Paper 9180-2016

Efficiently Create Rolling 12 Month and Year to Date Rates ….with PROC MEANS and PROC EXPAND

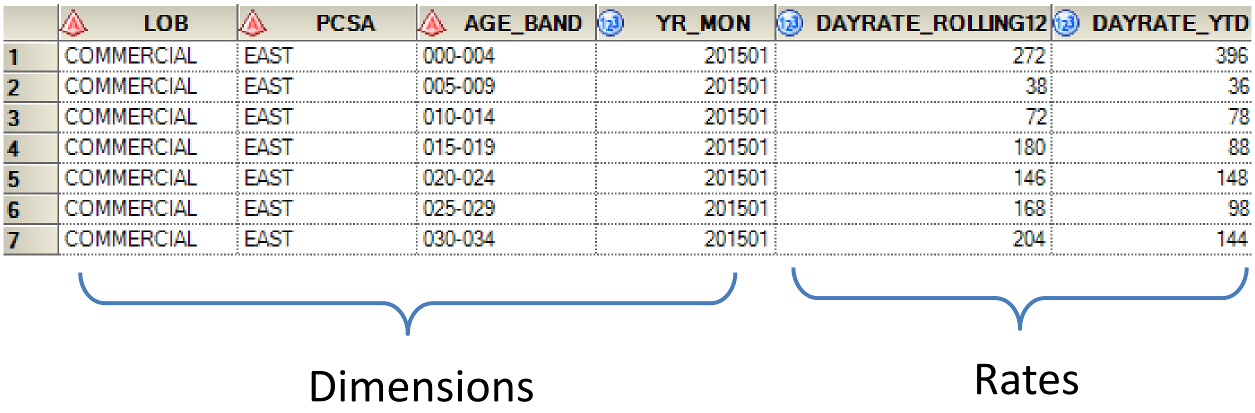
Thomas Gant, Kaiser Permanente

ABSTRACT

This session illustrates how to quickly create rates over a specified period of time using the MEANS and EXPAND procedures. If you want to know how to use the power of SAS® to create a year-to-date or rolling 12-month rate across several dimensions, this paper is for you. At Kaiser Permanente, we use this technique to develop Emergency Department (ED) use rates, ED admit rates, patient day rates, readmission rates, and more. Each of these metrics is important to understanding how the membership of a health care organization utilizes its hospital services.

INTRODUCTION

Our goal is to illustrate the power of PROC MEANS and PROC EXPAND to create a SAS dataset which gives rolling 12 month and year to date patient day rates (PDR) by every possible combination of calendar month, line of business (LOB), primary care service area (PCSA), and age band. An example of our desired output is in Output 1 below:



	LOB	PCSA	AGE_BAND	YR_MON	DAYRATE_ROLLING12	DAYRATE_YTD
1	COMMERCIAL	EAST	000-004	201501	272	396
2	COMMERCIAL	EAST	005-009	201501	38	36
3	COMMERCIAL	EAST	010-014	201501	72	78
4	COMMERCIAL	EAST	015-019	201501	180	88
5	COMMERCIAL	EAST	020-024	201501	146	148
6	COMMERCIAL	EAST	025-029	201501	168	98
7	COMMERCIAL	EAST	030-034	201501	204	144

Output 1. An extract from the final table we would like to produce. The complete table will have every possible combination of LOB * PCSA * age_band * yr_mon

Patient day rate is a measure of our membership’s utilization. It is an annualized rate of how many days per 1000 members that our membership spends in an ‘inpatient’ status in the hospital. It is calculated by the following equation:

$$\text{PDR} = \left(\frac{\text{Total } \textit{Discharge Days}}{(\text{Average } \textit{Memberships} \text{ / 1000 })} \right) \text{ X } \left(\text{365 / Time Period in Days} \right)$$

To obtain our Rolling 12 Months rates, for any given month we will need the following metrics:

1. Total Discharge Days for the past 12 months
2. Average Memberships for the past 12 months
3. Total Number of Calendar Days in the past 12 months (we assume 365 and will not calculate)

To obtain our YTD rates, for any given month we will need the following metrics:

1. Total Discharge Days YTD
2. Average Memberships YTD
3. Total Number of Calendar Days YTD

We want each of these metrics by every combination of the following dimensions:

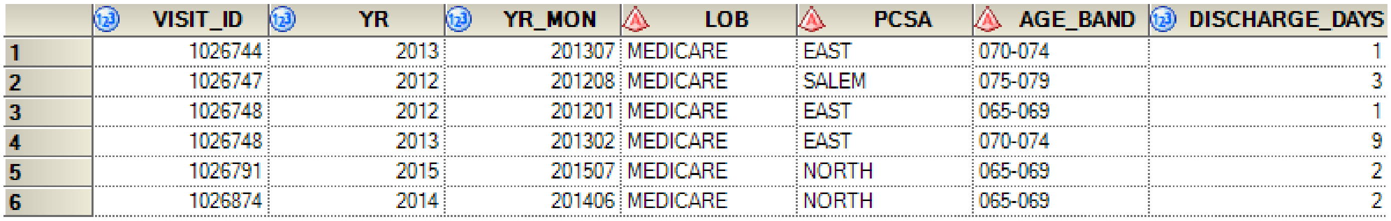
- Year-Month
- LOB
- PCSA
- Age Band

A powerful function of PROC MEANS, given a database table with several dimensions and one or more facts, is to perform a mathematical calculation on fact columns across several different combinations of dimensions. To create our patient day rates we will utilize a membership database table at the member and month level and a hospital inpatient visit table that includes discharge days as a variable. If each of these tables has the bulleted dimensions above as fields, PROC MEANS can easily determine and output the count of members and a sum of discharge days by every possible dimension combination into a SAS data set.

With PROC EXPAND, datasets, once sorted properly, can have mathematical functions performed on a column across several records. In our example we will calculate rolling 12-months and year-to-date total members and total discharge days by every possible combination of the aforementioned dimensions. The resulting membership and discharge day data sets will be joined with a MERGE statement, and we will apply our patient day rate equation to produce a patient day rate for the desired time frame and given dimensions.

DATA SETUP

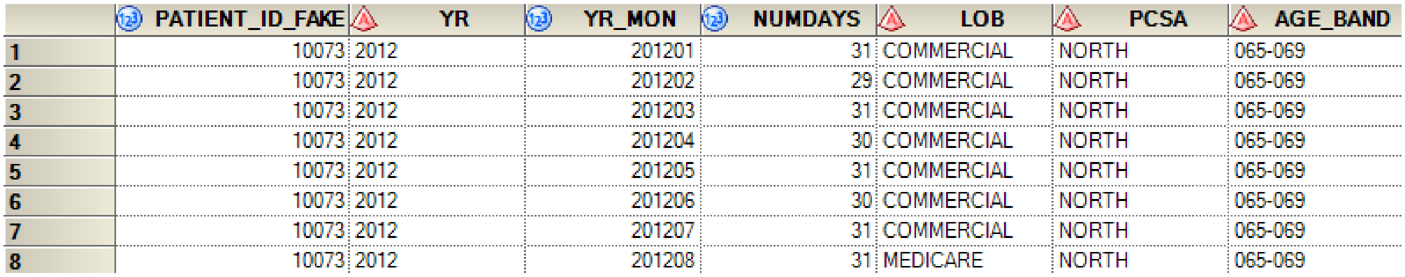
The first table that is required is a table with the 'numerator' facts, in our case 'discharge days', of our PDR equation on page 1, and all of the dimensions that the final rates will encompass. Discharge days can be thought of as the number of days a patient spends in the hospital for a particular hospital visit. Each record in Output 2 below represents a hospital visit:



	VISIT_ID	YR	YR_MON	LOB	PCSA	AGE_BAND	DISCHARGE_DAYS
1	1026744	2013	201307	MEDICARE	EAST	070-074	1
2	1026747	2012	201208	MEDICARE	SALEM	075-079	3
3	1026748	2012	201201	MEDICARE	EAST	065-069	1
4	1026748	2013	201302	MEDICARE	EAST	070-074	9
5	1026791	2015	201507	MEDICARE	NORTH	065-069	2
6	1026874	2014	201406	MEDICARE	NORTH	065-069	2

Output 2. A hospital visit level table with the number of discharge days for that visit

The second table that we require is a table with the ‘denominator’ facts of our equation on page 1, and all of the dimensions that the final rates will encompass. Each record in Output 3 below represents one membership in one given month:



	PATIENT_ID_FAKE	YR	YR_MON	NUMDAYS	LOB	PCSA	AGE_BAND
1	10073	2012	201201	31	COMMERCIAL	NORTH	065-069
2	10073	2012	201202	29	COMMERCIAL	NORTH	065-069
3	10073	2012	201203	31	COMMERCIAL	NORTH	065-069
4	10073	2012	201204	30	COMMERCIAL	NORTH	065-069
5	10073	2012	201205	31	COMMERCIAL	NORTH	065-069
6	10073	2012	201206	30	COMMERCIAL	NORTH	065-069
7	10073	2012	201207	31	COMMERCIAL	NORTH	065-069
8	10073	2012	201208	31	MEDICARE	NORTH	065-069

Output 3. A member-month level table, with each record representing one member in one month

SUMMARIZE NUMERATOR FACTS WITH PROC MEANS

PROC MEANS is a powerful procedure for summarizing data. We can use PROC MEANS to sum the discharge days, our fact for the numerator in the patient day rate equation, by the dimensions that we are interested in:

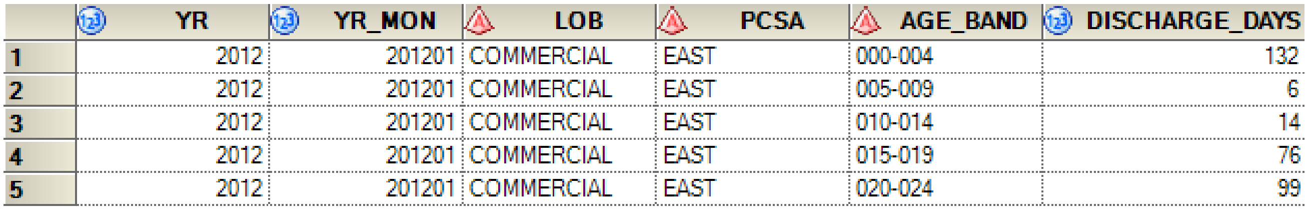
```
PROC MEANS data=encounters sum noprint;
     class yr yr_mon lob pcsa age_band;
     types yr*yr_mon*lob*pcsa*age_band;
     var discharge_days;
     output out=day_counts (drop=_FREQ_ _TYPE_) sum=;
RUN;
```

In the CLASS statement are the class variables, or dimensions, that we will use to group our facts.

The TYPES statement sets the combination of class variables that will be in each grouping. Any combination including any of the variables in the class statement can be created. By using the TYPES statement above we will group our data by every possible combination of yr_mon, lob, pcsa, and age band.

The VAR statement identifies the fact variable to be analyzed.

The ‘SUM=’ option will sum the variable in the VAR statement for our dimension combinations. Our output is the sum of ‘discharge days’ by every combination of class variables listed in the TYPES statement. Output 4 displays an extract from our results:



	YR	YR_MON	LOB	PCSA	AGE_BAND	DISCHARGE_DAYS
1	2012	201201	COMMERCIAL	EAST	000-004	132
2	2012	201201	COMMERCIAL	EAST	005-009	6
3	2012	201201	COMMERCIAL	EAST	010-014	14
4	2012	201201	COMMERCIAL	EAST	015-019	76
5	2012	201201	COMMERCIAL	EAST	020-024	99

Output 4. An extract from our PROC MEANS results, which summed discharge days by every combination of our dimensions

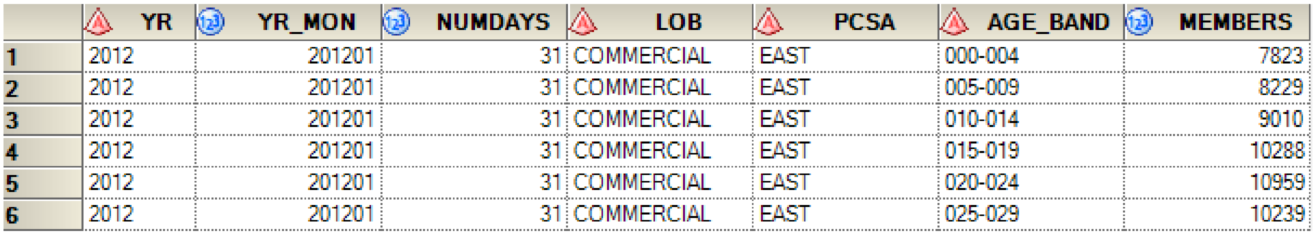
The ‘DISCHARGE_DAYS’ column gives discharge days summed by month, lob, pcsa, and age band.

SUMMARIZE DENOMINATOR FACTS WITH PROC MEANS

Now we use PROC MEANS to count memberships, the denominator in the patient day rate equation, by the dimensions that we are interested in:

```
PROC MEANS data=membership noprint;
     class yr yr_mon numdays lob pcsa age_band;
     types yr*yr_mon*numdays*lob*pcsa*age_band;
     output out=member_counts(rename=patient_ID_fake=MEMBERS
     drop=_FREQ_ _TYPE_) N=;
RUN;
```

In this PROC MEANS we are now using our membership table, where one record represents one membership in one month. We use the option “N=” to perform a count of records by every combination of yr_mon, lob, pcsa, and age band. Output 5 displays an extract from the results:



	YR	YR_MON	NUMDAYS	LOB	PCSA	AGE_BAND	MEMBERS
1	2012	201201	31	COMMERCIAL	EAST	000-004	7823
2	2012	201201	31	COMMERCIAL	EAST	005-009	8229
3	2012	201201	31	COMMERCIAL	EAST	010-014	9010
4	2012	201201	31	COMMERCIAL	EAST	015-019	10288
5	2012	201201	31	COMMERCIAL	EAST	020-024	10959
6	2012	201201	31	COMMERCIAL	EAST	025-029	10239

Output 5. An extract from our PROC MEANS results, which counted members by every combination of our dimensions

The ‘MEMBERS’ column gives the count of members by month, lob, pcsa, and age band.

CALCULATE ROLLING 12 MONTH TOTALS

Now that we have our monthly members and monthly discharge days by the dimensions we are interested in, we can calculate rolling 12 month and YTD totals.

ROLLING 12 MONTH DISCHARGE DAY TOTALS

Using PROC EXPAND we can get rolling 12 month Discharge Days and Memberships by each of our combinations of dimensions. First we must sort the data in preparation for using PROC EXPAND. Because we want to sum our facts across consecutive months (and consecutive records), we sort the data by each dimension, with our month (yr_mon) dimension as the last sort field.

```
PROC SORT data=day_counts;
     by lob pcsa age_band yr_mon;
RUN;
```

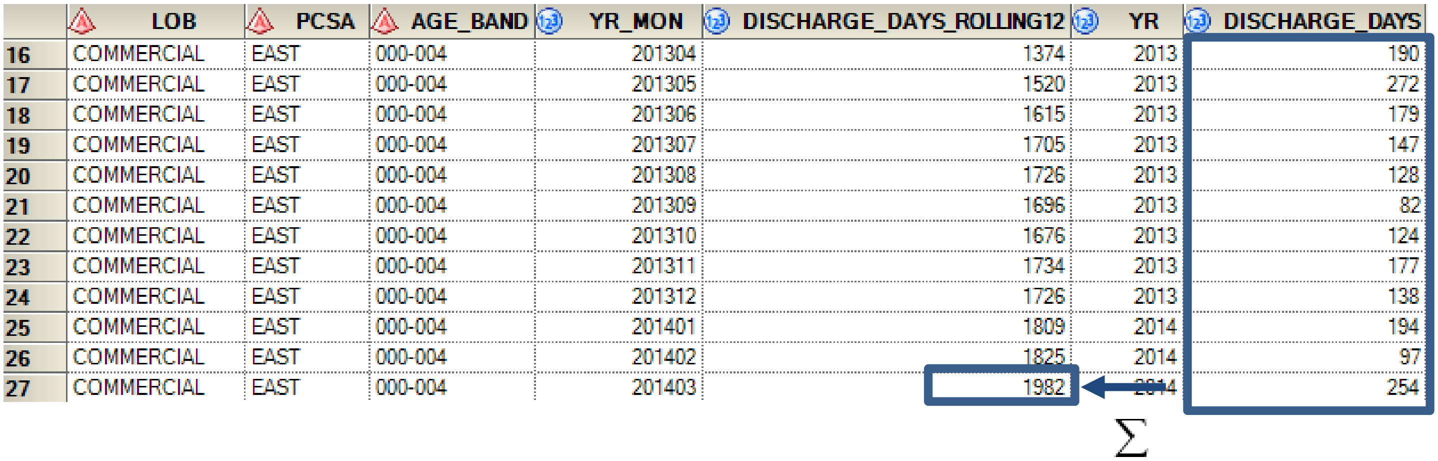
Now we are ready to utilize the power of PROC EXPAND:

```
PROC EXPAND data=day_counts
     out=days_summary;
     by lob pcsa age_band;
     id  yr_mon;
     convert discharge_days=DISCHARGE_DAYS_ROLLING12
     /transformout=(movsum 12);
RUN;
```

The BY statement contains the first three of our four dimensions that we are grouping our rolling 12 month patient days by.

The ID statement is the fourth variable we are grouping on, and it is usually a date or time dimension. Remember that this should be the last variable that your data is sorted on.

Finally, the TRANSFORMOUT option indicates the mathematical calculation to be performed. 'MOVSUM 12' will calculate a 12 month moving sum on the fact variable listed in the convert statement. The result will be a sum of the current and previous 11 records. Output 6 displays an extract of the results:



	LOB	PCSA	AGE_BAND	YR_MON	DISCHARGE_DAYS_ROLLING12	YR	DISCHARGE_DAYS
16	COMMERCIAL	EAST	000-004	201304	1374	2013	190
17	COMMERCIAL	EAST	000-004	201305	1520	2013	272
18	COMMERCIAL	EAST	000-004	201306	1615	2013	179
19	COMMERCIAL	EAST	000-004	201307	1705	2013	147
20	COMMERCIAL	EAST	000-004	201308	1726	2013	128
21	COMMERCIAL	EAST	000-004	201309	1696	2013	82
22	COMMERCIAL	EAST	000-004	201310	1676	2013	124
23	COMMERCIAL	EAST	000-004	201311	1734	2013	177
24	COMMERCIAL	EAST	000-004	201312	1726	2013	138
25	COMMERCIAL	EAST	000-004	201401	1809	2014	194
26	COMMERCIAL	EAST	000-004	201402	1825	2014	97
27	COMMERCIAL	EAST	000-004	201403	1982	2014	254

Output 6. An extract from our PROC EXPAND results, which created rolling 12 month sums of discharge days

ROLLING 12 MONTH AVERAGE MEMBERSHIP

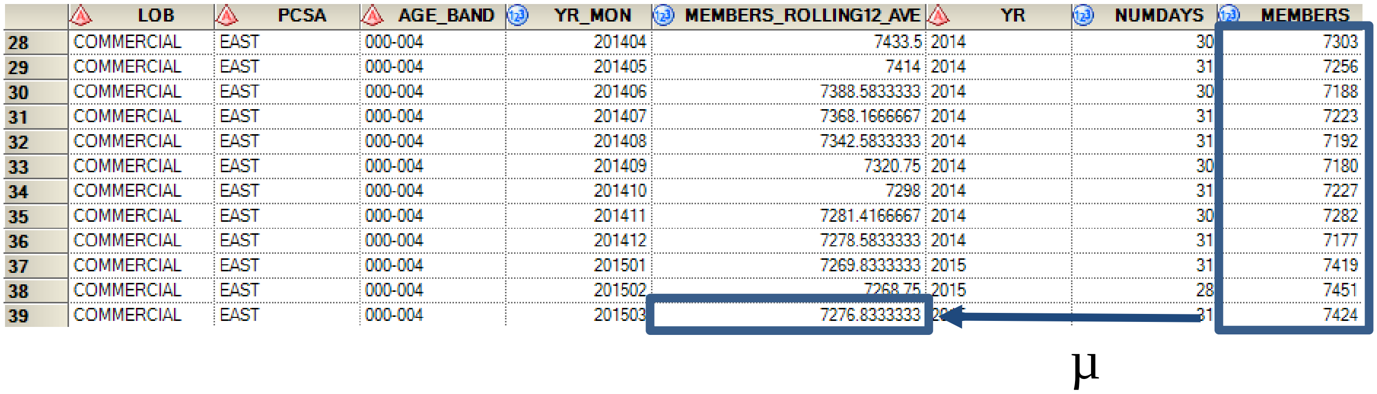
We start by again sorting our dataset:

```
PROC SORT data=member_counts;
     by lob pcsa age_band yr_mon;
RUN;
```

For membership, we are interested in calculating the rolling 12 month average of membership counts:

```
PROC EXPAND data=member_counts
     out=member_summary;
     by lob pcsa age_band;
     id  yr_mon;
     convert MEMBERS=MEMBERS_ROLLING12_AVE/
     transformout=(movave 12);
RUN;
```

Again, the TRANSFORMOUT option indicates the mathematical calculation to be performed. 'MOVAVE 12' will calculate a 12 month moving average on the fact variable listed in the convert statement. The result will be an average of members of the current and previous 11 records. Output 7 displays an extract of the results:



	LOB	PCSA	AGE_BAND	YR_MON	MEMBERS_ROLLING12_AVE	YR	NUMDAYS	MEMBERS
28	COMMERCIAL	EAST	000-004	201404	7433.5	2014	30	7303
29	COMMERCIAL	EAST	000-004	201405	7414	2014	31	7256
30	COMMERCIAL	EAST	000-004	201406	7388.5833333	2014	30	7188
31	COMMERCIAL	EAST	000-004	201407	7368.1666667	2014	31	7223
32	COMMERCIAL	EAST	000-004	201408	7342.5833333	2014	31	7192
33	COMMERCIAL	EAST	000-004	201409	7320.75	2014	30	7180
34	COMMERCIAL	EAST	000-004	201410	7298	2014	31	7227
35	COMMERCIAL	EAST	000-004	201411	7281.4166667	2014	30	7282
36	COMMERCIAL	EAST	000-004	201412	7278.5833333	2014	31	7177
37	COMMERCIAL	EAST	000-004	201501	7269.8333333	2015	31	7419
38	COMMERCIAL	EAST	000-004	201502	7268.75	2015	28	7451
39	COMMERCIAL	EAST	000-004	201503	7276.8333333	20	31	7424

Output 7. An extract from our PROC EXPAND results, which created rolling 12 month averages of members

CALCULATE YEAR TO DATE TOTALS

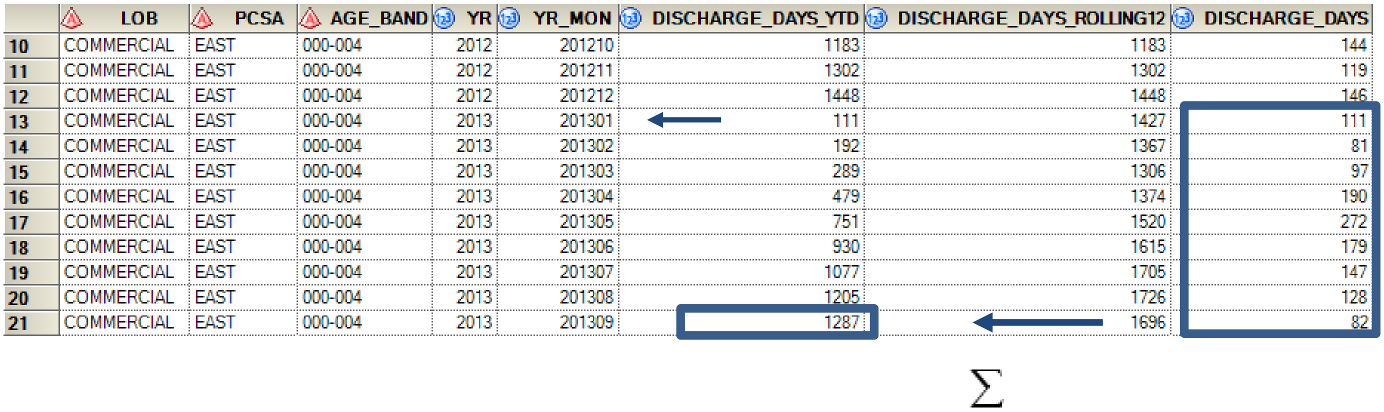
DISCHARGE DAY YEAR TO DATE TOTALS

Using PROC EXPAND we can get year to date Discharge Days and Memberships by each of our combinations of dimensions. To do this, we need a year variable in our BY statement. Since we're performing a year to date calculation, we will not be using data from different years, and adding a year dimension to the BY statement prevents this from occurring. We will run this code off of our 'days_summary' table, which was the output of the PROC EXPAND statement that created the sum of discharge days by rolling 12 months.

```
PROC EXPAND data=days_summary
     out=days_summary_ytd;
     by lob pcsa age_band yr;
     id  yr_mon;
     convert discharge_days=discharge_days_ytd/
     transformout=(cusum 1);
RUN;
```

The TRANSFORMOUT option indicates the mathematical calculation to be performed. ‘CUSUM 1’ will calculate a YTD sum on the fact variable listed in the CONVERT statement, which in our example is ‘discharge days’. The result will be a sum of the current record and the previous records going back to January. SAS knows to stop the calculation at the January record because if it goes back any further the ‘yr’ variable will change to the previous year.

Output 8 displays an extract of the results:



	LOB	PCSA	AGE_BAND	YR	YR_MON	DISCHARGE_DAYS_YTD	DISCHARGE_DAYS_ROLLING12	DISCHARGE_DAYS
10	COMMERCIAL	EAST	000-004	2012	201210	1183	1183	144
11	COMMERCIAL	EAST	000-004	2012	201211	1302	1302	119
12	COMMERCIAL	EAST	000-004	2012	201212	1448	1448	146
13	COMMERCIAL	EAST	000-004	2013	201301	111	1427	111
14	COMMERCIAL	EAST	000-004	2013	201302	192	1367	81
15	COMMERCIAL	EAST	000-004	2013	201303	289	1306	97
16	COMMERCIAL	EAST	000-004	2013	201304	479	1374	190
17	COMMERCIAL	EAST	000-004	2013	201305	751	1520	272
18	COMMERCIAL	EAST	000-004	2013	201306	930	1615	179
19	COMMERCIAL	EAST	000-004	2013	201307	1077	1705	147
20	COMMERCIAL	EAST	000-004	2013	201308	1205	1726	128
21	COMMERCIAL	EAST	000-004	2013	201309	1287	1696	82

Output 8. An extract from our PROC EXPAND results, which created year to date sums of discharge days

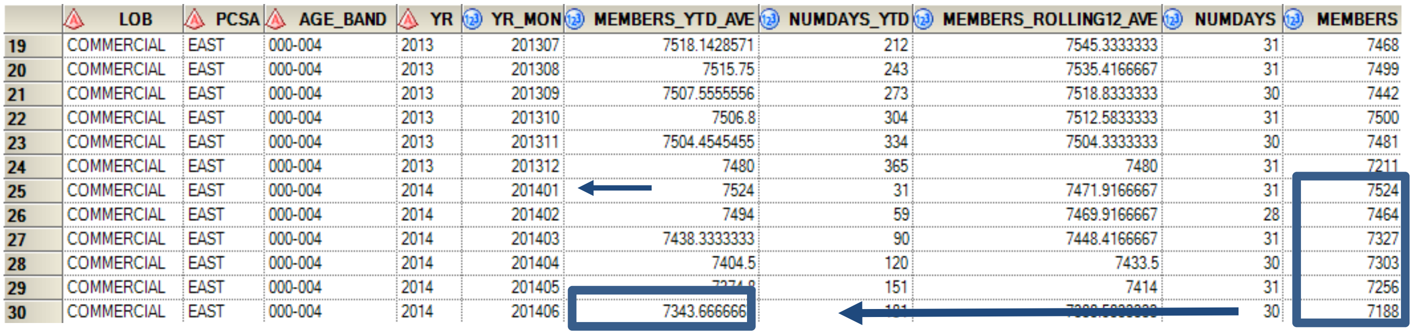
YEAR TO DATE AVERAGE MEMBERSHIP

For membership, we are now interested in calculating the YTD average. We will run this code off of our ‘member_summary’ table, which was the output of the PROC EXPAND statement that created the count of members by rolling 12 months.

```
PROC EXPAND data=member_summary
     out=member_summary_ytd;
     by lob pcsa age_band yr;
     id  yr_mon;
     convert members=members_ytd_ave /
     transformout=(cuave 1);
     convert numdays=numdays_ytd /
     transformout=(cusum 1);
RUN;
```

Again we have a year variable in our dimensions, since we are performing a YTD calculation. The TRANSFORMOUT=(CUAVE 1) option calculates the YTD average membership for the variable in the CONVERT statement, ‘members’, which is our membership counts.

Also note that to calculate a YTD rate we need the number of calendar days in the year through the given month. For this we can perform a TRANSFORMOUT=(CUSUM 1) on the variable ‘numdays’, which represents the number of days in a given month. Output 9 below is an extract of the results:



	LOB	PCSA	AGE_BAND	YR	YR_MON	MEMBERS_YTD_AVE	NUMDAYS_YTD	MEMBERS_ROLLING12_AVE	NUMDAYS	MEMBERS
19	COMMERCIAL	EAST	000-004	2013	201307	7518.1428571	212	7545.3333333	31	7468
20	COMMERCIAL	EAST	000-004	2013	201308	7515.75	243	7535.4166667	31	7499
21	COMMERCIAL	EAST	000-004	2013	201309	7507.5555556	273	7518.8333333	30	7442
22	COMMERCIAL	EAST	000-004	2013	201310	7506.8	304	7512.5833333	31	7500
23	COMMERCIAL	EAST	000-004	2013	201311	7504.4545455	334	7504.3333333	30	7481
24	COMMERCIAL	EAST	000-004	2013	201312	7480	365	7480	31	7211
25	COMMERCIAL	EAST	000-004	2014	201401	7524	31	7471.9166667	31	7524
26	COMMERCIAL	EAST	000-004	2014	201402	7494	59	7469.9166667	28	7464
27	COMMERCIAL	EAST	000-004	2014	201403	7438.3333333	90	7448.4166667	31	7327
28	COMMERCIAL	EAST	000-004	2014	201404	7404.5	120	7433.5	30	7303
29	COMMERCIAL	EAST	000-004	2014	201405	7374.8	151	7414	31	7256
30	COMMERCIAL	EAST	000-004	2014	201406	7343.666666	181	7388.5833333	30	7188

μ

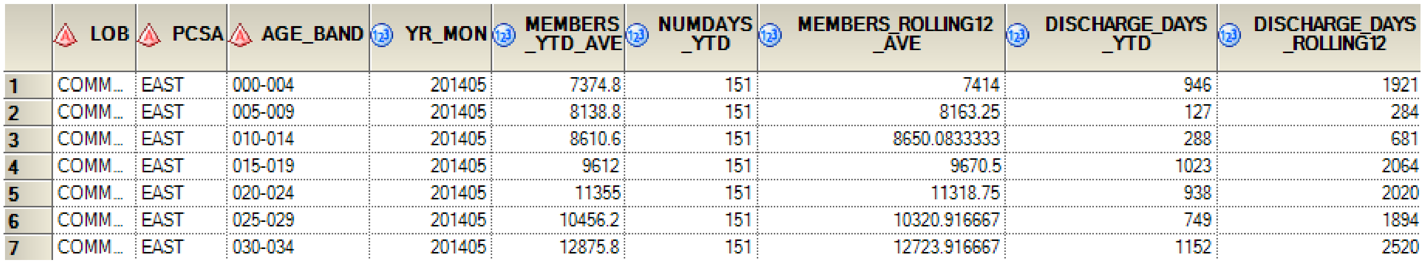
Output 9. An extract from our PROC EXPAND results, which created year to date averages of members

CALCULATE PATIENT DAY RATE

Both our datasets now have all the variables we need, on each record, to calculate the patient day rate for all possible combinations of dimensions. We now can merge our discharge days and membership datasets together by our dimension variables.

```
DATA inpatient_dayrate (drop= members numdays discharge_days);
    merge member_summary_ytd days_summary_ytd;
    by yr_mon lob pcsa age_band;
RUN;
```

Each of the metrics we need for our final days rates is present for every combination of our 4 dimensions. Output 10 is an extract of the results:



	LOB	PCSA	AGE_BAND	YR_MON	MEMBERS_YTD_AVE	NUMDAYS_YTD	MEMBERS_ROLLING12_AVE	DISCHARGE_DAYS_YTD	DISCHARGE_DAYS_ROLLING12
1	COMM...	EAST	000-004	201405	7374.8	151	7414	946	1921
2	COMM...	EAST	005-009	201405	8138.8	151	8163.25	127	284
3	COMM...	EAST	010-014	201405	8610.6	151	8650.0833333	288	681
4	COMM...	EAST	015-019	201405	9612	151	9670.5	1023	2064
5	COMM...	EAST	020-024	201405	11355	151	11318.75	938	2020
6	COMM...	EAST	025-029	201405	10456.2	151	10320.916667	749	1894
7	COMM...	EAST	030-034	201405	12875.8	151	12723.916667	1152	2520

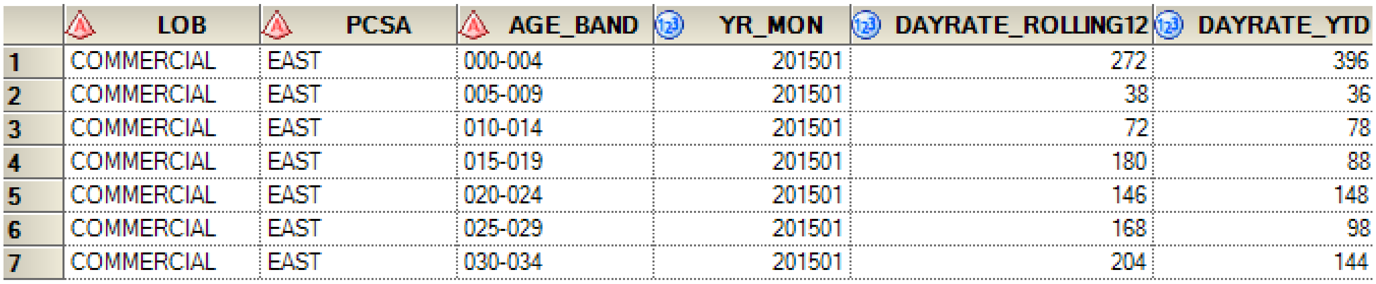
Output 10. An extract from our MERGE results, which includes all of the metrics needed for our final patient day rate calculations

Now we calculate our rolling 12 month and YTD day rates. Recall our patient day rate formula:

$$\text{PDR} = \left(\frac{\text{Total } \textit{Discharge Days}}{(\text{Average } \textit{Memberships} \text{ / 1000 })} \right) \text{ X } (365 \text{ / Time Period in Days })$$

```
DATA final_rates (drop=members_ytd_ave members_rolling12_ave
numdays_ytd discharge_days_rolling12 discharge_days_ytd);
     set inpatient_dayrate;
      DAYRATE_ROLLING12
      =round((discharge_days_rolling12/(members_rolling12_ave/1000))*(3
      65/365),2);
      DAYRATE_YTD
      =round((discharge_days_ytd/(members_ytd_ave/1000))*(365/numdays_y
      td),2);
RUN;
```

Output 11 below is an extract of our final results, which matches the example extract in Output 1 that we displayed as our goal at the beginning of this paper:



	LOB	PCSA	AGE_BAND	YR_MON	DAYRATE_ROLLING12	DAYRATE_YTD
1	COMMERCIAL	EAST	000-004	201501	272	396
2	COMMERCIAL	EAST	005-009	201501	38	36
3	COMMERCIAL	EAST	010-014	201501	72	78
4	COMMERCIAL	EAST	015-019	201501	180	88
5	COMMERCIAL	EAST	020-024	201501	146	148
6	COMMERCIAL	EAST	025-029	201501	168	98
7	COMMERCIAL	EAST	030-034	201501	204	144

Output 11. An extract from our final Patient Day Rate results

CONCLUSION

Understanding how a health plan membership utilizes inpatient hospital services is essential for decision making within any health care organization. In this paper we created year to date and rolling 12 month patient day rates by every combination of the dimensions month, line of business, primary service area, and age band. To accomplish this, we utilized the SUM= and N= options within PROC MEANS to sum our inpatient discharge days and count our members by every combination of our four dimensions. After properly sorting our data, we then used PROC EXPAND to create rolling and year to date sums and averages on our fact columns (discharge days and members) across several records. Finally we applied our patient day rate formula to calculate our final rates.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Thomas Gant
Data & Information Management Enhancement (DIME)
Kaiser Permanente
503-813-4906
thomas.e.gant@kp.org