LENGTH statement in the proper place in the DATA step to define the length of ParkCode as 4.
    - The length for the ParkName column could be optimized by determining the longest string and setting an appropriate length. Modify the DATA step to create a new column named NameLength that uses the LENGTH function to return the position of the last non-blank character for each value of ParkName.
    - Use a RETAIN statement to create a new numeric column named MaxLength that has an initial value of 0. Use an assignment statement and the MAX function to set the value of MaxLength to either the current value of NameLength or MaxLength, whichever is larger.
    - Use the END= option in the SET statement (see the SAS Documentation, especially Example 11) to create a temporary variable in the PDV named LastRow. LastRow will be 0 for all rows until the last row of the table, when it will be 1. Add an IF-THEN statement to write the value of MaxLength to the log if the value of LastRow is 1.

Show your final DATA step code and include a screenshot of the log that shows the resulting value of MaxLength.

Note: Although it would be the next logical step, you don't actually need to end up changing the length of ParkName to MaxLength. I know you know how to do that.

```
data parklookup;
    set pg2.np_unstructured_codes end=LastRow;
    length ParkCode $ 4;
    ParkCode=scan(Column1, 2, '{}:,"()-');
    ParkName=scan(Column1, 4, '{}:,"()');
    NameLength = length(ParkName);
    retain MaxLength 0;
    MaxLength = max(NameLength, MaxLength);
    if LastRow=1 then putlog MaxLength=;
run;
```

```
MaxLength=83
NOTE: There were 755 observations read from the data set PG2.NP_UNSTRUCTURED_CODES.
NOTE: The data set WORK.PARKLOOKUP has 755 observations and 5 variables.
NOTE: DATA statement used (Total process time):
      real time           0.00 seconds
      user cpu time       0.00 seconds
      system cpu time     0.00 seconds
      memory              1380.59k
      OS Memory           24744.00k
      Timestamp           11/01/2023 02:00:59 AM
      Step Count                        222  Switch Count  2
      Page Faults                       0
      Page Reclaims                     97
      Page Swaps                        0
      Voluntary Context Switches        16
      Involuntary Context Switches      0
      Block Input Operations            0
      Block Output Operations           776
```