SAS Advanced Programming

SAP3 SAS Advanced Programming Techniques

SAP301 DATA Step Review

SAP302 Using a variety of Advanced Functions and Performing Pattern Matching with Perl Regular Expressions

Libname.sas

```
%let path=~/EPG3M6;
%let pathout=&path/output;
libname pg3 "&path/data" filelockwait=20;

* FILELOCKWAIT=20 specifies SAS will wait up to 20 seconds
  for a locked file to become available. Use this option
  to avoid a lock error when using the FCMP procedure. */

*Scenario 1;
data work.PctGrowth18Yrs;
   length Continent $ 13 Country $ 18;
   set pg3.population_top25countries;
   Country=scan(CountryCodeName,2,'-');
   PctGrowth18Yrs=(Pop2017-Pop2000)/Pop2000*100;
   drop CountryCodeName;
   format Pop2000 Pop2017 comma16. PctGrowth18Yrs 5.1;
run;

*Scenario 2;
proc sort data=pg3.population_top25countries
        out=work.continent_sorted;
```

```
   by Continent descending Pop2017;
run;

data work.PctGrowth18Yrs_Cont;
   set work.continent_sorted;
   by Continent;
   if first.Continent=1 then do;
     Count=0; Pop2000Total=0; Pop2017Total=0;
   end;
   Count+1;
   Pop2000Total+Pop2000;
   Pop2017Total+Pop2017;
   if last.Continent=1 then do;
     PctGrowth18Yrs_Cont=
       (Pop2017Total-Pop2000Total)/Pop2000Total*100;
     output;
   end;
   format Pop2000Total Pop2017Total comma16.
        PctGrowth18Yrs_Cont 5.1;
   keep Continent Count Pop2000Total
      Pop2017Total PctGrowth18Yrs_Cont;
run;
```

Table: WORK.PCTGROWTH18YRS | View: Column names | Filter: (none)

Columns

- Select all
- Continent
- Country
- Pop2000
- Pop2017
- PctGrowthAvgYr
- PctGrowth18Yrs

Total rows: 25 Total columns: 6 Rows

	Continent	Country	Pop2000	Pop2017	PctGrowthAvgYr	PctGrowth18Yrs
1	Asia	China	1,262,645,000	1,386,395,000	0.5	9.8
2	Asia	India	1,053,050,912	1,339,180,127	1.4	27.2
3	North America	United States	282,162,411	325,719,178	0.8	15.4
4	Asia	Indonesia	211,540,429	263,991,379	1.3	24.8
5	South America	Brazil	175,287,587	209,288,278	1	19.4
6	Asia	Pakistan	138,523,285	197,015,955	2.1	42.2
7	Africa	Nigeria	122,352,009	190,886,311	2.6	56.0
8	Asia	Bangladesh	131,581,243	164,669,751	1.3	25.1
9	Europe	Russian Federation	146,596,557	144,495,044	-0.1	-1.4
10	North America	Mexico	101,719,673	129,163,276	1.4	27.0
11	Asia	Japan	126,843,000	126,785,797	0	-0.0

Table: WORK.CONTINENT_SORTED | View: Column names | Filter: (none)

Columns

- Select all
- Continent
- CountryCodeName
- Pop2000
- Pop2017
- PctGrowthAvgYr

Total rows: 25 Total columns: 5

	Continent	CountryCodeName	Pop2000	Pop2017	PctGrowthAvgYr
1	Africa	NGA-Nigeria	122352009	190886311	2.6
2	Africa	ETH-Ethiopia	66537331	104957438	2.7
3	Africa	EGY-Egypt, Arab Rep.	69905988	97553151	2
4	Africa	COD-Congo, Dem. Rep.	47076387	81339988	3.2
5	Africa	TZA-Tanzania	34178042	57310019	3
6	Africa	ZAF-South Africa	45728315	56717156	1.3
7	Asia	CHN-China	1262645000	1386395000	0.5
8	Asia	IND-India	1053050912	1339180127	1.4
9	Asia	IDN-Indonesia	211540429	263991379	1.3
10	Asia	PAK-Pakistan	138523285	197015955	2.1
11	Asia	BGD-Bangladesh	131581243	164669751	1.3
12	Asia	JPN-Japan	126843000	126785797	0

Table: WORK.PCTGROWTH18YRS_CONT | View: Column names | Filter: (none)

Columns

- Select all
- Continent
- Count
- Pop2000Total
- Pop2017Total
- PctGrowth18Yrs_Cont

Total rows: 5 Total columns: 5 Rows 1-5

	Continent	Count	Pop2000Total	Pop2017Total	PctGrowth18Yrs_Cont
1	Africa	6	385,778,072	588,764,063	52.6
2	Asia	11	3,274,790,996	3,909,442,220	19.4
3	Europe	5	405,555,187	420,882,381	3.8
4	North America	2	383,882,084	454,882,454	18.5
5	South America	1	175,287,587	209,288,278	19.4

**********;

* Activity 1.03 *;

* 1) View the DATA step syntax. Run the DATA step. *;

* View the output table. *;

```
*    - How many rows are in the output table?          *;
*    - What is the value of Year?                      *;
*  2) Uncomment the OUTPUT statement. Run the DATA step.  *;
*    View the output table.                            *;
*    - How many rows are in the output table?          *;
*    - What is the range of values for Year?           *;
*    View the SAS log.                                 *;
*    - How many times did SAS iterate through the DATA   *;
*      step based on the PUTLOG statement?             *;
*  3) Run the PROC SGPLOT step. In what year will the     *;
*    predicted population of India exceed China?       *;
************************************************************;

data work.PredictedPopulation;
   set pg3.population_top25countries;
   putlog 'Top of Step - Iteration #' _N_;
   PopPredicted=Pop2017;
   do Year=2018 to 2022;
     PopPredicted=PopPredicted+(PopPredicted*PctGrowthAvgYr/100);
     *output;
   end;
   keep CountryCodeName Pop2000 Pop2017 PctGrowthAvgYr PopPredicted Year;
   format Pop2000 Pop2017 PopPredicted comma16.;
run;
```

Table: WORK.PREDICTEDPOPULATION | View: Column names | Filter: (none)

Columns

- Select all
- CountryCodeName
- Pop2000
- Pop2017
- PctGrowthAvgYr
- PopPredicted
- Year

Total rows: 25 Total columns: 6 Rows 1-25

	CountryCodeName	Pop2000	Pop2017	PctGrowthAvgYr	PopPredicted	Year
1	CHN-China	1,262,645,000	1,386,395,000	0.5	1,421,403,211	2023
2	IND-India	1,053,050,912	1,339,180,127	1.4	1,435,584,534	2023
3	USA-United States	282,162,411	325,719,178	0.8	338,958,080	2023
4	IDN-Indonesia	211,540,429	263,991,379	1.3	281,602,802	2023
5	BRA-Brazil	175,287,587	209,288,278	1	219,964,084	2023
6	PAK-Pakistan	138,523,285	197,015,955	2.1	218,589,909	2023
7	NGA-Nigeria	122,352,009	190,886,311	2.6	217,025,911	2023
8	BGD-Bangladesh	131,581,243	164,669,751	1.3	175,655,218	2023
9	RUS-Russian Federation	146,596,557	144,495,044	-0.1	143,774,012	2023
10	MEX-Mexico	101,719,673	129,163,276	1.4	138,461,434	2023
11	JPN-Japan	126,843,000	126,785,797	0	126,785,797	2023

```
data work.PredictedPopulation;
   set pg3.population_top25countries;
   putlog 'Top of Step - Iteration #' _N_;
   PopPredicted=Pop2017;
   do Year=2018 to 2022;
     PopPredicted=PopPredicted+(PopPredicted*PctGrowthAvgYr/100);
     output;
   end;
   keep CountryCodeName Pop2000 Pop2017 PctGrowthAvgYr PopPredicted Year;
   format Pop2000 Pop2017 PopPredicted comma16.;
run;
```

Table: WORK.PREDICTEDPOPULATION | View: Column names | Filter: (none)

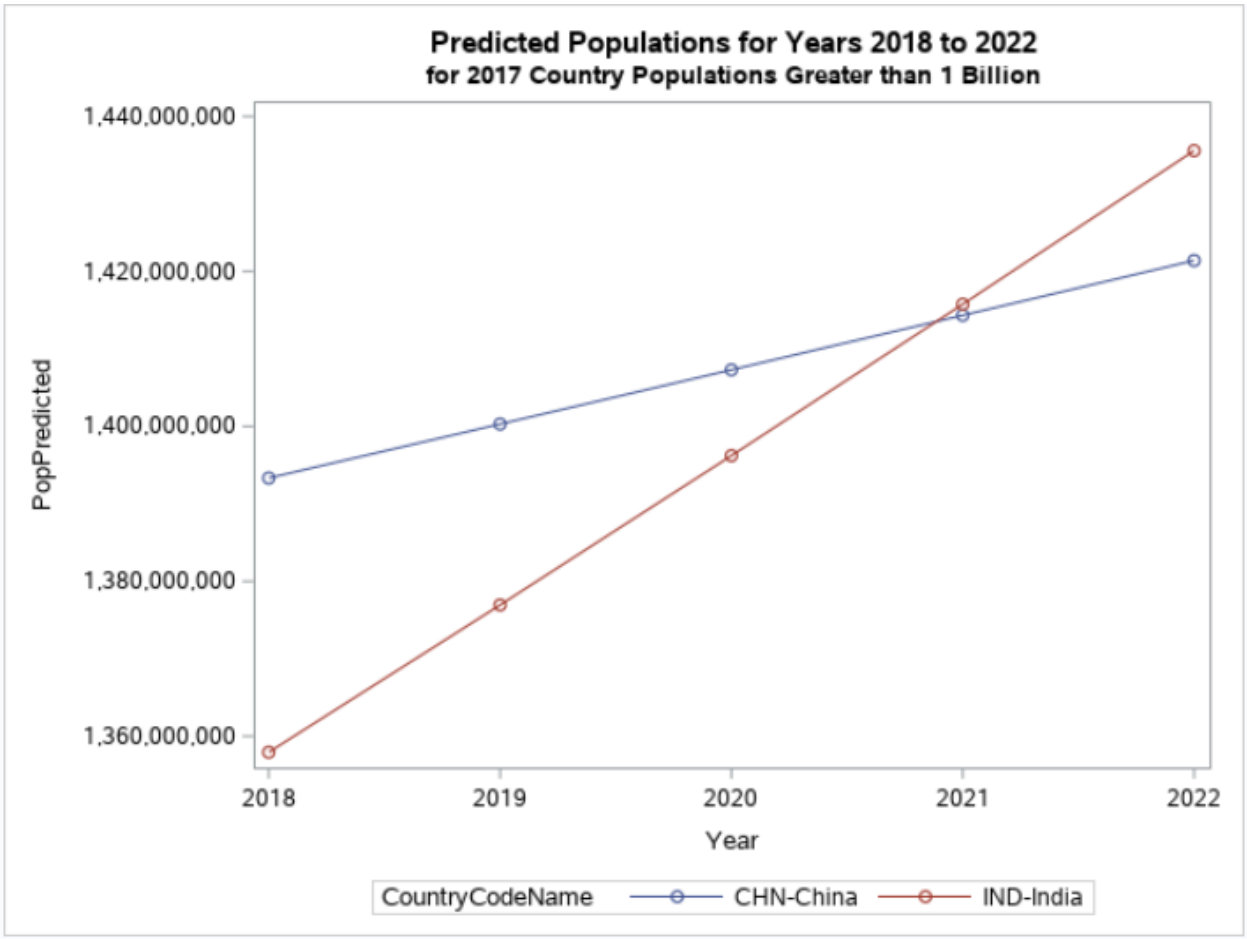
Columns: Select all, CountryCodeName, Pop2000, Pop2017, PctGrowthAvgYr, PopPredicted, Year

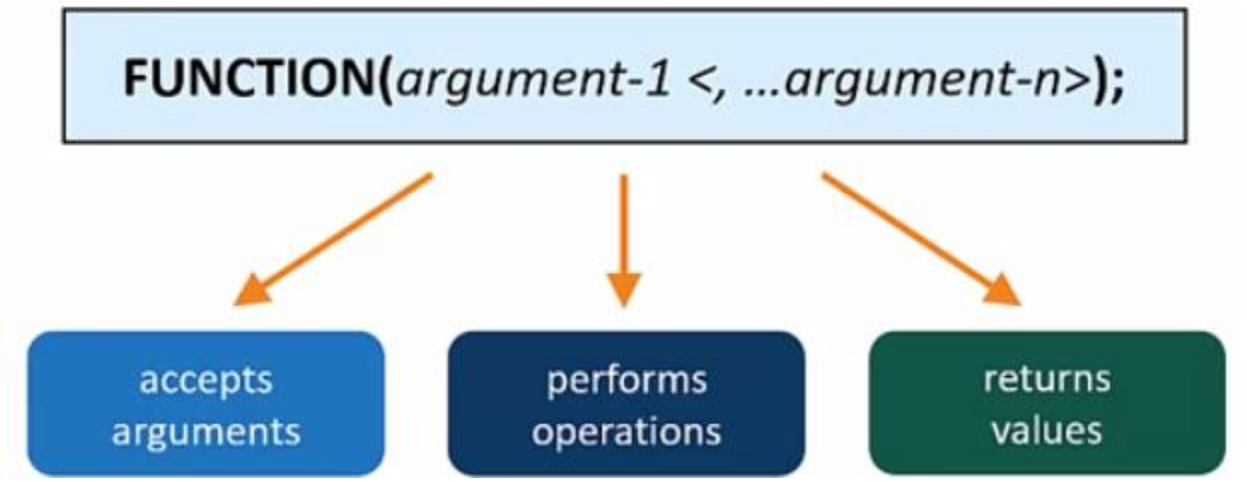
Property | Value

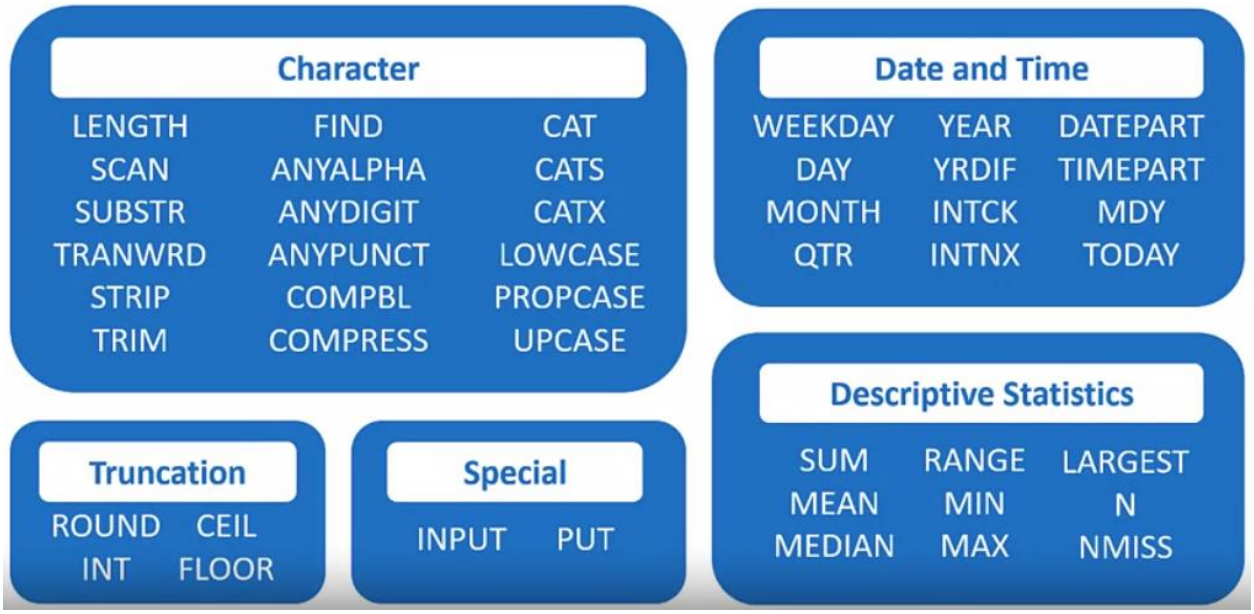
Total rows: 125 Total columns: 6 — Rows 1-100

	CountryCodeName	Pop2000	Pop2017	PctGrowthAvgYr	PopPredicted	Year
1	CHN-China	1,262,645,000	1,386,395,000	0.5	1,393,326,975	2018
2	CHN-China	1,262,645,000	1,386,395,000	0.5	1,400,293,610	2019
3	CHN-China	1,262,645,000	1,386,395,000	0.5	1,407,295,078	2020
4	CHN-China	1,262,645,000	1,386,395,000	0.5	1,414,331,553	2021
5	CHN-China	1,262,645,000	1,386,395,000	0.5	1,421,403,211	2022
6	IND-India	1,053,050,912	1,339,180,127	1.4	1,357,928,649	2018
7	IND-India	1,053,050,912	1,339,180,127	1.4	1,376,939,650	2019
8	IND-India	1,053,050,912	1,339,180,127	1.4	1,396,216,805	2020
9	IND-India	1,053,050,912	1,339,180,127	1.4	1,415,763,840	2021
10	IND-India	1,053,050,912	1,339,180,127	1.4	1,435,584,534	2022
11	USA-United States	282,162,411	325,719,178	0.8	328,324,931	2018
12	USA-United States	282,162,411	325,719,178	0.8	330,951,531	2019
13	USA-United States	282,162,411	325,719,178	0.8	333,599,143	2020
14	USA-United States	282,162,411	325,719,178	0.8	336,267,936	2021
15	USA-United States	282,162,411	325,719,178	0.8	338,958,080	2022

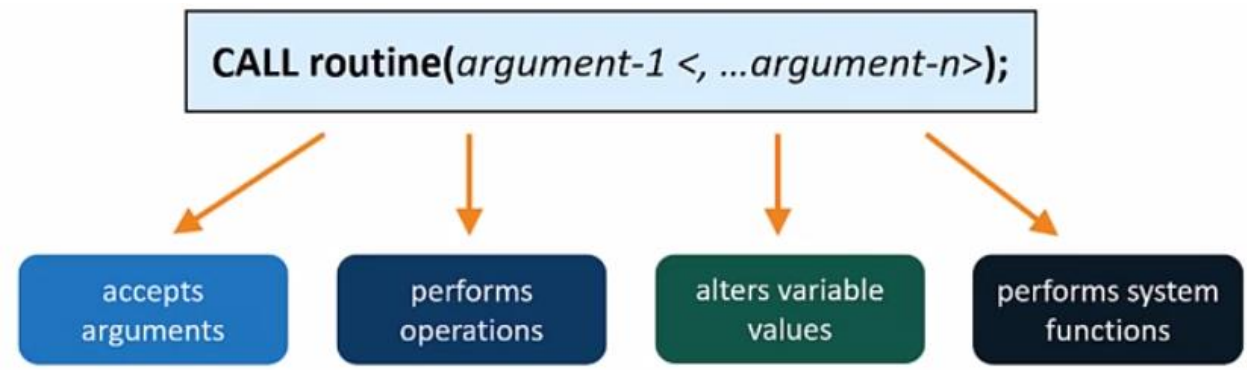
```
title1 'Predicted Populations for Years 2018 to 2022';
title2 'for 2017 Country Populations Greater than 1 Billion';
proc sgplot data=work.PredictedPopulation;
   where Pop2017>1000000000;
   series x=Year y=PopPredicted / group=CountryCodeName markers;
run;
title;
```

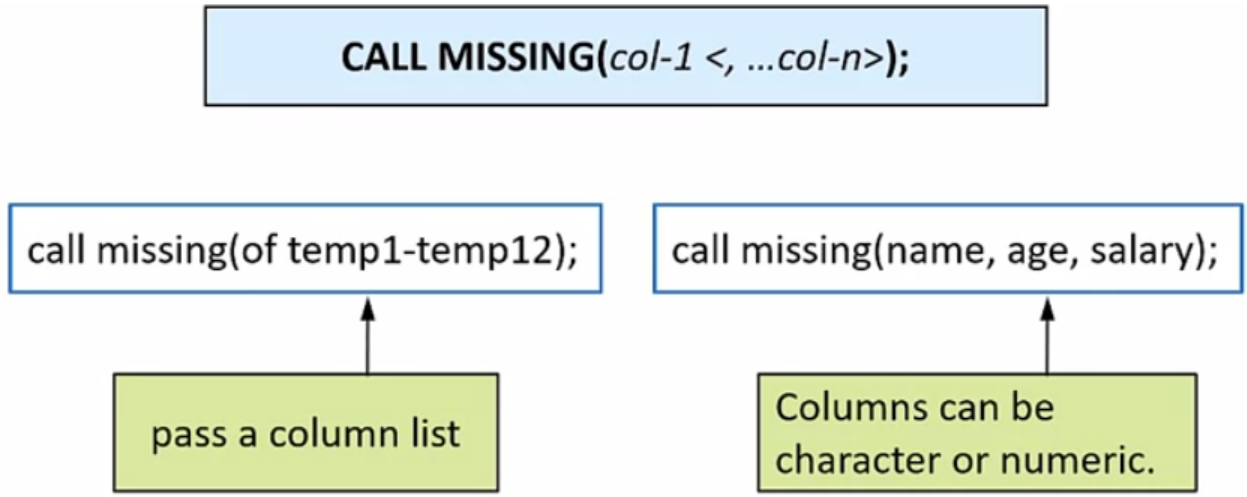

Predicted Populations for Years 2018 to 2022
for 2017 Country Populations Greater than 1 Billion
1,440,000,000
1,420,000,000
1,400,000,000
1,380,000,000
1,360,000,000
PopPredicted
2018
2019
2020
2021
2022
Year
CountryCodeName
CHN-China
IND-India


FUNCTION(argument-1 <, ...argument-n>);
accepts arguments
performs operations
returns values


Character
LENGTH FIND CAT
SCAN ANYALPHA CATS
SUBSTR ANYDIGIT CATX
TRANWRD ANYPUNCT LOWCASE
STRIP COMPBL PROPCASE
TRIM COMPRESS UPCASE
Date and Time
WEEKDAY YEAR DATEPART
DAY YRDIF TIMEPART
MONTH INTCK MDY
QTR INTNX TODAY
Truncation
ROUND CEIL
INT FLOOR
Special
INPUT PUT
Descriptive Statistics
SUM RANGE LARGEST
MEAN MIN N
MEDIAN MAX NMISS


COUNT
COUNTC
COUNTW
LAG
FINDC
FINDW


CALL routine(argument-1 <, ...argument-n>);
accepts arguments
performs operations
alters variable values
performs system functions




```
FirstPrevDay =lag1(TavgC);
SecondPrevDay=lag2(TavgC);
ThirdPrevDay =lag3(TavgC);
FourthPrevDay=lag4(TavgC);
```

City	Date	TavgC	FirstPrevDay	SecondPrevDay	ThirdPrevDay	FourthPrevDay
Beijing	01/01/2017	-4.4	.	.	.	.
Beijing	01/02/2017	-1.1	-4.4	.	.	.
Beijing	01/03/2017	-1.7	-1.1	-4.4	.	.
Beijing	01/04/2017	0.0	-1.7	-1.1	-4.4	.
Beijing	01/05/2017	-1.1	0.0	-1.7	-1.1	-4.4
Beijing	01/06/2017	0.6	-1.1	0.0	-1.7	-1.1
Beijing	01/07/2017	1.1	0.6	-1.1	0.0	-1.7

```
*************************************************************;
*  Retrieving Previous Values with the LAG Function         *;
*************************************************************;

data work.china_temps1;
   set pg3.weather_china_daily2017(keep=City Date TavgC);
   by City;
   FirstPrevDay =lag1(TavgC);
   SecondPrevDay=lag2(TavgC);
```

```
   ThirdPrevDay =lag3(TavgC);
   FourthPrevDay=lag4(TavgC);
run;

title 'Using LAG1 through LAG4 Functions';
proc print data=work.china_temps1 noobs;
run;
title;

*************************************************************;
*  Demo                                    *;
*  1) Highlight and run the DATA step. View the output table   *;
*     and notice that the first 365 rows contain the daily     *;
*     average temperatures for Beijing and the last 365 rows   *;
*     contain the daily average temperatures for Shanghai.     *;
*  2) Uncomment the two assignment statements.                 *;
*       TavgCPrevDay=lag1(TavgC);                              *;
*       TempIncrease=TavgC-TavgCPrevDay;                       *;
*  3) Run the DATA step and view the output table. Notice the  *;
*     values of TavgCPrevDay and TempIncrease. Specifically,   *;
*     look at the value of TavgCPrevDay for the first row of   *;
*     Shanghai data (row 366). The last temperature for        *;
*     Beijing is being used as the previous value for the      *;
*     first temperature of Shanghai.                           *;
*  4) Add a BY statement and a conditional statement to        *;
*     correct the previous value anytime that there is a       *;
*     switch to a new city.                                    *;
*       by City;                                               *;
*       TavgCPrevDay=lag1(TavgC);                              *;
```

```
*      if first.City=1 then TavgCPrevDay=.;                *;
*      TempIncrease=TavgC-TavgCPrevDay;                    *;
*  5) Run the DATA step and view the output table. Confirm     *;
*     that the first row of Shanghai data (row 366) contains   *;
*     a missing value for the previous temperature.           *;
*  6) Run the ODS statements, the PROC MEANS step, and the     *;
*     PROC SGPLOT step to determine the biggest difference in  *;
*     daily average temperature between consecutive days.      *;
*     In the HTML results, place your cursor on data points to *;
*     see a tooltip of Date and TempIncrese.                   *;
*     - The biggest decrease in temperature (-8.9) occurred in *;
*       Beijing on 10/2/2017.                                  *;
*     - The biggest increase in temperature (7.2) occurred in  *;
*       Beijing on 6/7/2017.                                   *;
*     - The biggest decrease in temperature (-10.5) occurred   *;
*       in Shanghai on 2/20/2017.                              *;
*     - The biggest increase in temperature (8.9) occurred in  *;
*       Shanghai on 2/19/2017.                                 *;
*************************************************************;

data work.china_temps;
   set pg3.weather_china_daily2017(keep=City Date TavgC);
   by City;
   TavgCPrevDay=lag1(TavgC);
   if first.city = 1 then TavgCPrevDay=.;
   TempIncrease=TavgC-TavgCPrevDay;
run;
```

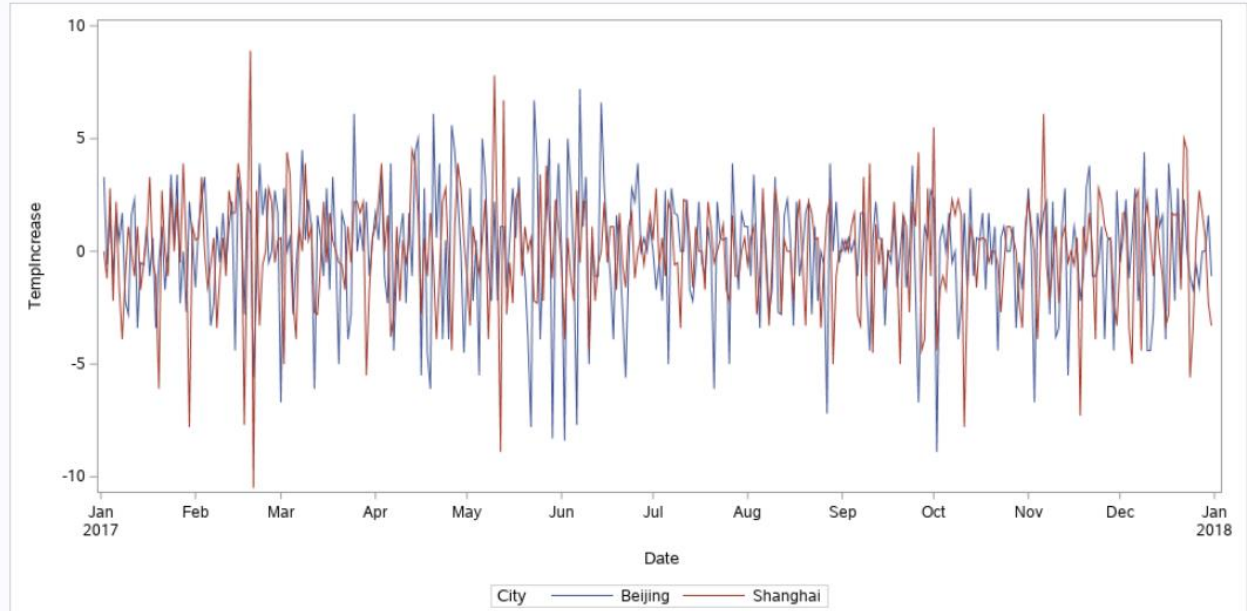
Total rows: 730 Total columns: 5

	City	Date	TavgC	TavgCPrevDay	TempIncrease
363	Beijing	12/29/2017	-3.3	-3.3	0
364	Beijing	12/30/2017	-1.7	-3.3	1.6
365	Beijing	12/31/2017	-2.8	-1.7	-1.1
366	Shanghai	01/01/2017	10.6	.	.
367	Shanghai	01/02/2017	10.6	10.6	0
368	Shanghai	01/03/2017	9.4	10.6	-1.2

```
ods html path="&pathout" file='china_temps.html';
proc means data=work.china_temps;
   class City;
   var TempIncrease;
run;

ods graphics / width=10in height=5in imagemap=on tipmax=800;
proc sgplot data=work.china_temps;
   series x=Date y=TempIncrease / group=City tip=(Date TempIncrease);
run;
ods html close;
```

Analysis Variable : TempIncrease						
City	N Obs	N	Mean	Std Dev	Minimum	Maximum
Beijing	365	364	0.0043956	2.7308325	-8.9000000	7.2000000
Shanghai	365	364	-0.0153846	2.4823864	-10.5000000	8.9000000



Create three additional variables.

Stock	Date	Open	Open1MnthBack	Open2MnthBack	Open3MnthAvg
ABC Company	02JAN2010	104.50	.	.	104.50
ABC Company	02FEB2010	100.00	104.50	.	102.25
ABC Company	02MAR2010	104.19	100.00	104.50	102.90
ABC Company	01APR2010	103.87	104.19	100.00	102.69
ABC Company	01MAY2010	115.94	103.87	104.19	108.00

```
Open3MnthAvg=mean(Open,Open1MnthBack,Open2MnthBack);
```

```
**********************************************************;
*  Activity 2.02                                         *;
**********************************************************;
*  1) Complete the assignment statement for Open1MnthBack, *;
*     which is equal to the Open value from one previous   *;
```

```
*    month.                                  *;
*  2) Complete the assignment statement for Open2MnthBack, *;
*    which is equal to the Open value from two previous   *;
*    months.                                 *;
*  3) Run the program and view the results.              *;
*  4) What is the three-month average (Open3MnthAvg) for   *;
*    02MAR2010?                              *;
************************************************************;

data work.stockmovingavg;
   set pg3.stocks_ABC(drop=Close);
   Open1MnthBack=lag1(Open);
   Open2MnthBack=lag2(Open);
   Open3MnthAvg=mean(Open,Open1MnthBack,Open2MnthBack);
   format Open3MnthAvg 8.2;
run;

title 'Three Month Moving Average on Opening Stock Price';
proc print data=work.stockmovingavg noobs;
run;
title;
```

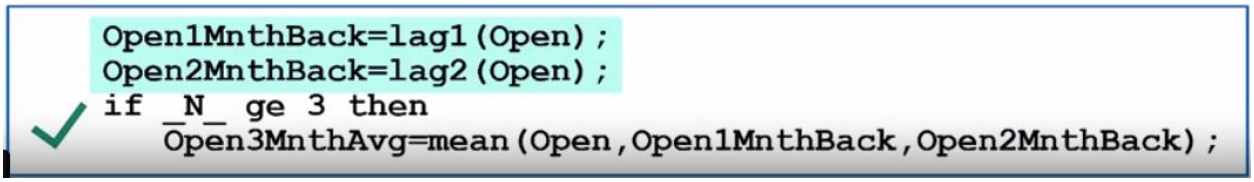
Three Month Moving Average on Opening Stock Price

Stock	Date	Open	Open1MnthBack	Open2MnthBack	Open3MnthAvg
ABC Company	02JAN2010	104.50	.	.	104.50
ABC Company	02FEB2010	100.00	104.50	.	102.25
ABC Company	02MAR2010	104.19	100.00	104.50	102.90
ABC Company	01APR2010	103.87	104.19	100.00	102.69
ABC Company	01MAY2010	115.94	103.87	104.19	108.00
ABC Company	01JUN2010	117.37	115.94	103.87	112.39
ABC Company	01JUL2010	116.00	117.37	115.94	116.44
ABC Company	03AUG2010	134.00	116.00	117.37	122.46
ABC Company	01SEP2010	113.00	134.00	116.00	121.00
ABC Company	01OCT2010	125.06	113.00	134.00	124.02
ABC Company	02NOV2010	148.44	125.06	113.00	128.83
ABC Company	01DEC2010	163.50	148.44	125.06	145.67
ABC Company	04JAN2011	185.00	163.50	148.44	165.65
ABC Company	01FEB2011	184.50	185.00	163.50	177.67
ABC Company	01MAR2011	169.50	184.50	185.00	179.67

Stock	Date	Open	Open1MnthBack	Open2MnthBack	Open3MnthAvg
ABC Company	02JAN2010	104.50	.	.	104.50
ABC Company	02FEB2010	100.00	104.50	.	102.25
ABC Company	02MAR2010	104.19	100.00	104.50	102.90

Stock	Date	Open	Open3MnthAvg
ABC Company	02JAN2010	104.50	.
ABC Company	02FEB2010	100.00	.
ABC Company	02MAR2010	104.19	104.19
ABC Company	01APR2010	103.87	104.03
ABC Company	01MAY2010	115.94	108.00

```
if _N_ ge 3 then
   Open3MnthAvg=mean(Open,lag1(Open),lag2(Open));
```



```
Open1MnthBack=lag1(Open);
Open2MnthBack=lag2(Open);
if _N_ ge 3 then
   Open3MnthAvg=mean(Open,Open1MnthBack,Open2MnthBack);
```

COUNT(*string*, *substring* <, *modifier(s)*>**)**	Counts the number of times that a specified substring appears within a character string.
COUNTC(*string*, *character-list* <, *modifier(s)*>**)**	Counts the number of characters in a string that appear or do not appear in a list of characters.
COUNTW(*string* <, *delimiter(s)*> <, *modifier(s)*>**)**	Counts the number of words in a character string.

COUNT(*string*, *substring* <, *modifier(s)*>**)**

```
NumEF=count(Narrative,'EF');
```

COUNTW(*string* <, *delimiter(s)*> <, *modifier(s)*>**)**

```
NumWord=countw(Narrative,' ');
```

If no delimiters are specified, a default list is used, which includes these:

blank ! $ % & ( ) * + , - . / ; < ^ |

FIND(*string*, *substring* <, *modifier(s)*> <, *start-position*>**)**	Returns the starting position where a substring is found in a string.
FINDC(*string*, *character-list* <, *modifier(s)*> <, *start-position*>**)**	Returns the starting position where a character from a list of characters is found in a string.
FINDW(*string*, *word*, <, *delimiter(s)*> <, *modifier(s)*> <, *start-position*>**)**	Returns the starting position of a word in a string or the number of the word in a string.

FINDW(*string*, *word*, <, *delimiter(s)*> <, *modifier(s)*> <, *start-position*>**)**

```
EFWordNum=findw(Narrative,'EF','012345- .,','e');
```

- counts the number of words scanned until the specified word is found
- returns the number of the word in the string, not the starting position

FINDW(*string*, *word*, <, *delimiter(s)*> <, *modifier(s)*> <, *start-position*>**)**

```
EFWordNum=findw(Narrative,'EF','012345- .,','e');
```

- counts the number of words scanned until the specified word is found
- returns the number of the word in the string, not the starting position

```
**************************************************************;
* Counting and Finding Words with Character Functions        *;
**************************************************************;
```

```
************************************************************;
* Demo                                    *;
* 1) Highlight and run the DATA step and the PROC PRINT step. *;
*    View the results and notice the values of Narrative.     *;
* 2) Uncomment the two assignment statements relating to the  *;
*    COUNT functions. Run the DATA step and the PROC PRINT    *;
*    step. View the results. Verify the values of NumEF and   *;
*    NumWord.                                 *;
*      NumEF=count(Narrative,'EF');                    *;
*      NumWord=countw(Narrative,' ');                  *;
* 3) Uncomment the two assignment statements relating to the  *;
*    FIND functions. Run the DATA step and the PROC PRINT     *;
*    step. View the results. Notice that EFWordNum is equal   *;
*    to EFStartPos anytime the first occurrence of EF is      *;
*    followed by a hyphen and that EFWordNum is equal to 0    *;
*    anytime the first occurrence of EF is followed by a      *;
*    number.                                  *;
*      EFStartPos=find(Narrative,'EF');                *;
*      EFWordNum=findw(Narrative,'EF');                *;
* 4) Modify the EFWordNum assignment statement to add a third *;
*    argument that includes a set of delimiters that          *;
*    separates words. Run the DATA step and the PROC PRINT    *;
*    step. View the results. Notice that EFWordNum is now     *;
*    equal to EFStartPos for all rows except row 240.         *;
*      EFWordNum=findw(Narrative,'EF','012345- .,');         *;
* 5) Modify the EFWordNum assignment statement to add a       *;
*    fourth argument that returns the number of the word      *;
*    instead of the starting position. Run the DATA step and  *;
*    the PROC PRINT step. View the results. Notice that       *;
```

```
*    EFWordNum is now the number of the word, so the number   *;
*    is smaller than EFStartPos.                              *;
*     EFWordNum=findw(Narrative,'EF','012345- .,','e');      *;
*  6) Uncomment the conditional statement, which, if true,     *;
*    scans the narrative for the word after the EF word. Run  *;
*    the DATA step and the PROC PRINT step. View the results. *;
*    Verify that AfterEF contains the word after the first    *;
*    occurrence of the EF word.                               *;
*     if EFWordNum>0 then                                     *;
*       AfterEF=scan(Narrative,EFWordNum+1,'012345- .,');   *;
*  7) Run the PROC FREQ step and the PROC MEANS step. View the *;
*    results.                                                 *;
*    - On average, EF is referenced 0.88 times within a       *;
*     narrative with a range of 0 to 6 times.                 *;
*    - On average, 101.7 words are written in a narrative     *;
*     with a range of 3 to 676 words.                         *;
*    - Tornado is the word that tends to follow the EF value. *;
*  8) Self-study: Refer to program p302d03 for examples of     *;
*    using FIND and FINDW functions with a DO loop to find    *;
*    all occurrences of EF within a narrative.                *;
*************************************************************;

data work.narrative;
   set pg3.tornado_2017narrative;
   /* COUNT Functions */
   *NumEF=count(Narrative,'EF');
   *NumWord=countw(Narrative,' ');
   /* FIND Functions */
   *EFStartPos=find(Narrative,'EF');
```

```
   *EFWordNum=findw(Narrative,'EF');
   /* SCAN Function */
   *if EFWordNum>0 then
     AfterEF=scan(Narrative,EFWordNum+1,'012345- .,');
run;

proc print data=work.narrative;
run;

proc means data=work.narrative maxdec=2;
   var NumEF NumWord ;
run;

proc freq data=work.narrative order=freq;
   tables AfterEF;
run;
```

Obs	State	County	BeginDate	BeginTime	Narrative
1	GA	COOK CO.	22JAN2017	335	This is a continuation of the northeast Brooks county tornado. The tornado then continued into Cook County. Still at EF-3 strength, it swept about 35 manufactured homes into a pile of rubble at the far end of the Sunshine Acres mobile home park. Seven people lost their lives. The tornado then went on to destroy about two thirds of a brick home on Val Del Road, collapsing in two walls and removing most of the second story. Another home built of concrete blocks was destroyed. A nearby farm had several concrete anchors for a large metal structure pulled from the ground. Max winds were estimated near 140 mph. Damage cost was estimated.
2	GA	DOUGHERTY CO.	22JAN2017	1515	A large, long-track tornado touched down near Dougherty/Baker Co. line and traveled over 70 miles across Dougherty, Worth, Turner, and Wilcox Counties in South Georgia. The tornado lifted just east of Abbeville. The tornado caused significant damage along the track, resulting in 5 fatalities in Albany. Severe tree damage was observed along the entire path which was up to 1.2 miles wide. In many spots, 90 to 100 percent of the trees in the path were uprooted or snapped. In Dougherty County, the tornado touched down on Tarva Road. By the time it reached Newton, the tornado was approximately 1.25 miles wide. There was extensive tree damage and some minor to moderate damage to a few homes in this area, consistent with EF2 damage. The tornado moved through the Radium Springs area, destroying nearly every tree in its path and causing EF2 damage to several houses. Most houses in this area had significant damage from falling trees. The tornado then moved through several mobile home parks just west of U.S. 319, destroying many mobile homes and causing the 4 fatalities. Damage consistent with an EF3 tornado was observed just east of U.S. 319. The tornado caused a large portion of a warehouse at the Proctor and Gamble Plant to collapse and tossed several semi-trailers across Mock Road. Additional EF3 damage was observed at the Marine Corp Logistics Base, where multiple anchored double-wide trailers were completely destroyed. In addition, several concrete light poles were snapped near the base, and a large solid concrete building had its solid concrete roof shifted more than 2 inches. A well-built concrete block church on Sylvester Rd was demolished with only parts of a few walls remaining. The estimated wind speed at this point is 150 mph, the highest analyzed along the track. EF3 damage was also observed on Harris Road where a cement block church was destroyed. Damage estimates exceeded $300 million according to a media article citing the Dougherty County Commissioner.
3	MS	FORREST CO.	21JAN2017	347	This tornado began along Purvis-Oloh Road, about 5 miles west northwest of Purvis. It tracked northeast across portions of Lamar County causing mainly tree damage, uprooting and snapping softwood and hardwood trees. It caused some minor to moderate structural damage as well. The tornado crossed Old Highway 11, continuing to cause mainly tree damage, although at least a few structures were also damaged. The tornado began to gain strength as it reached Slade Road, where some homes received roof damage. As the tornado crossed Sullivan Kilrain Road, additional homes received significant roof and structural damage, especially on Carter Circle and Tatum Camp Road. Comprehensive assessments from Lamar County emergency management officials count 26 homes in the county being destroyed or with major damage and 52 homes receiving minor damage. The tornado continued to gain strength as it crossed into Forrest County. There it struck a subdivision along Nellwood Drive, Lakeland Drive and Crestwood Drive and caused significant damage to many homes. Several homes received significant roof and structural damage. One home sustained significant damage and also was the site of one of the fatalities. As the tornado continued to track northeast, it caused extensive tree damage and powerline damage. It struck a church along Helveston Road, which suffered damage to the top floor. As the tornado approached William Carey College, it intensified to EF3 strength, causing damage to numerous buildings on campus. The tornado then affected a mobile home park, downing trees and causing damage to mobile homes. Here, two more fatalities occurred. It then caused structural damage to several homes, churches and businesses on James Street and Alcorn Road. Another fatality occurred on Alcorn Road, when a tree fell on a home. The tornado continued to track northeast, crossing the Leaf River and headed into Petal. Here the tornado got very wide and continued to cause EF2 type damage to businesses along Main Street and into the neighborhoods on the southern side of the city. Extensive tree damage occurred and countless homes had minor to major roof damage in the neighborhood south of Hillcrest Loop. As the tornado reached Sun Circle, it intensified to EF3 strength again and caused significant damage to a few homes as well as moderate damage to many homes in the neighborhood. Beyond the subdivision along Sun Circle, the tornado caused tree damage again before reaching Evelyn Gandy|Parkway. It caused structural damage to an AT&T store by lifting the roof of the strip mall. It caused additional roof damage to several homes and a church along the Parkway as well as along Springridge Road and Corinth Road. As the tornado tore across Shawnee Trail, it caused impressive tree damage, which included snapping and uprooting hardwood and softwood trees. As the tornado tracked just south of|HDR Lane, it took down two metal electrical transmission lines. The tornado then destroyed a house along Macedonia Road, as well as causing additional roof to other homes and taking down numerous trees. Additional impressive tree damage occurred along Old Richton Road and Tyroby Lane. The tornado continued to down trees as it|tracked across Old Richton Road into Perry County. Comprehensive assessments from Forrest County emergency management officials count 499 homes in the county being destroyed or with major damage and 632 homes receiving minor damage.||The tornado path length in just Lamar and Forrest counties was 24.2 miles, although the entire path of the tornado in total was 31.3 miles. Along the entire tornado path the total number of homes destroyed or receiving major damage is estimated to be 531 with the number of homes having minor damage estimated at 689. Maximum estimated winds were 145 mph. The maximum path width was 900 yards, which occurred in the Petal area of Forrest County. This tornado affected a large number of forested areas. Approximately 1570 forested acres were damaged, with 1453 acres being privately owned. In total, 4320 acres were damaged from this tornado. The total economic impact was $410,784 in Lamar, Forrest and Perry counties. 777 total acres were damaged in Lamar County, and 400 in Forrest.
4	TX	VAN ZANDT CO.	29APR2017	1708	This tornado began nearly as the last tornado dissipated to the south and west. The parent supercell cycled another tornado about a mile to the east and northeast of the previous storm. This tornado quickly grew to a large tornado, becoming slightly less than one mile wide at its widest point. The tornado was at the strongest near Interstate 20 and FM 17 just north of Canton. The survey crews found continuous damage between Canton and Fruitvale, and then additional damage as far north as Emory, |and Lake Fork. Several homes, businesses, and farm buildings were damaged or destroyed in the 40 mile continuous damage path. This tornado occurred for over an hour, and spanned most of Van Zandt, and nearly all of Rains Counties during the 80 minute track.

```
data work.narrative;
   set pg3.tornado_2017narrative;
```

```
   /* COUNT Functions */
   NumEF=count(Narrative,'EF');
   NumWord=countw(Narrative,' ');
   /* FIND Functions */
   EFStartPos=find(Narrative,'EF');
   EFWordNum=findw(Narrative,'EF','012345- .,','e');
   /* SCAN Function */
   if EFWordNum>0 then
     AfterEF=scan(Narrative,EFWordNum+1,'012345- .,');
run;

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

Obs	State	County	BeginDate	BeginTime	Narrative	NumEF	NumWord	EFStartPos	EFWordNum	AfterEF
1	GA	COOK CO.	22JAN2017	335	This is a continuation of the northeast Brooks county tornado. The tornado then continued into Cook County. Still at EF-3 strength, it swept about 35 manufactured homes into a pile of rubble at the far end of the Sunshine Acres mobile home park. Seven people lost their lives. The tornado then went on to destroy about two thirds of a brick home on Val Del Road, collapsing in two walls and removing most of the second story. Another home built of concrete blocks was destroyed. A nearby farm had several concrete anchors for a large metal structure pulled from the ground. Max winds were estimated near 140 mph. Damage cost was estimated.	1	112	119	20	strength
2	GA	DOUGHERTY CO.	22JAN2017	1515	A large, long-track tornado touched down near Dougherty/Baker Co. line and traveled over 70 miles across Dougherty, Worth, Turner, and Wilcox Counties in South Georgia. The tornado lifted just east of Abbeville. The tornado caused significant damage along the track, resulting in 5 fatalities in Albany. Severe tree damage was observed along the entire path which was up to 1.2 miles wide. In many spots, 90 to 100 percent of the trees in the path were uprooted or snapped. In Dougherty County, the tornado touched down on Tarva Road. By the time it reached Newton, the tornado was approximately 1.25 miles wide. There was extensive tree damage and some minor to moderate damage to a few homes in this area, consistent with EF2 damage. The tornado moved through the Radium Springs area, destroying nearly every tree in its path and causing EF2 damage to several houses. Most houses in this area had significant damage from falling trees. The tornado then moved through several mobile home parks just west of U.S. 319, destroying many mobile homes and causing the 4 fatalities. Damage consistent with an EF3 tornado was observed just east of U.S. 319. The tornado caused a large portion of a warehouse at the Proctor and Gamble Plant to collapse and tossed several semi-trailers across Mock Road. Additional EF3 damage was observed at the Marine Corp Logistics Base, where multiple anchored double-wide trailers were completely destroyed. In addition, several concrete light poles were snapped near the base, and a large solid concrete building had its solid concrete roof shifted more than 2 inches. A well-built concrete block church on Sylvester Rd was demolished with only parts of a few walls remaining. The estimated wind speed at this point is 150 mph, the highest analyzed along the track. EF3 damage was also observed on Harris Road where a cement block church was destroyed. Damage estimates exceeded $300 million according to a media article citing the Dougherty County Commissioner.	5	327	725	120	damage
3	MS	FORREST CO.	21JAN2017	347	This tornado began along Purvis-Oloh Road, about 5 miles west northwest of Purvis. It tracked northeast across portions of Lamar County causing mainly tree damage, uprooting and snapping softwood and hardwood trees. It caused some minor to moderate structural damage as well. The tornado crossed Old Highway 11, continuing to cause mainly tree damage, although at least a few structures were also damaged. The tornado began to gain strength as it reached Slade Road, where some homes received roof damage. As the tornado crossed Sullivan Kilrain Road, additional homes received significant roof and structural damage, especially on Carter Circle and Tatum Camp Road. Comprehensive assessments from Lamar County emergency management officials count 26 homes in the county being destroyed or with major damage and 52 homes receiving minor damage. The tornado continued to gain strength as it crossed into Forrest County. There it struck a subdivision along Nellwood Drive, Lakeland Drive and Crestwood Drive and caused significant damage to many homes. Several homes received significant roof and structural damage. One home sustained significant damage and also was the site of one of the fatalities. As the tornado continued to track northeast, it caused extensive tree damage and powerline damage. It struck a church along Helveston Road, which suffered damage to the top floor. As the tornado approached William Carey College, it intensified to EF3 strength, causing damage to numerous buildings on campus. The tornado then affected a mobile home park, downing trees and causing damage to mobile homes. Here, two more fatalities occurred. It then caused structural damage to several homes, churches and businesses on James Street and Alcorn Road. Another fatality occurred on Alcorn Road, when a tree fell on a home. The tornado continued to track northeast, crossing the Leaf River and headed into Petal. Here the tornado got very wide and continued to cause EF2 type damage to businesses along Main Street and into the neighborhoods on the southern side of the city. Extensive tree damage occurred and countless homes had minor to major roof damage in the neighborhood south of Hillcrest Loop. As the tornado reached Sun Circle, it intensified to EF3 strength again and caused significant damage to a few homes as well as moderate damage to many homes in the neighborhood. Beyond the subdivision along Sun Circle, the tornado caused tree damage again before reaching Evelyn Gandy|Parkway. It caused structural damage to an AT&T store by lifting the roof of the strip mall. It caused additional roof damage to several homes and a church along the Parkway as well as along Springridge Road and Corinth Road. As the tornado tore across Shawnee Trail, it caused impressive tree damage, which included snapping and uprooting hardwood and softwood trees. As the tornado tracked just south of|HDR Lane, it took down two metal electrical transmission lines. The tornado then destroyed a house along Macedonia Road, as well as causing	3	676	1448	222	strength

```
proc means data=work.narrative maxdec=2;
   var NumEF NumWord ;
run;
```

```
proc freq data=work.narrative order=freq;
   tables AfterEF;
run;
```

The MEANS Procedure

Variable	N	Mean	Std Dev	Minimum	Maximum
NumEF	500	0.88	0.96	0.00	6.00
NumWord	500	101.70	81.93	3.00	676.00

The FREQ Procedure

AfterEF	Frequency	Percent	Cumulative Frequency	Cumulative Percent
tornado	172	58.11	172	58.11
damage	29	9.80	201	67.91
with	19	6.42	220	74.32
intensity	7	2.36	227	76.69
in	6	2.03	233	78.72
strength	6	2.03	239	80.74
The	4	1.35	243	82.09
along	4	1.35	247	83.45
and	3	1.01	250	84.46
as	3	1.01	253	85.47

```
************************************************************;
*  LESSON 2, PRACTICE 1                                    *;
************************************************************;
```

Practice Level 1: Using the LAG Function

TOTAL POINTS 4

1.

Question 1

If necessary, start SAS Studio before you begin. If you restarted your SAS session, submit your **libname.sas** program to access the practice data.

The **pg3.np_2016traffic** table contains monthly traffic counts for national parks for the year 2016. There are 12 rows for each value of **ParkCode**, and each row gives the traffic count for months 1 through 12. Calculate the change in traffic count between consecutive months for each park.

Open the **p302p01.sas** program in the **practices** folder and review the code.

Which column is named in the BY statement, and what 2 variables are available as a result?

ParkCode is named in the BY statement, making **FIRST.ParkCode** and **LAST.ParkCode** available.

- Run the program to view the 2016 traffic count data. The report includes the numeric columns **Year**, **Month**, and **TrafficCount**.
- Return to the code.
- In an assignment statement, use the LAG function to create the column **PrevMthTC**, which is the previous value of **TrafficCount**.
- In another assignment statement, create the column **OneMthChange**, which is **TrafficCount** minus **PrevMthTC**.
- Run the program and view the results.

  What is the value of **PrevMthTC** for row 13, the first row for the park code *ACAD*? **Note**: Type the value as it is shown in the results.

  In order to calculate the change in traffic count between consecutive months for each park, the first row for each park should be set to a missing value.
- Before the **OneMthChange** assignment statement, add an IF/THEN statement that changes the **PrevMthTC** value to missing for the first occurrence of a park code.
- Run the program and verify the results.

  What is the value of **OneMthChange** when **Month** is equal to 1 for **ParkCode** values *ACAD* and *ALPO*?

```
data work.ParkTraffic2016;
   set pg3.np_2016traffic;
   by ParkCode;
         PrevMthTC=lag1(TrafficCount);
         if first.ParkCode = 1 then PrevMthTC=.;
   OneMthChange=TrafficCount-PrevMthTC;
run;


title '2016 National Park Traffic Counts';
proc print data=work.ParkTraffic2016;
run;
```

2016 National Park Traffic Counts

Obs	ParkCode	Year	Month	TrafficCount	PrevMthTC	OneMthChange
1	ABLI	2016	1	2159.00	.	.
2	ABLI	2016	2	2057.00	2159.00	-102.00
3	ABLI	2016	3	4630.00	2057.00	2573.00
4	ABLI	2016	4	6602.00	4630.00	1972.00
5	ABLI	2016	5	6459.00	6602.00	-143.00
6	ABLI	2016	6	7739.00	6459.00	1280.00
7	ABLI	2016	7	14016.00	7739.00	6277.00
8	ABLI	2016	8	10237.00	14016.00	-3779.00
9	ABLI	2016	9	5437.00	10237.00	-4800.00
10	ABLI	2016	10	7305.00	5437.00	1868.00
11	ABLI	2016	11	4134.00	7305.00	-3171.00
12	ABLI	2016	12	2265.00	4134.00	-1869.00
13	ACAD	2016	1	5979.00	.	.
14	ACAD	2016	2	5705.00	5979.00	-274.00
15	ACAD	2016	3	8082.00	5705.00	2377.00

**********;

* LESSON 2, PRACTICE 2 *;

**********;

/*Practice Level 2: Using the COUNT and FINDW Functions

If necessary, start SAS Studio before you begin.

If you restarted your SAS session, submit your libname.sas program to access the practice data.

The pg3.np_grandcanyon table contains comments regarding Grand Canyon National Park.

The canyon consists of the North Rim and the South Rim. Determine how many times the word South appears in each comment.

Also, retrieve the word after the first occurrence of the word South in each comment.

Open the p302p02.sas program in the practices folder.

Run the program to view the Grand Canyon comments.

How many rows are in the report?

Use the COUNT function to create a column named NumSouth that is equal to the number of times the word South

appears in each comment. Use the modifier i to ignore case.

Subset the data to include only the rows that contain a comment with the word South.

Run the program.

How many rows contain a comment with the word South?

Return to the code tab to modify the program.

Use the FINDW function to create a column named SouthWordPos that is equal to the word number for the first occurrence

of the word South in each comment.

Specify the space and period as the delimiters that separate words.

Use the modifier i to ignore case and the modifier e to return the word number instead of the starting position.

Use the SCAN function to create a column named AfterSouth that is equal to the word after the first occurrence

of the word South. Use the same delimiters in the SCAN function as the FINDW function.

Run the program and view the results.

What word occurs most often after the word South?

*/

```
data work.SouthRim;
   set pg3.np_grandcanyon;
   /* COUNT Functions */
   NumSouth=count(Comments,'South','i');
   NumWord=countw(Comments,' ','i');
   /* FIND Functions */
   SouthStartPos=find(Comments,'South');
   SouthWordPos=findw(Comments,'South','012345- .,','ie');
   /* SCAN Function */
   if SouthWordPos>0 then
```

```
    AfterSouth=scan(Comments,SouthWordPos+1,'012345- .,');
run;


title 'Grand Canyon Comments Regarding South Rim';
proc print data=work.SouthRim;
run;
title;
```

Grand Canyon Comments Regarding South Rim

Obs	CollectedDate	Comments	NumSouth	NumWord	SouthStartPos	SouthWordPos	AfterSouth
1	01/01/2019	Non-reportable, non-recreation, and bus numbers are estimated due to partial government shutdown. The park was open, but measured data was not collected for the entire month. Estimates were produced using January 2019's most solid pieces of data, the South Entrance and Desert View traffic counters. The counter showed a 13% decrease in traffic at South Entrance and a 10% decrease in traffic at Desert View compared to January 2018. This decrease was applied across South District and Desert View January 2018 stats where needed to produce the January 2019 stats...Mather Campground numbers are estimated due to partial government shutdown. Estimate produced from average number of nights/campers from January 26-31, 2019.	3	110	252	41	Entrance
2	12/01/2018	Traffic data for South Entrance is estimated. Traffic counter was inoperable from December 1-7 due to new counter installation. Estimated data generated by taking daily average from December 8-31...Non-reportable, non-recreation, and bus numbers are estimated due to partial government shutdown. The park was open, but measured data was not collected. Estimates were produced using December 2018's most solid piece of data, the Desert View traffic counter. The counter showed a 3% decrease in traffic compared to December 2017. This was applied across December 2017 stats where needed to produce the December 2018 stats...Mather Campground numbers are estimated due to partial government shutdown. Estimate produced from average number of nights/campers from December 1-21.	1	112	18	4	Entrance
3	11/01/2018	South Rim traffic count estimated due to road construction causing traffic counter to be inoperable during the month of November.	1	20	1	1	Rim
4	10/01/2018	South Entrance and North Rim traffic is estimated due to traffic counters being down.	1	14	1	1	Entrance
5	09/01/2018	South Entrance traffic count is estimated due to counters being down due to road construction.	1	15	1	1	Entrance
6	08/01/2018	Traffic for South Entrance estimated due to traffic counter being out of commission due to road construction.	1	17	13	3	Entrance
7	07/01/2018	South Entrance traffic count estimated due to counter being out of commission due to road construction.	1	16	1	1	Entrance
8	06/01/2018	Traffic count at South Entrance estimated because of broken counters caused by road construction.	1	14	18	4	Entrance

```
*Solution;
data work.SouthRim;
   set pg3.np_grandcanyon;
   NumSouth=count(Comments,'South','i');
   if NumSouth>0;
run;


title 'Grand Canyon Comments Regarding South Rim';
proc print data=work.SouthRim;
run;
title;
```

Grand Canyon Comments Regarding South Rim

Obs	CollectedDate	Comments	NumSouth
1	01/01/2019	Non-reportable, non-recreation, and bus numbers are estimated due to partial government shutdown. The park was open, but measured data was not collected for the entire month. Estimates were produced using January 2019's most solid pieces of data, the South Entrance and Desert View traffic counters. The counter showed a 13% decrease in traffic at South Entrance and a 10% decrease in traffic at Desert View compared to January 2018. This decrease was applied across South District and Desert View January 2018 stats where needed to produce the January 2019 stats...Mather Campground numbers are estimated due to partial government shutdown. Estimate produced from average number of nights/campers from January 26-31, 2019.	3
2	12/01/2018	Traffic data for South Entrance is estimated. Traffic counter was inoperable from December 1-7 due to new counter installation. Estimated data generated by taking daily average from December 8-31...Non-reportable, non-recreation, and bus numbers are estimated due to partial government shutdown. The park was open, but measured data was not collected. Estimates were produced using December 2018's most solid piece of data, the Desert View traffic counter. The counter showed a 3% decrease in traffic compared to December 2017. This was applied across December 2017 stats where needed to produce the December 2018 stats...Mather Campground numbers are estimated due to partial government shutdown. Estimate produced from average number of nights/campers from December 1-21.	1
3	11/01/2018	South Rim traffic count estimated due to road construction causing traffic counter to be inoperable during the month of November.	1
4	10/01/2018	South Entrance and North Rim traffic is estimated due to traffic counters being down.	1
5	09/01/2018	South Entrance traffic count is estimated due to counters being down due to road construction.	1
6	08/01/2018	Traffic for South Entrance estimated due to traffic counter being out of commission due to road construction.	1
7	07/01/2018	South Entrance traffic count estimated due to counter being out of commission due to road construction.	1
8	06/01/2018	Traffic count at South Entrance estimated because of broken counters caused by road construction.	1
9	07/01/2016	Non Reportables at South Entrance counts are considerably lower than previous years, and haven't determined the cause(s).	1
10	08/01/2015	estimations for traffic count south entrance. 10/8/15; MRA..final update 12/14/15; MRA	1
11	07/01/2015	Partial Data; Traffic counts estimated for south entrance. 8/7/15..final update 12/14/15; MRA	1
12	06/01/2012	South Entrance lane 5 still not working; N. Rim count repaired on the 20th. Daily averages used to complete month's data...	1
13	05/01/2012	South Entrance and North Rim Traffic Counters not functioning. Counts are estimates based on previous 2 years of May and last month April...	1
14	04/01/2012	South Entrance Counter malfunctioned on lane 5. Hourly averages for operational times were used on 14 days (April 6-20) for 69 hourly counts...	1
15	05/01/2010	North Rim opened in May and shuttle buses began running on the South Rim	1
16	01/01/2010	Severe winter weather closed the Desert View entrance and virtually closed the South Entrance on a number of occasions.	1
17	04/01/2009	Work bagan on the Transportation Plan around Mather Point & the VC: realign South Entrance Road, 3 new visitor parking lots with capacity up to 600 vehicles & 1 parking lot for 40 commercial buses.	1
18	11/01/2008	Nov 15: Hermit Road re-opened on South Rim. Projectstarted in February. Also parking area at Mariposa Point removed...	1
19	06/01/2008	June 2: start of pilot shuttle bus program between Tusayan & South Rim..The # of passengers on buses (1-25)capacity has been increased by the number of individuals (9950) that rode the free shuttle from the town of Tusayan. The shuttle is running every 20 minutes under a trial basis until September 1...FYI: 2,000 parking spots & 6,000 daily vehicles...Jun 4: restrictions on N Kaibab trail were lifted. N Rim stock traffic resumes from Jan 10...	1

```
data work.SouthRim;
   set pg3.np_grandcanyon;
   NumSouth=count(Comments,'South','i');
   if NumSouth>0;
   SouthWordPos=findw(Comments,'South',' .','ei');
   AfterSouth=scan(Comments,SouthWordPos+1,' .');
run;

title 'Grand Canyon Comments Regarding South Rim';
proc print data=work.SouthRim;
run;
title;
```

Grand Canyon Comments Regarding South Rim

Obs	CollectedDate	Comments	NumSouth	SouthWordPos	AfterSouth
1	01/01/2019	Non-reportable. non-recreation, and bus numbers are estimated due to partial government shutdown. The park was open, but measured data was not collected for the entire month. Estimates were produced using January 2019's most solid pieces of data, the South Entrance and Desert View traffic counters. The counter showed a 13% decrease in traffic at South Entrance and a 10% decrease in traffic at Desert View compared to January 2018. This decrease was applied across South District and Desert View January 2018 stats where needed to produce the January 2019 stats...Mather Campground numbers are estimated due to partial government shutdown. Estimate produced from average number of nights/campers from January 26-31, 2019.	3	39	Entrance
2	12/01/2018	Traffic data for South Entrance is estimated. Traffic counter was inoperable from December 1-7 due to new counter installation. Estimated data generated by taking daily average from December 8-31...Non-reportable. non-recreation, and bus numbers are estimated due to partial government shutdown. The park was open, but measured data was not collected. Estimates were produced using December 2018's most solid piece of data, the Desert View traffic counter. The counter showed a 3% decrease in traffic compared to December 2017. This was applied across December 2017 stats where needed to produce the December 2018 stats...Mather Campground numbers are estimated due to partial government shutdown. Estimate produced from average number of nights/campers from December 1-21.	1	4	Entrance
3	11/01/2018	South Rim traffic count estimated due to road construction causing traffic counter to be inoperable during the month of November.	1	1	Rim
4	10/01/2018	South Entrance and North Rim traffic is estimated due to traffic counters being down.	1	1	Entrance
5	09/01/2018	South Entrance traffic count is estimated due to counters being down due to road construction.	1	1	Entrance
6	08/01/2018	Traffic for South Entrance estimated due to traffic counter being out of commission due to road construction.	1	3	Entrance
7	07/01/2018	South Entrance traffic count estimated due to counter being out of commission due to road construction.	1	1	Entrance
8	06/01/2018	Traffic count at South Entrance estimated because of broken counters caused by road construction.	1	4	Entrance
9	07/01/2016	Non Reportables at South Entrance counts are considerably lower than previous years, and haven't determined the cause(s).	1	4	Entrance
10	08/01/2015	estimations for traffic count south entrance. 10/8/15; MRA...final update 12/14/15; MRA	1	5	entrance
11	07/01/2015	Partial Data; Traffic counts estimated for south entrance. 8/7/15..final update 12/14/15; MRA	1	7	entrance
12	06/01/2012	South Entrance lane 5 still not working; N. Rim count repaired on the 20th. Daily averages used to complete month's data...	1	1	Entrance
13	05/01/2012	South Entrance and North Rim Traffic Counters not functioning. Counts are estimates based on previous 2 years of May and last month April...	1	1	Entrance
14	04/01/2012	South Entrance Counter malfunctioned on lane 5. Hourly averages for operational times were used on 14 days (April 6-20) for 69 hourly counts...	1	1	Entrance
15	05/01/2010	North Rim opened in May and shuttle buses began running on the South Rim	1	13	Rim
16	01/01/2010	Severe winter weather closed the Desert View entrance and virtually closed the South Entrance on a number of occasions.	1	13	Entrance
17	04/01/2009	Work bagan on the Transportation Plan around Mather Point & the VC: realign South Entrance Road, 3 new visitor parking lots with capacity up to 600 vehicles & 1 parking lot for 40 commercial buses.	1	14	Entrance
18	11/01/2008	Nov 15: Hermit Road re-opened on South Rim. Projectstarted in February. Also parking area at Mariposa Point removed...	1	7	Rim
19	06/01/2008	June 2: start of pilot shuttle bus program between Tusayan & South Rim..The # of passengers on buses (1-25)capacity has been increased by the number of individuals (9950) that rode the free shuttle from the town of Tusayan. The shuttle is running every 20 minutes under a trial basis until September 1...FYI: 2,000 parking spots & 6,000 daily vehicles...Jun 4: restrictions on N Kaibab trail were lifted. N Rim stock traffic resumes from Jan 10...	1	12	Rim

```
/([2-9]\d\d)-([2-9]\d\d)-(\d{4})/
```

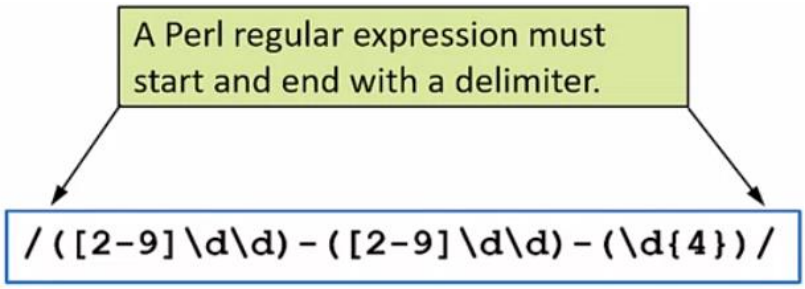
Phone
123-456-7890
234-567-8901
1-345-678-9012
567-8901
789-012-3456
890-123-456
901-234-5678 US

Metacharacter	Behavior
/.../	Forward slash is starting and ending delimiter.
(...)	Parentheses are for grouping.
\d	Matches a digit (0-9).
\D	Matches a non-digit such as letter or special character.
\s	Matches a whitespace character such as space or tab.
\w	Matches a word character (a-z, A-Z, 0-9, or underscore).
\W	Matches a non-word character.
\b	Matches a word boundary (most special characters).
\B	Matches a non-word boundary (letter, digit, or underscore).

Metacharacter	Behavior
.	Matches any character.
[...]	Matches a character in the brackets.
[^...]	Matches a character not in the brackets.

```
/^([2-9]\d\d)-([2-9]\d\d)-(\d{4})$/
```

Metacharacter	Behavior
^	Matches the beginning of the string.
$	Matches the end of the string.
*	Matches the preceding character 0 or more times.
+	Matches the preceding character 1 or more times.
?	Matches the preceding character 0 or 1 times.
{n}	Matches exactly *n* times.
\	Overrides the next metacharacter, such as a (or ?.

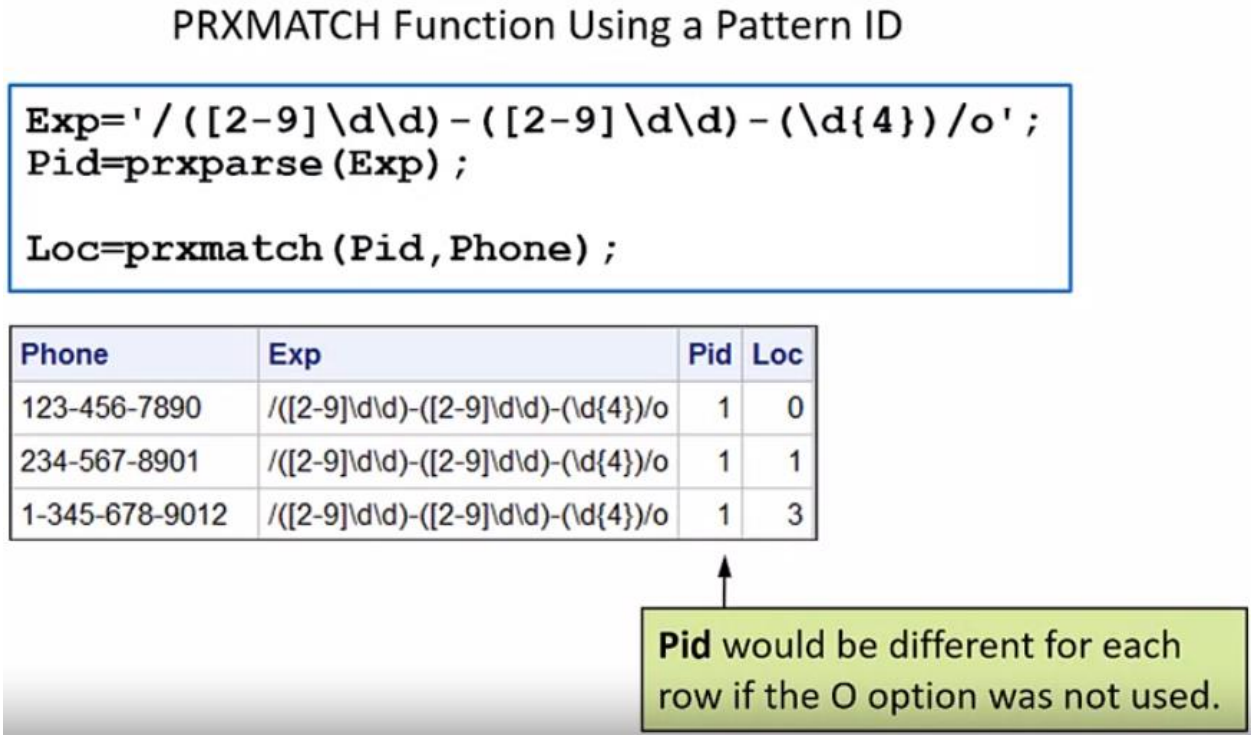


Phone
123-456-7890
234-567-8901
1-345-678-9012
567-8901
789-012-3456
890-123-456
901-234-5678 US

pattern-ID = **PRXPARSE(***perl-regular-expression***);**

```
Pid=prxparse('/([2-9]\d\d)-([2-9]\d\d)-(\d{4})/';
```

```
Exp='/([2-9]\d\d)-([2-9]\d\d)-(\d{4})/o';
Pid=prxparse(Exp);
```




```
*****************************************************************;
*  Validating Data with the PRXMATCH Function                   *;
*****************************************************************;


*Constant;
data work.ValidPhoneNumbers;
   set pg3.phonenumbers_us;
   Loc=prxmatch('/([2-9]\d\d)-([2-9]\d\d)-(\d{4})/',Phone);
run;


proc print data=work.ValidPhoneNumbers noobs;
run;


*Column;
data work.ValidPhoneNumbers;
   set pg3.phonenumbers_us;
```

```
   Exp='/([2-9]\d\d)-([2-9]\d\d)-(\d{4})/o';
   Loc=prxmatch(Exp,Phone);
run;

proc print data=work.ValidPhoneNumbers noobs;
run;

*Pattern ID Number;
data work.ValidPhoneNumbers;
   set pg3.phonenumbers_us;
   Exp='/([2-9]\d\d)-([2-9]\d\d)-(\d{4})/o';
   Pid=prxparse(Exp);
   Loc=prxmatch(Pid,Phone);
run;

proc print data=work.ValidPhoneNumbers noobs;
run;

*****************************************************************;
*  Demo                                        *;
*  1) In the first DATA step, notice the incomplete assignment *;
*    statement.                                  *;
*      Loc=prxmatch('/ /',Phone);                      *;
*  2) Add a Perl regular expression to the first argument of   *;
*    the PRXMATCH function to find valid phone numbers.      *;
*      Loc=prxmatch('/([2-9]\d\d)-([2-9]\d\d)-(\d{4})/',     *;
*               Phone);                            *;
*  3) Highlight and run the first DATA step and PROC PRINT     *;
*    step. Verify that the Loc value represents the starting  *;
```

```
*    location of the 10-digit phone number. Rows 2, 3, and 7  *;
*    should have a Loc value greater than 0.                   *;
*  4) Copy and paste the Loc assignment statement. Modify the  *;
*    statement to create a column named LocStartEnd and to     *;
*    find only values that start and end with the 10-digit     *;
*    number (no leading or trailing text).                     *;
*      LocStartEnd=                                            *;
*        prxmatch('/^([2-9]\d\d)-([2-9]\d\d)-(\d{4})$/',       *;
*               strip(Phone));                                 *;
*  5) Highlight and run the first DATA step and PROC PRINT     *;
*    step. Verify that only row 2 has a LocStartEnd value      *;
*    greater than 0.                                           *;
*  6) Copy and paste the Loc assignment statement. Modify the  *;
*    statement to create a column named LocParen. Alter the    *;
*    expression to find area codes in parentheses. In          *;
*    addition, instead of the first hyphen, there might or     *;
*    might not be a space. There is no longer a hyphen after   *;
*    the area code.                                            *;
*      LocParen=                                               *;
*        prxmatch('/\(([2-9]\d\d)\)\s*([2-9]\d\d)-(\d{4})/',   *;
*               Phone);                                        *;
*  7) Highlight and run the first DATA step and PROC PRINT     *;
*    step. Verify that only rows 8 and 9 have a LocParen       *;
*    value greater than 0.                                     *;
*  8) Add a subsetting IF statement to subset the rows where a *;
*    pattern was matched. Highlight and run the first DATA     *;
*    step and PROC PRINT step. Verify that only five rows are  *;
*    in the results.                                           *;
*      if Loc ne 0 or LocStartEnd ne 0 or LocParen ne 0;       *;
```

```
*  9) In the last DATA step, notice the CALL PRXDEBUG routine. *;
*     Run the DATA step and view the SAS log. Notice the       *;
*     Compiling line after each iteration.                     *;
*      call prxdebug(1);                                       *;
* 10) Add the O option to the end of the Perl regular          *;
*     expression. Run the DATA step and view the SAS log.      *;
*     Notice that the Compiling line is now only after the     *;
*     first iteration.                                         *;
*      Exp='/([2-9]\d\d)-([2-9]\d\d)-(\d{4})/o';               *;
*************************************************************;

data work.ValidPhoneNumbers;
   set pg3.phonenumbers_us;
   Loc=prxmatch('/([2-9]\d\d)-([2-9]\d\d)-(\d{4}) /',Phone);
run;

title 'Pattern Matching Phone Numbers';
proc print data=work.ValidPhoneNumbers;
run;
title;
```

Pattern Matching Phone Numbers

Obs	Phone	Loc
1	123-456-7890	0
2	234-567-8901	1
3	1-345-678-9012	3
4	567-8901	0
5	789-012-3456	0
6	890-123-456	0
7	901-234-5678 US	1
8	(345)678-9012	0
9	(456) 789-0123	0

```
data work.ValidPhoneNumbers;
   set pg3.phonenumbers_us;
   Loc=prxmatch('/([2-9]\d\d)-([2-9]\d\d)-(\d{4}) /',Phone);
   LocStartEnd=prxmatch('/^([2-9]\d\d)-([2-9]\d\d)-(\d{4})$/',strip(Phone));
run;

title 'Pattern Matching Phone Numbers';
proc print data=work.ValidPhoneNumbers;
run;
title;
```

Pattern Matching Phone Numbers

Obs	Phone	Loc	Loc StartEnd
1	123-456-7890	0	0
2	234-567-8901	1	1
3	1-345-678-9012	3	0
4	567-8901	0	0
5	789-012-3456	0	0
6	890-123-456	0	0
7	901-234-5678 US	1	0
8	(345)678-9012	0	0
9	(456) 789-0123	0	0

```
data work.ValidPhoneNumbers;
   set pg3.phonenumbers_us;
   Loc=prxmatch('/([2-9]\d\d)-([2-9]\d\d)-(\d{4}) /',Phone);
   LocStartEnd=prxmatch('/^([2-9]\d\d)-([2-9]\d\d)-(\d{4})$/',strip(Phone));
   LocParen=prxmatch('/\(([2-9]\d\d)\)\s*([2-9]\d\d)-(\d{4})$/',strip(Phone));
run;

title 'Pattern Matching Phone Numbers';
proc print data=work.ValidPhoneNumbers;
run;
```

title;

Pattern Matching Phone Numbers

Obs	Phone	Loc	LocStartEnd	LocParen
1	123-456-7890	0	0	0
2	234-567-8901	1	1	0
3	1-345-678-9012	3	0	0
4	567-8901	0	0	0
5	789-012-3456	0	0	0
6	890-123-456	0	0	0
7	901-234-5678 US	1	0	0
8	(345)678-9012	0	0	1
9	(456) 789-0123	0	0	1

```
data work.ValidPhoneNumbers;
   set pg3.phonenumbers_us;
   Loc=prxmatch('/([2-9]\d\d)-([2-9]\d\d)-(\d{4}) /',Phone);
   LocStartEnd=prxmatch('/^([2-9]\d\d)-([2-9]\d\d)-(\d{4})$/',strip(Phone));
   LocParen=prxmatch('/\(([2-9]\d\d)\)\s*([2-9]\d\d)-(\d{4})$/',strip(Phone));
   if Loc ne 0 or LocStartEnd ne 0 or LocParen ne 0;
run;

title 'Pattern Matching Phone Numbers';
proc print data=work.ValidPhoneNumbers;
run;
title;
```

Pattern Matching Phone Numbers

Obs	Phone	Loc	LocStartEnd	LocParen
1	234-567-8901	1	1	0
2	1-345-678-9012	3	0	0
3	901-234-5678 US	1	0	0
4	(345)678-9012	0	0	1
5	(456) 789-0123	0	0	1

```
data work.ValidPhoneNumbers;
   set pg3.phonenumbers_us;
```

```
   putlog 'Iteration: ' _N_=;
   call prxdebug(1); /* Sends debugging output to the SAS log. */
   Exp='/([2-9]\d\d)-([2-9]\d\d)-(\d{4})/o';
   Loc=prxmatch(Exp,Phone);
run;
```

```
Iteration: _N_=1
Compiling REx `([2-9]\d\d)-([2-9]\d\d)-(\d{4})'
size 42 first at 3
rarest char - at 0
   1: OPEN1(3)
   3:   ANYOF[2-9](12)
  12:   DIGIT(13)
  13:   DIGIT(14)
  14: CLOSE1(16)
  16: EXACT <->(18)
  18: OPEN2(20)
  20:   ANYOF[2-9](29)
  29:   DIGIT(30)
  30:   DIGIT(31)
  31: CLOSE2(33)
  33: EXACT <->(35)
  35: OPEN3(37)
  37:   CURLY {4,4}(40)
  39:     DIGIT(0)
  40: CLOSE3(42)
  42: END(0)
anchored `-' at 3 (checking anchored) stclass `ANYOF[2-9]' minlen 12

Guessing start of match, REx `([2-9]\d\d)-([2-9]\d\d)-(\d{4})' against `123-456-7890   '...
Found anchored substr `-' at offset 3...
This position contradicts STCLASS...
Looking for anchored substr starting at offset 4...
Did not find anchored substr `-'...
Match rejected by optimizer
```

Total rows: 9 Total columns: 3 Rows 1-9

	Phone	Exp	Loc
1	123-456-7890	/([2-9]\d\d)-([2-9]\d\d)-(\d{4})/o	0
2	234-567-8901	/([2-9]\d\d)-([2-9]\d\d)-(\d{4})/o	1
3	1-345-678-9012	/([2-9]\d\d)-([2-9]\d\d)-(\d{4})/o	3
4	567-8901	/([2-9]\d\d)-([2-9]\d\d)-(\d{4})/o	0
5	789-012-3456	/([2-9]\d\d)-([2-9]\d\d)-(\d{4})/o	0
6	890-123-456	/([2-9]\d\d)-([2-9]\d\d)-(\d{4})/o	0
7	901-234-5678 US	/([2-9]\d\d)-([2-9]\d\d)-(\d{4})/o	1
8	(345)678-9012	/([2-9]\d\d)-([2-9]\d\d)-(\d{4})/o	0
9	(456) 789-0123	/([2-9]\d\d)-([2-9]\d\d)-(\d{4})/o	0

```
**********************************************************;
*  Activity 2.06                          *;
```

```
*  1) Run the PROC PRINT step and view the results.      *;
*  2) Modify the WHERE statement in the PROC PRINT step   *;
*     to find all the values of Narrative that contain    *;
*     EF3, EF-3, EF4, or EF-4.                            *;
*  3) Run the PROC PRINT step. View the results and the   *;
*     SAS log. How many rows were read based on the       *;
*     WHERE statement?                                    *;
*************************************************************;

title 'Category EF3 and EF4 Tornados';
proc print data=pg3.tornado_2017narrative;
   where prxmatch('/ /',Narrative)>0; /* Returns 21 rows */
run;
title;
```

```
NOTE: There were 500 observations read from the data set
PG3.TORNADO_2017NARRATIVE.
       WHERE PRXMATCH('/ /', Narrative)>0;
```

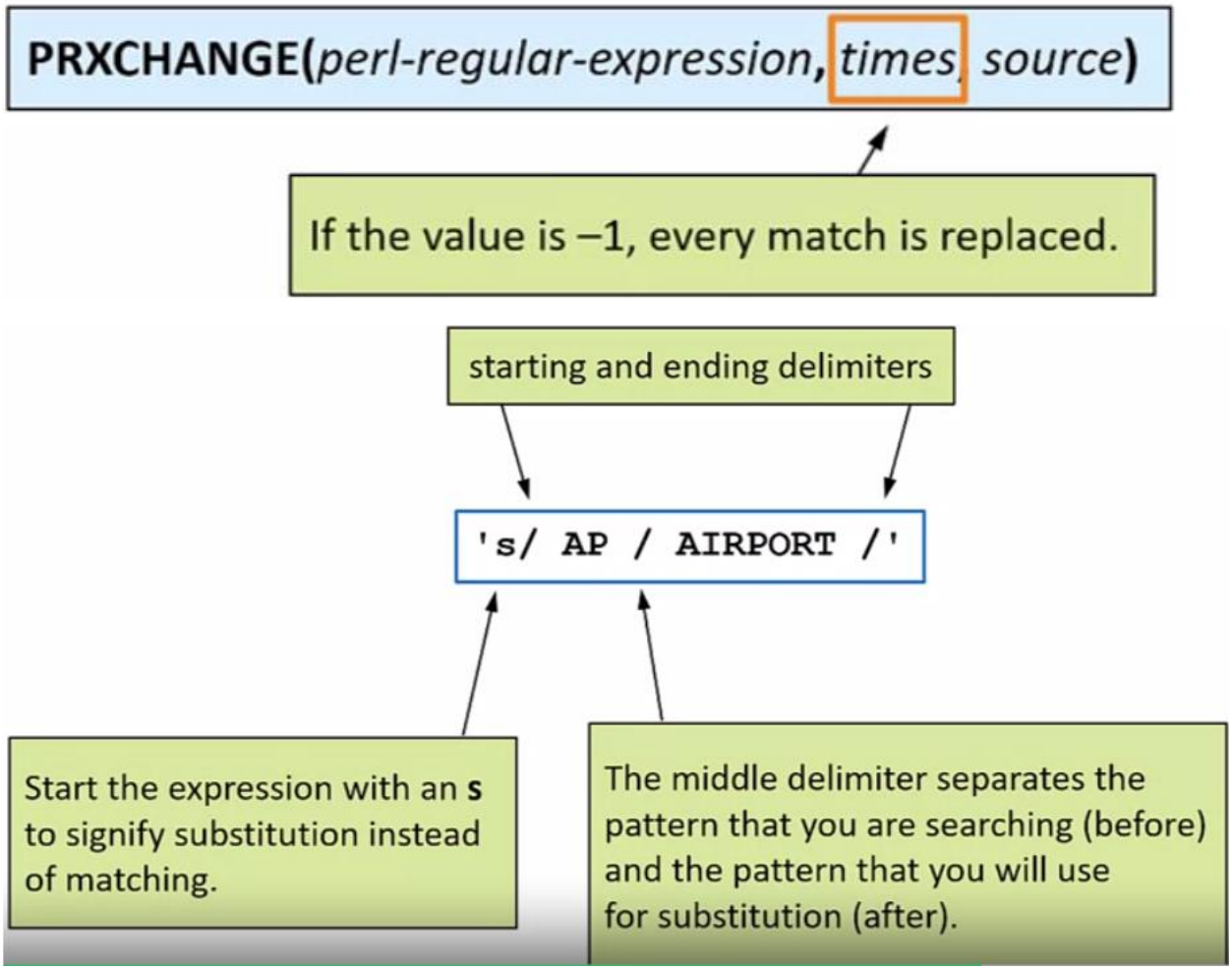
```
title 'Category EF3 and EF4 Tornados';
proc print data=pg3.tornado_2017narrative;
   where prxmatch('/(EF3|EF-3|EF4|EF-4)/',Narrative)>0; /* Returns 21 rows */
run;
title;
```

```
NOTE: There were 21 observations read from the data set
PG3.TORNADO_2017NARRATIVE.
       WHERE PRXMATCH('/(EF3|EF-3|EF4|EF-4)/', Narrative)>0;
```

```
where prxmatch('/(EF3|EF-3|EF4|EF-4)/',Narrative)>0;
where prxmatch('/(EF-?3|EF-?4)/',Narrative)>0;
where prxmatch('/EF-?(3|4)/',Narrative)>0;
where prxmatch('/EF-?[34]/',Narrative)>0;
```

Name
JUNEAU INTL. AP
KETCHIKAN INTL AP
FAIRBANKS INT AP
ANCHORAGE intl AP

Name_New
JUNEAU INTERNATIONAL AIRPORT
KETCHIKAN INTERNATIONAL AIRPORT
FAIRBANKS INTERNATIONAL AIRPORT
ANCHORAGE INTERNATIONAL AIRPORT



Name
JUNEAU INTL. AP
KETCHIKAN INTL AP
FAIRBANKS INT AP
ANCHORAGE intl AP

```
's/ INT( |L |L. )/ INTERNATIONAL /i'
```

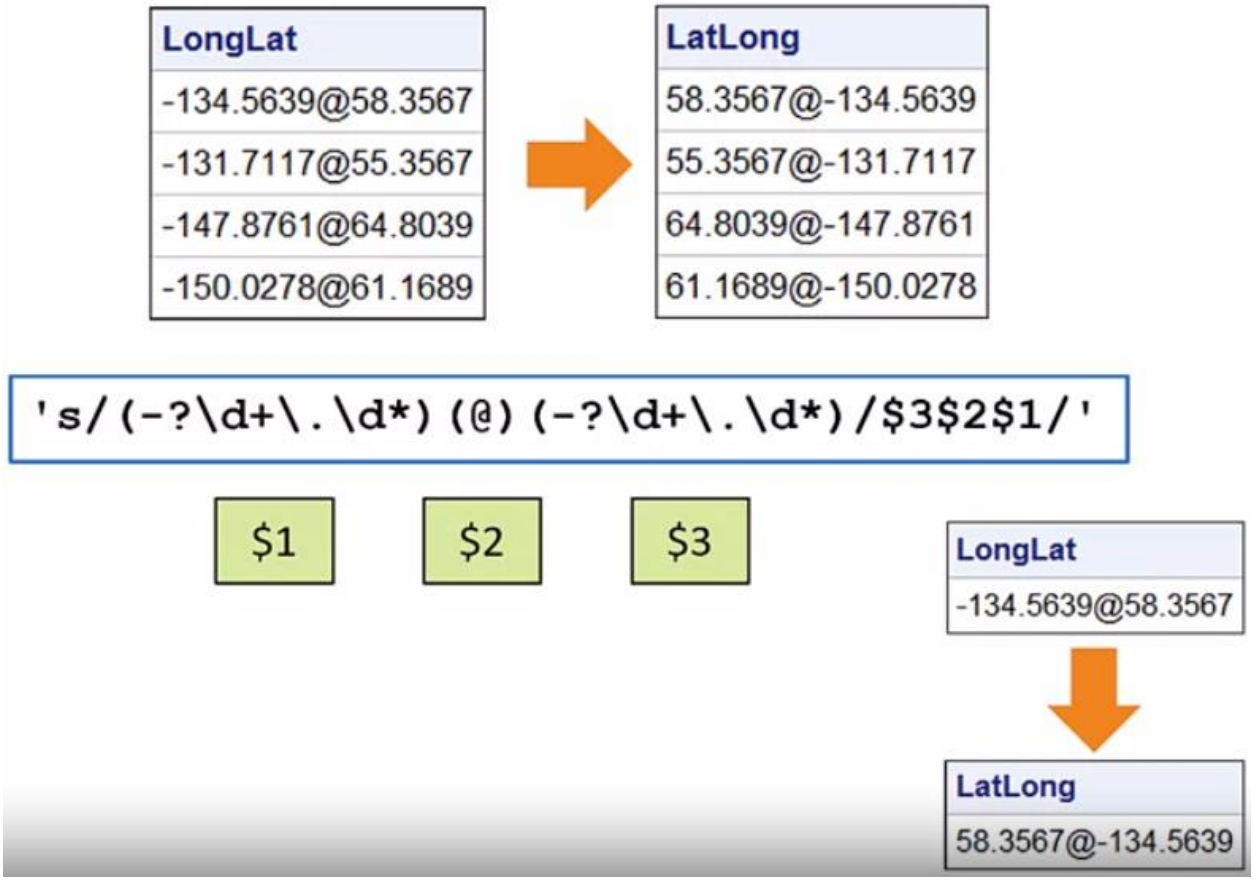
Ignore case for the pattern being searched.

LongLat
-134.5639@58.3567
-131.7117@55.3567
-147.8761@64.8039
-150.0278@61.1689

LatLong
58.3567@-134.5639
55.3567@-131.7117
64.8039@-147.8761
61.1689@-150.0278

longitude values: -180 to 180

latitude values: -90 to 90




```
*************************************************************;
*  Standardizing Data with the PRXCHANGE Function           *;
*************************************************************;

data work.AKstations;
   set pg3.weather_usstationshourly;
   where State='AK';
   Name_New=prxchange('s/ AP/ AIRPORT/',-1,Name);
   Name_New=prxchange('s/ INT( |L |L. )/ INTERNATIONAL /i',-1,Name_New);
   LatLong=prxchange('s/(-?\d+\.\d*)(@)(-?\d+\.\d*)/$3$2$1/',-1,LongLat);
run;

proc print data=work.AKstations;
   where prxmatch('/(JUNEAU|KETCHIKAN|FAIRBANKS|ANCHORAGE i)/',Name);
```

```
run;

************************************************************;
*  Demo                                    *;
*  1) In the DATA step, notice the incomplete syntax for the   *;
*    first assignment statement for Name_New and the complete *;
*    syntax for the second assignment statement for Name_New. *;
*      Name_New=prxchange('s/ / /',-1,Name);              *;
*      Name_New=prxchange                          *;
*            ('s/ INT( |L |L. )/ INTERNATIONAL /i',       *;
*             -1,Name_New);                        *;
*  2) In the first assignment statement for Name_New, modify   *;
*    the Perl regular expression to replace the letters AP    *;
*    with the word AIRPORT for all occurrences.            *;
*      Name_New=prxchange('s/ AP / AIRPORT /',-1,Name);      *;
*    Alternatively, you could use \b (word boundary) in place *;
*    of the leading and trailing spaces around the string AP. *;
*      Name_New=prxchange('s/\bAP\b/AIRPORT/',-1,Name);      *;
*  3) Run the DATA step and the PROC step. View the results    *;
*    and verify that the Name_New column contains the        *;
*    standardized values of AIRPORT and INTERNATIONAL.       *;
*  4) Uncomment the LatLong assignment statement in the DATA   *;
*    step and the VAR statement in the PROC PRINT step.      *;
*    Modify the expression in the assignment statement to     *;
*    specify the substitution of the third capture buffer     *;
*    followed by the second and first buffers.              *;
*      LatLong=prxchange                           *;
*            ('s/(-*\d+\.\d*)(@)(-*\d+\.\d*)/$3$2$1/',      *;
*             -1,LongLat);                         *;
```

```
*  5) Run the DATA step and the PROC step. View the results    *;
*     and verify that the latitude value now appears before    *;
*     the longitude value in the LatLong column.               *;
*************************************************************;

data work.weather_stations;
   set pg3.weather_usstationshourly;
   Name_New=prxchange('s/ AP / AIRPORT /',-1,Name);
   Name_New=prxchange('s/ INT( |L |L. )/ INTERNATIONAL /i',
                -1,Name_New);
   *LatLong=prxchange('s/(-*\d+\.\d*)(@)(-*\d+\.\d*)/ /',
               -1,LongLat);
run;

title 'US Weather Stations with Hourly Readings';
proc print data=work.weather_stations;
   *var Code State Name Name_New LongLat LatLong;
run;
title;
```

US Weather Stations with Hourly Readings

Obs	Code	LongLat	State	Name	Name_New
1	USW00025308	-131.5786@55.0389	AK	ANNETTE ISLAND AP	ANNETTE ISLAND AIRPORT
2	USW00025309	-134.5639@58.3567	AK	JUNEAU INTL. AP	JUNEAU INTERNATIONAL AIRPORT
3	USW00025325	-131.7117@55.3567	AK	KETCHIKAN INTL AP	KETCHIKAN INTERNATIONAL AIRPORT
4	USW00025333	-135.3647@57.0481	AK	AP SITKA AIRPORT	AP SITKA AIRPORT
5	USW00025339	-139.6711@59.5119	AK	YAKUTAT STATE AP	YAKUTAT STATE AIRPORT
6	USW00025501	-152.4856@57.7511	AK	KODIAK AP	KODIAK AIRPORT
7	USW00025503	-156.6294@58.6794	AK	KING SALMON	KING SALMON
8	USW00025507	-151.4908@59.6419	AK	HOMER AP	HOMER AIRPORT
9	USW00025624	-162.7325@55.2208	AK	COLD BAY AP	COLD BAY AIRPORT
10	USW00025713	-170.2222@57.1553	AK	ST PAUL ISLAND AP	ST PAUL ISLAND AIRPORT
11	USW00026401	-149.8@61.25	AK	ELMENDORF AFB	ELMENDORF AFB
12	USW00026409	-149.855@61.2169	AK	ANCHORAGE MERRILL FLD	ANCHORAGE MERRILL FLD
13	USW00026410	-145.4511@60.4889	AK	CORDOVA M K SMITH AP	CORDOVA M K SMITH AIRPORT
14	USW00026411	-147.8761@64.8039	AK	FAIRBANKS INT AP	FAIRBANKS INTERNATIONAL AIRPORT
15	USW00026412	-141.9292@62.9614	AK	NORTHWAY AP	NORTHWAY AIRPORT
16	USW00026415	-145.7214@63.9944	AK	BIG DELTA AP	BIG DELTA AIRPORT
17	USW00026425	-145.4589@62.1592	AK	GULKANA AP	GULKANA AIRPORT
18	USW00026442	-146.3517@61.1303	AK	VALDEZ WSO	VALDEZ WSO
19	USW00026451	-150.0278@61.1689	AK	ANCHORAGE intl AP	ANCHORAGE INTERNATIONAL AIRPORT
20	USW00026501	-156.9344@64.7367	AK	GALENA AP	GALENA AIRPORT
21	USW00026510	-155.6103@62.9575	AK	MCGRATH AP	MCGRATH AIRPORT
22	USW00026523	-151.2392@60.5797	AK	KENAI MUNI AP	KENAI MUNI AIRPORT
23	USW00026528	-150.095@62.32	AK	TALKEETNA AP	TALKEETNA AIRPORT
24	USW00026533	-151.5089@66.9161	AK	BETTLES AP	BETTLES AIRPORT
25	USW00026615	-161.8292@60.785	AK	BETHEL AP	BETHEL AIRPORT

```
data work.weather_stations;
   set pg3.weather_usstationshourly;
   Name_New=prxchange('s/ AP / AIRPORT /',-1,Name);
   Name_New=prxchange('s/ INT( |L |L. )/ INTERNATIONAL /i',
                -1,Name_New);
   LatLong=prxchange('s/(-*\d+\.\d*)(@)(-*\d+\.\d*)/$3$2$1 /',
                -1,LongLat);
run;

title 'US Weather Stations with Hourly Readings';
proc print data=work.weather_stations;
   var Code State Name Name_New LongLat LatLong;
run;
title;
```

Obs	Code	LongLat	State	Name	Name_New	LatLong
2	USW00025309	-134.5639@58.3567	AK	JUNEAU INTL. AP	JUNEAU INTERNATIONAL AIRPORT	58.3567@-134.5639
3	USW00025325	-131.7117@55.3567	AK	KETCHIKAN INTL AP	KETCHIKAN INTERNATIONAL AIRPORT	55.3567@-131.7117
14	USW00026411	-147.8761@64.8039	AK	FAIRBANKS INT AP	FAIRBANKS INTERNATIONAL AIRPORT	64.8039@-147.8761
19	USW00026451	-150.0278@61.1689	AK	ANCHORAGE intl AP	ANCHORAGE INTERNATIONAL AIRPORT	61.1689@-150.0278

US Weather Stations with Hourly Readings

Obs	Code	State	Name	Name_New	LongLat	LatLong
1	USW00025308	AK	ANNETTE ISLAND AP	ANNETTE ISLAND AIRPORT	-131.5786@55.0389	55.0389@-131.5786
2	USW00025309	AK	JUNEAU INTL. AP	JUNEAU INTERNATIONAL AIRPORT	-134.5639@58.3567	58.3567@-134.5639
3	USW00025325	AK	KETCHIKAN INTL AP	KETCHIKAN INTERNATIONAL AIRPORT	-131.7117@55.3567	55.3567@-131.7117
4	USW00025333	AK	AP SITKA AIRPORT	AP SITKA AIRPORT	-135.3647@57.0481	57.0481@-135.3647
5	USW00025339	AK	YAKUTAT STATE AP	YAKUTAT STATE AIRPORT	-139.6711@59.5119	59.5119@-139.6711
6	USW00025501	AK	KODIAK AP	KODIAK AIRPORT	-152.4856@57.7511	57.7511@-152.4856
7	USW00025503	AK	KING SALMON	KING SALMON	-156.6294@58.6794	58.6794@-156.6294
8	USW00025507	AK	HOMER AP	HOMER AIRPORT	-151.4908@59.6419	59.6419@-151.4908

```
**********************************************************;
*  Activity 2.07                                         *;
*  1) Run the program and view the results. Notice that  *;
*     the Loc column contains the first position where   *;
*     EF- is found in the Narrative column.              *;
*  2) Uncomment the Narrative_New assignment statement.  *;
*  3) Modify the first argument of the PRXCHANGE function *;
*     to find the pattern of EF- and substitute it with  *;
*     the value of EF.                                   *;
*  4) Modify the second argument of the PRXCHANGE        *;
*     function so that all occurrences of the pattern    *;
*     are substituted.                                   *;
*  5) Run the program and verify that the Narrative_New  *;
*     column no longer contains the string EF- for       *;
*     every Loc value greater than 0.                    *;
*  6) For row 7, how many EF- values were substituted    *;
*     by EF?                                             *;
**********************************************************;
/*Activity 2.07
If necessary, start SAS Studio before you begin. If you restarted your SAS session,
```

submit your libname.sas program to access the practice data.

Open p302a07.sas from the activities folder and perform the following tasks:

Run the program and view the results.

What does the value in the Loc column represent?

Modify the program as follows:

Uncomment the Narrative_New assignment statement.

Modify the first argument of the PRXCHANGE function to find the pattern of EF- and change it to the value EF.

Modify the second argument of the PRXCHANGE function so that all occurrences of the pattern are substituted.

Run the program.

For row 7, how many EF- values were changed to EF. Note: Type a numeric value.

*/

```
data work.tornadoEF;
   set pg3.tornado_2017narrative;
   length Narrative_New $ 4242;
   Loc=prxmatch('/EF-/',Narrative);
   Narrative_New=prxchange('s/EF-/EF/',-1,Narrative);
run;

title 'US Tornados';
proc print data=work.tornadoEF;
run;
title;
```

US Tornados

Obs	State	County	BeginDate	BeginTime	Narrative	Narrative_New	Loc
1	GA	COOK CO.	22JAN2017	335	This is a continuation of the northeast Brooks county tornado. The tornado then continued into Cook County. Still at EF-3 strength, it swept about 35 manufactured homes into a pile of rubble at the far end of the Sunshine Acres mobile home park. Seven people lost their lives. The tornado then went on to destroy about two thirds of a brick home on Val Del Road, collapsing in two walls and removing most of the second story. Another home built of concrete blocks was destroyed. A nearby farm had several concrete anchors for a large metal structure pulled from the ground. Max winds were estimated near 140 mph. Damage cost was estimated.	This is a continuation of the northeast Brooks county tornado. The tornado then continued into Cook County. Still at EF3 strength, it swept about 35 manufactured homes into a pile of rubble at the far end of the Sunshine Acres mobile home park. Seven people lost their lives. The tornado then went on to destroy about two thirds of a brick home on Val Del Road, collapsing in two walls and removing most of the second story. Another home built of concrete blocks was destroyed. A nearby farm had several concrete anchors for a large metal structure pulled from the ground. Max winds were estimated near 140 mph. Damage cost was estimated.	119

**********;

```
*  LESSON 2, PRACTICE 4                         *;
*************************************************************;
```

/*Practice Level 1: Using the PRXMATCH and PRXCHANGE Functions

If necessary, start SAS Studio before you begin. If you restarted your SAS session,

submit your libname.sas program to access the practice data.

The pg3.np_acres table contains acreage amounts for national parks. The ParkName column contains the descriptive name

for each park. Find the national preserves by locating all rows with a ParkName value that contains the string

N PRES, N PRESERVE, NPRES, or NPRE followed by a space. Within the ParkName values,

modify the national preserve string to be displayed with the string of NPRES.

Open the p302p04.sas program in the practices folder.

Run the program to view the ParkName values.

What is the value of ParkName in row 12?

Return to the code tab to modify the program.

In an assignment statement, use the PRXMATCH function to create the column Position,

which is equal to the starting position of a string that represents national preserves.

All national preserves will contain one of the following strings followed by a space (\s):

N PRES, N PRESERVE, NPRES, or NPRE.

Add a subsetting IF statement to include only the rows where the Position values are greater than zero.

Run the program and verify the results.

How many rows are returned for national preserves? Note: Type a numeric value.

Add an assignment statement that calls the PRXCHANGE function to change the strings N PRES, N PRESERVE, or NPRES to be NPRE.

Store the changed values in a column named NewName.

Run the program and verify that NPRE is displayed in the NewName column for all rows in the table.

How many national preserves are in the state of Alaska (AK)? Note: Type a numeric value.

```
*/
data work.NationalPreserves;
   set pg3.np_acres;
run;

title 'National Preserves (NPRE)';
proc print data=work.NationalPreserves;
run;
title;
```

National Preserves (NPRE)

Obs	Region	ParkCode	ParkName	State	GrossAcres
1	Southeast	ABLI	A LINCOLN BIRTHPL NHP	KY	344.50
2	Northeast	ACAD	ACADIA NP	ME	49,057.36
3	Northeast	ADAM	ADAMS NHP	MA	23.82
4	Northeast	AFBG	AFRICAN BURIAL GROUND NM	NY	0.35
5	Midwest	AGFO	AGATE FOSSIL BEDS NM	NE	3,057.87
6	Alaska		ALAGNAK WILD RVR	AK	30,664.79

```
data work.NationalPreserves;
   set pg3.np_acres;
   Position=prxmatch('/\s(N PRES|N PRESERVE|NPRES|NPRE)/',ParkName);
   if Position ne 0;
run;

title 'National Preserves (NPRE)';
proc print data=work.NationalPreserves;
run;
title;
```

National Preserves (NPRE)

Obs	Region	ParkCode	ParkName	State	GrossAcres	Position
1	Alaska		ANIAKCHAK N PRESERVE	AK	464,117.93	10
2	Alaska	BELA	BERING LAND BRIDGE N PRES	AK	2,697,391.01	19
3	Southeast	BICY	BIG CYPRESS N PRESERVE	FL	574,453.32	12
4	Intermountain	BITH	BIG THICKET N PRESERVE	TX	112,500.81	12
5	Pacific West		CRATERS OF THE MOON NPRES	ID	410,732.92	20
6	Alaska	NOAT	NOATAK N PRESERVE	AK	6,587,071.39	7
7	Midwest	TAPR	TALLGRASS PRAIRIE NPRE	KS	10,882.67	18
8	Intermountain	VALL	VALLES CALDERA N PRES	NM	89,766.09	15
9	Alaska		WRANGELL-ST ELIAS N PRES	AK	4,852,644.52	18

```
data work.NationalPreserves;
   set pg3.np_acres;
   Position=prxmatch('/\s(N PRES|N PRESERVE|NPRES|NPRE)/',ParkName);
   if Position ne 0;
   NewName=prxchange('s(N PRES\s|N PRESERVE\s|NPRES\s|NPRE\s)/NPRE/',1,ParkName);
run;

title 'National Preserves (NPRE)';
proc print data=work.NationalPreserves;
run;
title;
```

National Preserves (NPRE)

Obs	Region	ParkCode	ParkName	State	GrossAcres	Position	NewName
1	Alaska		ANIAKCHAK N PRESERVE	AK	464,117.93	10	ANIAKCHAK NPRE
2	Alaska	BELA	BERING LAND BRIDGE N PRES	AK	2,697,391.01	19	BERING LAND BRIDGE NPRE
3	Southeast	BICY	BIG CYPRESS N PRESERVE	FL	574,453.32	12	BIG CYPRESS NPRE
4	Intermountain	BITH	BIG THICKET N PRESERVE	TX	112,500.81	12	BIG THICKET NPRE
5	Pacific West		CRATERS OF THE MOON NPRES	ID	410,732.92	20	CRATERS OF THE MOON NPRE
6	Alaska	NOAT	NOATAK N PRESERVE	AK	6,587,071.39	7	NOATAK NPRE
7	Midwest	TAPR	TALLGRASS PRAIRIE NPRE	KS	10,882.67	18	TALLGRASS PRAIRIE NPRE
8	Intermountain	VALL	VALLES CALDERA N PRES	NM	89,766.09	15	VALLES CALDERA NPRE
9	Alaska		WRANGELL-ST ELIAS N PRES	AK	4,852,644.52	18	WRANGELL-ST ELIAS NPRE

```
*Solution;
/* p302p04_s.sas */
```

```
data work.NationalPreserves;
   set pg3.np_acres;
   Position=prxmatch('/N PRES\s|N PRESERVE\s|NPRES\s|NPRE\s/',ParkName);
   if Position ne 0;
   NewName=prxchange('s/N PRES\s|N PRESERVE\s|NPRES\s/NPRE /',1,ParkName);
run;

title 'National Preserves (NPRE)';
proc print data=work.NationalPreserves;
run;
title;
************************************************************;
*  LESSON 2, PRACTICE 5                                    *;
************************************************************;
/*Practice Level 2: Using the PRXCHANGE Function with Capture Buffers
If necessary, start SAS Studio before you begin. If you restarted your SAS session,
submit your libname.sas program to access the practice data.
```

The sashelp.baseball data set contains salary and performance information for Major League Baseball players

(excluding pitchers) who played at least one game in both the 1986 and 1987 seasons.

The Name column contains the player's name in the form LastName, FirstName (that is, Mattingly, Don).

For each player, rearrange the order of the player's name to be in the form FirstName LastName (that is, Don Mattingly).

Open the p302p05.sas program in the practices folder. Run the program.

What are the values of Name for rows 41 and 236?

1) In an assignment statement, create a column named FirstLastName based on using the PRXCHANGE function to rearrange

the order of the Name column. Use three sets of parentheses to create three capture buffers that represent the pattern

of the Name column:

The first set of parentheses represents the last name. The last name can contain an embedded blank (for example, Van Slyke)

or a special character (for example, O'Brien).

The second set of parentheses represents the comma and space.

The third set of parentheses represents the first name. The first name can contain an embedded blank (that is, Billy Jo).

Include a word boundary metacharacter at the end of the pattern to avoid trailing spaces on the first name.

2) Use capture buffer references to rearrange the order of the capture buffers so that the player's name is

in the form of FirstName LastName.

3) Run the program and view the results.

What is the value of FirstLastName for row 41?

What is the value of FirstLastName for row 236?

What is the byte size of the FirstLastName column?

*/

```
data work.BaseballPlayers;
   set sashelp.baseball(keep=Name);
   FirstLastName=prxchange('s/(\D+)(,\s)(\D*)/$3 $1/',-1,Name);
run;

title 'Names of Baseball Players';
proc print data=work.BaseballPlayers;
run;
title;
```

Names of Baseball Players

Obs	Name	FirstLastName
1	Allanson, Andy	Andy Allanson
2	Ashby, Alan	Alan Ashby
3	Davis, Alan	Alan Davis
4	Dawson, Andre	Andre Dawson
5	Galarraga, Andres	Andres Galarraga
6	Griffin, Alfredo	Alfredo Griffin
7	Newman, Al	Al Newman
8	Salazar, Argenis	Argenis Salazar
9	Thomas, Andres	Andres Thomas
10	Thornton, Andre	Andre Thornton
11	Trammell, Alan	Alan Trammell
12	Trevino, Alex	Alex Trevino
13	Van Slyke, Andy	Andy Van Slyke
14	Wiggins, Alan	Alan Wiggins

*Solution;

/* p302p05_s.sas */

```
data work.BaseballPlayers;
   set sashelp.baseball(keep=Name);
   FirstLastName=prxchange('s/(\w+\D*\w*)(, )(\w+\s*\w*\b)/$3 $1/',-1,Name);
run;

title 'Names of Baseball Players';
proc print data=work.BaseballPlayers;
run;
title;
```