Applications - Examples

This section introduces some array applications in

- Data manipulations from data search - Example 1
- Count consecutive days - Example 2
- LOCF - Example 3
- Find and replace - Example 4
- Shift - Example 5

Leading to a more complicated efficient process.

Example 1 - Search Specified Value

- The example is to find on which day the maximum efficacy is reached.
- The algorithm is to compare the target value against an array and perform an action if the target value is found in the array.

SUBJECT	DAY1	DAY2	DAY3	DAY4	TMAX
101	0.0	0.0	0.0	0.0	.
102	.	0.5	0.5	0.0	2
106	0.5	0.0	0.0	0.0	1
107	0.5	2.0	0.5	2.0	2
111	1.0	3.0	2.0	2.5	2
112	2.0	3.0	2.5	3.5	4

```
data pd2;
  set pd1;
  array days[4] day1-day4;
  maxscore=max (of days [*]);
  do i=1 to dim(days);
      if maxscore >0 and
             days[i]=maxscore then do;
          tmax=i;
          return;
      end;
  end;
  drop i maxscore;
run;
```

Example 2 - Count Consecutive Days (1)

SUBJID	DATE	DATECNT
1	25MAR2004	1
1	26MAR2004	2
1	27MAR2004	3
1	28MAR2004	4
1	29MAR2004	5
2	26MAR2004	1
2	27MAR2004	2
2	29MAR2004	3
3	26MAR2004	1
3	27MAR2004	2
3	28MAR2004	3
3	02APR2004	4
4	02APR2004	1

- We check to see whether a subject has experienced night awakening for more than 3 consecutive days.
- From the DIARY data set and program below, we can easily list the subjects and their consecutive days along with start date and stop date by using array.

Target dataset

SUBJID	count	f_date	l_date
1	5	25MAR2004	29MAR2004
3	3	26MAR2004	28MAR2004

Example 2 - Count Consecutive Days (2)

Step1. Transpose

Temp1

SUBJID	_NAME_	_dat1	_dat2	_dat3	_dat4	_dat5
1	date	25MAR2004	26MAR2004	27MAR2004	28MAR2004	29MAR2004
2	date	26MAR2004	27MAR2004	29MAR2004	.	.
3	date	26MAR2004	27MAR2004	28MAR2004	02APR2004	.
4	date	02APR2004	.	.	.	.

```
proc transpose data=diary prefix=_dat out=temp1;
    by subjid;
    var date;
run;
```

Example 2 - Count Consecutive Days (3)

Step2. Count Consecutive Days using Array

```
data temp2
  (keep=subjid flag count I rename=(i=datecnt));
  set temp1 ;
  array dates {*} _dat: dummy ;
  retain flag count 1;
  do i=1 to dim(dates)-1;
    if dates[i]^=. then do;
      if dates[i] = dates[i+1]-1 then do;
        output;  count=count + 1;
      end;
      else do;
        output;  flag =flag + 1;  count=1;
      end;
    end;
  end;
run;
```

Temp2

SUBJID	flag	count	datecnt
1	1	1	1
1	1	2	2
1	1	3	3
1	1	4	4
1	1	5	5
2	2	1	1
2	2	2	2
2	3	1	3
3	4	1	1
3	4	2	2
3	4	3	3
3	5	1	4
4	6	1	1

Example 2 - Count Consecutive Days (4)

Step3. Manipulation

```
data temp3;
  merge temp2 diary;
  by subjid datecnt;
run;
```

Temp3

SUBJID	flag	count	datecnt	date
1	1	1	1	25MAR2004
1	1	2	2	26MAR2004
1	1	3	3	27MAR2004
1	1	4	4	28MAR2004
1	1	5	5	29MAR2004
2	2	1	1	26MAR2004
2	2	2	2	27MAR2004
2	3	1	3	29MAR2004
3	4	1	1	26MAR2004
3	4	2	2	27MAR2004
3	4	3	3	28MAR2004
3	5	1	4	02APR2004
4	6	1	1	02APR2004

Example 2 - Count Consecutive Days (5)

Step3. Manipulation

```
data continue (where=(count >=3 ));
  /*3 consecutive days defined*/
  set temp3;
    by subjid flag;
    retain f_date ;
    if first.flag then f_date=date ;
    if last.flag then do;
        l_date=date ;
        output;
    end;
    keep subjid f_date l_date count;
    format f_date l_date date9. ;
run;
```

Target dataset

SUBJID	*count*	*f_date*	*l_date*
1	5	25MAR2004	29MAR2004
3	3	26MAR2004	28MAR2004

Example 3 - Data LOCF (1)

"LOCF" stands for "Last Observation Carried Forward", it means last non-missing value carried forward.

TIME1	TIME2	TIME3	TIME4	TIME5
A	B	.	.	E

TIME1	TIME2	TIME3	TIME4	TIME5
A	B	B	B	E

In LOCF analyses, when a patient drops out of a trial, the results of the last evaluation are carried forward as if the he had continued to the completion of the trial without further change.

Since patients who discontinue medication are regarded as treatment failures, LOCF analyses are widely considered to provide a more conservative test of drug effects.

Example 3 - Data LOCF (2)

The following data set called SCORE will be used as the example.

SUBJID	TIME1	TIME2	TIME3	TIME4	TIME5	MAKEUP
1	0.5	0.5	0.0	0.0	0.5	0.5
2	0.0	0.5	.	.	1.5	0.0
3	0.0	0.0	1.0	.	0.0	0.0
4	0.0	0.0	0.0	.	0.0	0.0
5	0.0	0.5	1.5	.	0.5	0.5
6	0.0	1.0	1.5	0.5	.	1.0

```
data locf ;
  set score ;
  array time [*] time: ;
  do i=1 to dim(time);
    if time[i]=. then time[i]=time[i-1];
  end;
  drop i makeup;
run;
```

SUBJID	TIME1	TIME2	TIME3	TIME4	TIME5
1	0.5	0.5	0.0	0.0	0.5
2	0.0	0.5	0.5	0.5	1.5
3	0.0	0.0	1.0	1.0	0.0
4	0.0	0.0	0.0	0.0	0.0
5	0.0	0.5	1.5	1.5	0.5
6	0.0	1.0	1.5	0.5	0.5

Example 4 - Find and Replace (1)

The array-implemented find& replace is exceptionally powerful and fast. The algorithm replaces elements referred to by iterator i in the array with new value when the condition holds, such as to find and replace the missing data. In some cases, the experiment measurements are not conducted continuously. They are discrete instead. To test the irritation of skins to patch or ointment as the example, the skin at different positions are supposed to be tested in the order of left arm, right arm, back, ... If one or more points somehow are skipped, the makeup tests would be done to get those data missed.

TIME1	TIME2	TIME3	TIME4	TIME5	MAKEUP
A	B	.	D	E	C

Example 4 - Find and Replace (2)

SUBJID	TIME1	TIME2	TIME3	TIME4	TIME5	MAKEUP
1	0.5	0.5	0.0	0.0	0.5	0.5
2	0.0	0.5	.	.	1.5	0.0
3	0.0	0.0	1.0	.	0.0	0.0
4	0.0	0.0	0.0	.	0.0	0.0
5	0.0	0.5	1.5	.	0.5	0.5
6	0.0	1.0	1.5	0.5	.	1.0

```
data replace;
    set score;
    array apps [5] time1- time5;
    do i=1 to dim(apps);
        if apps[i] =. then apps[i]=makeup ;
    end;
    drop i ;
run;
```

SUBJID	TIME1	TIME2	TIME3	TIME4	TIME5	MAKEUP
1	0.5	0.5	0.0	0.0	0.5	0.5
2	0.0	0.5	0.0	0.0	1.5	0.0
3	0.0	0.0	1.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0	0.0	0.0
5	0.0	0.5	1.5	0.5	0.5	0.5
6	0.0	1.0	1.5	0.5	1.0	1.0

Example 5 - Data Shift (1)

One subject should undergo a certain times of tests in some situations, and the time order should be kept, then a data shift process can be applied with the help of array.

TIME1	TIME2	TIME3	TIME4	TIME5	MAKEUP
A	B	.	D	E	C

↓

TIME1	TIME2	TIME3	TIME4	TIME5
A	B	D	E	C

Example 5 - Data Shift (2)

```
data shift;
  set score;
  array apps[*] time: makeup;
  do i = 1 to dim(apps)-1;
    if apps[i] = . then do;
       do j = i to dim(apps)-1;
           apps[j] = apps[j+1];
       end;
       mu='-'||compress(i);
       if apps[i] = . then i=i-1;
    end;
  end;
  drop i j ;
run;
```

SUBJID	TIME1	TIME2	TIME3	TIME4	TIME5	MAKEUP
1	0.5	0.5	0.0	0.0	0.5	0.5
2	0.0	0.5	.	.	1.5	0.0
3	0.0	0.0	1.0	.	0.0	0.0
4	0.0	0.0	0.0	.	0.0	0.0
5	0.0	0.5	1.5	.	0.5	0.5
6	0.0	1.0	1.5	0.5	.	1.0

SUBJID	TIME1	TIME2	TIME3	TIME4	TIME5	MAKEUP	mu
1	0.5	0.5	0.0	0.0	0.5	0.5	
2	0.0	0.5	1.5	0.0	0.0	0.0	-3
3	0.0	0.0	1.0	0.0	0.0	0.0	-4
4	0.0	0.0	0.0	0.0	0.0	0.0	-4
5	0.0	0.5	1.5	0.5	0.5	0.5	-4
6	0.0	1.0	1.5	0.5	1.0	1.0	-5

A do loop would be useful if there are more than one missing data in the rows.

Example 6 - Data Merge (1)

It is often required to merge dose data with other safety data, such as adverse events, vital signs, ECG, and lab results, and locate the dose-related safety profiles. For example, a patient is given several doses at certain time points. After each dose, some adverse events may occur to the patient. We need to know which adverse event is associated with which dose. Suppose there is a dose dataset and an adverse event dataset.

AE

SUBJID	AE	AEDTTM
1	NERVOUSNESS	30JAN1999:06:00:00
1	TACHYCARDIA	30JAN1999:12:15:00
1	NAUSEA	06FEB1999:16:20:00
1	DIZZINESS	20FEB1999:09:20:00
2	HEADACHE	06FEB1999:14:10:00
2	NAUSEA	06FEB1999:17:40:00

Dose

SUBJID	DOSEN1	DOSEN2	DOSEN3	DOSEN4
1	30JAN1999:08:00:00	06FEB1999:08:00:00	20FEB1999:08:00:00	27FEB1999:08:00:00
2	30JAN1999:08:01:00	06FEB1999:08:01:00	20FEB1999:08:01:00	06MAR1999:08:01:00
3	30JAN1999:08:02:00	06FEB1999:08:02:00	13FEB1999:08:02:00	27FEB1999:08:02:00

Example 6 - Data Merge (2)

SUBJID	AEDTTM	AE	dosedttm	dosenum	hrpostds
1	30JAN99:12:15:00	TACHYCARDIA	30JAN1999:08:00:00	1	4.3
1	06FEB99:16:20:00	NAUSEA	06FEB1999:08:00:00	2	8.3
1	20FEB99:09:20:00	DIZZINESS	20FEB1999:08:00:00	3	1.3
2	06FEB99:14:10:00	HEADACHE	06FEB1999:08:01:00	2	6.2
2	06FEB99:17:40:00	NAUSEA	06FEB1999:08:01:00	2	9.7

```
data dose_ae;
    merge dose ae;
    by subjid;
    array dosen {*} dosen:;
    do i=1 to dim(dosen);
        if dosen[i]^=. and aedttm > dosen[i] then do;
            dosedttm = dosen[i];            dosenum = i;
        end;
    end;
    if dosenum^=.;
    hrpostds = round(((aedttm-dosedttm)/3600), 0.1);
    format dosedttm datetime20.;
    drop i dosen1 - dosen4;
run;
```

The final result is shown above, variable DOSENUM is the order number of doses, and HRPOSTDS is time in hours after dosing.