

**Using Dastep programming and Proc SQL in SAS®
To calculate daily hospital occupancy**

**Renu Gehring
Managing Partner
Ace-Cube, LLP, Beaverton, Oregon**

**Rajiv Sharma
Department of Economics,
Portland State University, Portland, Oregon**

ABSTRACT

This paper describes the advantages and disadvantages of two distinct approaches to computing hospital daily occupancy in large datasets. The first approach employs an incremental method for computing occupancy. Initial occupancy is calculated by using the *SQL* procedure and then a *Retain* statement in a datastep creates a running sum of daily occupancy. The second approach uses a data expansion technique; A *do loop* in a datastep converts the patient level data to patient-day grain. After the data is expanded, dummy (zero/one) variables are created to indicate whether a patient has been admitted, discharged, or is currently in the hospital on a given day. The counter variables for admissions, discharges, and current occupancy are summed to create a daily tabulation of hospital census, discharges, and admissions. Although both approaches yield similar results in the absence of data quality issues, they are quite different in their logic and programming technique. The two methodologies were developed in the context of research into how fluctuations in demand affect hospital behavior and are relevant and applicable to any business situation involving a stock and flow relationship. The two techniques can also pinpoint deficiencies in the data. The computations yield results that are used to analyze patterns in hospital occupancy.

INTRODUCTION

Accurate computation of hospital occupancy is important for many reasons. Hospital administrators and other personnel utilize hospital and departmental occupancy in making long-term decisions about expansion or contraction of hospital services as well as the nature of hospital organization. In the 1990s, as managed care penetration resulted in lower occupancy rates for hospitals not owning/offering HMO plans, hospitals responded by converting their status from non-profit to for-profit. In order to increase occupancy rates, hospitals also began to create/buy health plans. (Managed Care Interface, 1997)

Occupancy rates are also important for evaluating hospitals' day-to-day operational decisions. Often, emergency departments (ED) may divert ambulances on the basis of ED overcrowding. Such actions are more costly in terms of foregone revenue when

occupancy is low than when occupancy is close to capacity. High hospital occupancy in turn poses a physical constraint to EDs as the backlog of patients waiting to be admitted through the ED increases.

Health care researchers utilize hospital occupancy as a proxy for the demand for hospital services. Baker et. al (2004) study the impact of fluctuations in hospital occupancy in California on hospital costs. Sharma et. al (2005) use hospital occupancy in their formulation of demand for hospital services to analyze how hospitals' discharge and admission practices are impacted during peak and off-peak times. Health care researchers such as Forrster et. al (2000) are also interested in the link, negative or positive, between high hospital occupancy and quality of care.

This paper outlines two distinct programming approaches to computing hospital daily occupancy from patient discharge records. Occupancy is defined as the number of patients in a hospital on a given day. The first approach details an incremental method for computing occupancy. Using the dates of admission and discharge, initial occupancy is calculated. The initial value of occupancy is then incremented by a *Retain* statement in a datastep. The second approach employs a data expansion technique; Using the date of admission and length of stay variables, a *do loop* in a datastep converts the patient level data to patient-day. Dummy (zero/one) variables are created for admissions, discharges, and current occupancy which are summed to create a daily tabulation of hospital census, discharges, and admissions. Although both approaches yield similar results in the absence of data quality issues, they are quite different in their logic and programming technique. The pros and cons of both methods are described in this paper. Selected results from an analysis of fluctuations in hospital occupancy are also presented.

EXISTING COMPUTATIONS OF HOSPITAL OCCUPANCY

Daily hospital census has been computed by Baker et.al (2004) They use discharge records to compute daily hospital occupancy for California hospitals in order to estimate how variability in demand affects hospital costs. For the purposes of methodology comparison, Professor Baker was contacted but was unable to give details on the hospital occupancy computations used in their paper.

Daily hospital occupancy was calculated as a part of an academic paper by Dr. Rajiv Sharma, professor of economics at Portland State University, Renu Gehring, Managing Partner at Ace-Cube, LLP, and Dr. Miron Stano, professor of economics at Oakland University. The paper entitled "Fluctuations in Short-term Demand: Implications for Hospital Admission and Discharge Behavior" examines how short- term fluctuations in demand affect hospitals' admission and discharge behavior. This paper outlines the approaches used in Sharma et. al (2005).

DATA USED

Discharge data from Oregon hospitals from 1990 to 1998 was obtained from Oregon Office of Health Policy and Research. These data have been stripped of elements such as names and addresses that would permit patients to be individually identified. Daily occupancy, at the hospital and at the hospital-department level, was computed for each of the 9 years between 1990 and 1998. Table 1 gives number of discharge records processed and analyzed by year.

Table 1.—Discharges from Oregon Hospitals

Year	Number of Discharge Records
1990	348,217
1991	342,156
1992	341,200
1993	368,610
1994	364,927
1995	350,365
1996	369,870
1997	376,207
1998	381,246

For the purpose of Sharma et. al (2005), we analyze the 381,499 records that comprise inpatient discharges from Oregon hospitals between December 1, 1997 and November 30, 1998. The data contain discharge records from 65 hospitals. The single hospital with the largest number of patients accounted for about 8% of discharges, whereas the smallest accounted for less than .02%. The 7 largest hospitals comprised 41% of all discharges. Each record contained patient diagnosis, date of birth, date of admission, date of discharge, and information on payor characteristics. The data do not include “day-surgery” or “short-stay” patients who obtain treatment in a hospital, but are never formally admitted. We exclude records associated with lengths of stay over one year, as well as records where my computation of length of stay based on admission and discharge dates was not in concert with the value contained in length of stay variable included in the raw data. Together these exclusions amounted to less than 0.2% of records.

COMPUTING DAILY HOSPITAL OCCUPANCY

Using discharge date, admission date, and length of stay variables, two approaches to calculating hospital occupancy are outlined. Both approaches use datastep programming and work well in dealing with large datasets.

Calculating hospital occupancy by increments

In this approach, an initial date, for example, January 1, 1998, is picked as the starting point of all calculations. Conditional logic is used to determine hospital occupancy on that day. Patients with an admission date prior to or the equal to January 1, 1998 and with a discharge date later than January 1, 1998 are said to be in the pool of inpatient patients in the hospital. Net daily admissions, the difference between admissions and discharges, beginning with the following day, January 2, 1998, are calculated. Using running sum logic in a datastep, net daily admissions are added to the initial occupancy value to compute daily hospital and within department occupancy.

The following code segments describe the computation of overall hospital occupancy. With the help of arrays and do-loops, this approach was extended to calculate occupancy by major diagnostic codes (MDC) and aggregations of ICD-9 codes. The use of macro variables also allows for variable start dates.

After selecting a subset of data, the following code computes the initial value of hospital occupancy.

```
*t_num is set to 1 for each patient. When summed;  
*this represents total number of patients;1  
t_num=1;  
if date_admit<='1JAN98'D and date_disch>'1JAN98'D;
```

```
proc sql;  
  create table occup_st as  
  select h_name, sum(t_num) as st_total  
  from occup_begin  
  group by h_name;  
run;
```

Daily admissions and discharges are calculated and the two datasets are merged to create net daily admissions.

```
proc sql;  
  create table admit as  
  select h_name, date_admit as date1, count(*) as num_admit,  
  from occup  
  where date_admit>'1JAN98'D  
  group by h_name, date_admit;  
run;
```

```

proc sql;
  create table discharge as
  select h_name,date_disch as date1, count(*) as num_disch,
  from occup
  where date_disch >'1JAN98'D
  group by h_name, date_disch;
run;

```

```

data all;
  merge discharge (in=in1) admit(in=in2);
  by date1;
  if in1 or in2;
  if num_disch=. then num_disch=0;
  if num_admit=. then num_admit=0;
  net_admit=num_admit-num_disch;
run;

```

The following code concatenates the dataset containing the initial value of hospital occupancy with the dataset containing net daily admissions. It then creates a running sum of hospital occupancy.

```

data occup1;
  retain occup_total 0;
  set occup_st all
  *the first observation is from dataset occup_st;
  if _N_=1 then occup_total=st_total;
  else occup_total=occup_total+net_admit;
run;.

```

The advantages of the incremental approach are that it is elegant and lends itself well to macro programming. Macro variables can be used to create user defined initial values, such as initial dates. The disadvantage of the approach is that hospital occupancy on any given day is affected by the data quality of the discharge dataset on any of the previous days. Poor data on discharges and admissions on any given day may affect the accuracy of the entire hospital occupancy calculation.

Calculating hospital occupancy by data expansion

Using admission date and length of stay variables, the data grain is converted from patient level to patient-day level. Table 2 shows two discharge records (mock examples) at the patient level.

Table 2.—Sample Records (Patient level)

Patient ID	Date of Admission	Date of Discharge	Length of Stay in Days
100001	01/03/1999	01/06/1999	3
100002	04/05/1998	04/10/1998	5

Table 3 shows the data at the patient-day level. The two date variables, dates of admission and discharge, are now replaced by a non specific date variable, converting the data grain from patient to patient-day level.

Table 3.—Sample Records (Patient-Day level)

Patient ID	Date	Length of Stay in Days	Day of Stay	Actual Remaining Length of Stay in Days
100001	01/03/1999	3	0	3
100001	01/04/1999	3	1	2
100001	01/05/1999	3	2	1
100001	01/06/1999	3	3	0
100002	04/05/1998	5	0	5
100002	04/06/1998	5	1	4
100002	04/07/1998	5	2	3
100002	04/08/1998	5	3	2
100002	04/09/1998	5	4	1
100002	04/10/1998	5	5	0

We also compute day of stay, which is defined as 0 for the day the patient is admitted and equals the length of stay when the patient is discharged. Subtracting day of stay from length of stay gives us actual remaining days. Note that on the day of admission, the actual remaining days equals length of day. On the day of discharge, the actual remaining days is equal to 0.

Below code sample shows the technique used to expand the data given in Tables 2 and 3.

```

losplus1=los+1;
do i=1 to losplus1;
  *nlos is day of stay;
  nlos=i-1;
  date1=intnx('DAY',date_admit,i-1);
  *actual remaining days in hospital;
  act_rlos=los-nlos;
  output;
end;
drop i;

```

Counter variables are created for occupancy, discharge, and admissions.

```
*Discharges;  
if act_rlos=0 then dischg=1;  
else dischg=0;  
*Admissions;  
if act_rlos=los then admit=1;  
else admit=0;  
*Occupancy counter, t_num, set to 0 when discharge occurs;  
if dischg=1 then t_num=0;
```

This data is now summed so that we have daily hospital occupancy. Admissions and discharges can also be summed up in a similar manner.

```
proc sql;  
create table occup as  
select hosp_id,date1, sum(t_num) as occup  
from occup_exp2  
group by hosp_id,date1;  
run;
```

Although data intensive, this process works well with large datasets. Poor data quality on a given day affects the measure of hospital occupancy on the same day, rather than perpetuating the problem in the entire dataset. This approach is elegant as well as simple as the code used is relatively short and easy to understand. With minor code changes, the approach can be easily extended to computing department level occupancy.

Comparisons and contrasts between two techniques

Data quality affects the two techniques in a different manner. As the incremental approach calculates hospital occupancy by incrementing net daily admissions to the previous day's tally, it perpetuates sporadic data errors throughout the entire computation. On the other hand, poor data quality on any given day adversely impacts the computation yielded by the data expansion technique on that day only. This is the reason that data expansion technique is preferred when there are any concerns as to the quality of the data.

In the absence of data quality issues, the two approaches yield similar results², verifying each other's logic and code. Table 4 evaluates the two techniques according to four main performance criteria.

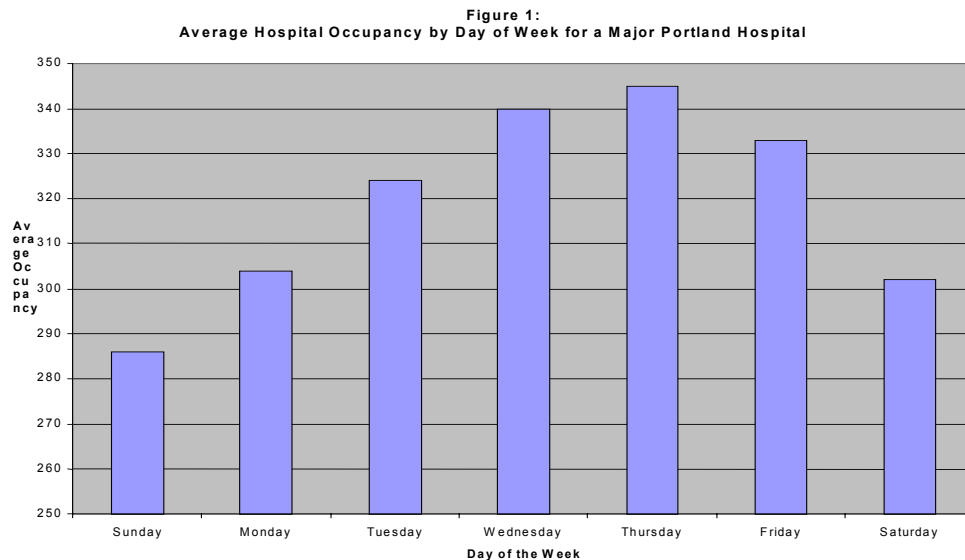
Table 4.—Relative Performance of Incremental and Data Expansion approaches

Criterion	Relative Performance
Processing Speed	Incremental Approach is 2-10 times faster.
Lines of Code	Data Expansion Approach is 2 times more compact.
Code Flexibility	Data Expansion Approach requires fewer changes.
Ease of Understanding	Incremental Approach is easier for the novice SAS programmer to grasp and utilize whereas the Data Expansion Approach uses more advanced code.

As Table 4 shows, the incremental approach is considerably faster than the data expansion approach, although both techniques are well adapted to large datasets. Despite its relative slow processing speed, the data expansion approach was used, with relative ease, to compute occupancy at the hospital level for each of the nine years in the dataset. The data expansion approach is more compact, requiring half as many lines of code generated by the incremental approach. This provides a considerable advantage as the compactness of code leads to efficiencies in time spent on code maintenance and enhancement. Perhaps the best feature of the data expansion technique is its portability. Minor code changes in the data expansion approach allow the user to drill down occupancy counts to department level or MDC level whereas adapting the incremental technique requires more code changes.

SELECTED RESULTS

Calculations of hospital occupancy are used by Sharma et. al (2005) in their demand for hospital services to analyze how hospitals' discharge and admission practices are impacted during peak and off-peak times. Although the results discussed in Sharma et. al are beyond the scope of this paper, we present fluctuations in daily hospital occupancy by day of week for a major Portland hospital. In Figure 1, note that average hospital occupancy increases throughout the week, peaking on Thursday and reaching its lowest point on Sunday. The finding that days of the week are significant predictors of hospital demand can be utilized by hospital scheduling department to schedule elective procedures on comparatively less busy days, thereby smoothing resource use and optimizing revenue.



CONCLUSION

This paper outlined the importance of accurately computing daily hospital occupancy. It detailed two programmatic approaches to calculating hospital census and gave the pros and cons of each approach. Selected results from an analysis of fluctuations in hospital daily occupancy were presented in this paper.

The approaches outlined in this paper are applicable beyond the health care industry. Hotels are also interested in keeping an accurate tally of guests and capacity and would benefit from the approaches outlined in this paper. Other useful applications include inventory management, customer retention management and any other business situation involving a stock and flow relationship.

REFERENCES

Baker L, Phibbs C, Guarino C, Supina, D, Reynolds, J 2004 Within-year variation in hospital utilization and its implications for hospital costs. *Journal of Health Economics* 23 (2004)

Hospital Occupancy Rates: The effect of Managed Care on Inpatient Utilization. Managed Care Interface, 1997

Forrester A, Tugwell, P, Van Walraven C 2000. An hypothesis paper on practice environment and the provision of health care: Could hospital occupancy rates effect quality? *Journal of Quality Clinical Practises* 20 (2000)

Sharma R, Gehring, R., Stano, M. 2005. Fluctuations in Short Term Demand: Implications for Hospital Admission and Discharge Behavior.

ENDNOTES

- 1 Any code sample beginning with a * and ending with ; is a comment.
- 2 A maximum difference of 2.5% exists between the daily census results generated by the two techniques. This is because the incremental technique uses the date of discharge and the data expansion approach uses the length of stay (LOS) variable already computed in the data.

ACKNOWLEDGEMENTS

SAS is a registered trademark of SAS Institute, Inc. in the USA and other countries. Other brand and product names are registered trademarks or trademarks of their respective companies.

® indicates USA registration.

CONTACT INFORMATION

If you have any questions regarding this paper, please contact:

Renu Gehring
Ace-Cube, LLP
17715 NW Fall Ct
Beaverton, OR 97006
Phone: 503-629-8677
Email: Renu.Gehring@ace-cube.com

SAS is a registered trademark of SAS Institute Inc. in the USA and other countries. ® Indicates USA registration.

