Top 5 Handy PROC SQL Tips You Didn’t Think Were Possible

Montreal SAS users Group
30 May 2018
11:00-11:40

Charu Shankar
SAS Institute, Toronto

About your presenter

SAS Senior Technical Training Specialist, Charu Shankar teaches by engaging with logic, visuals and analogies to spark critical thinking. She interviews clients to recommend the right SAS training. She is a frequent blogger for the SAS Training Post.

When she's not teaching technology, she is passionate about helping people come alive with yoga and is a food blogger. www.charuyoga.com

Charu has presented at over 100 SAS international user group conferences on topics related to SAS programming, SQL , DS2 programming, tips and tricks with coding, new features of SAS and SAS Enterprise Guide.

AGENDA

1. Join tables dynamically
2. Create Inline views for joins
3. Pivot like a dancer with the Boolean
4. Recognize patterns in data
5. Create running totals
6. Questions
7. Useful Links

PROC SQL is a powerful language that can express many of your queries simply and with clarity. Users who are continuously improving process and looking to stay within PROC SQL to analyze and process data will benefit from this session. Come learn to maximize human & computing efficiency elegantly

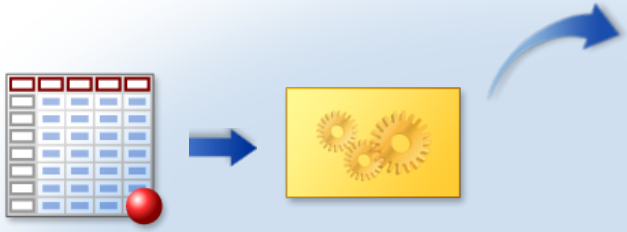
1. JOIN TABLES DYNAMICALLY

How can I look up Metadata?

1. JOIN TABLES DYNAMICALLY

Business Scenario

List all common columns for a SQL Join. Added bonus: Use a technique that's not PROC CONTENTS or PROC DATASETS because they simply analyze a single dataset. We want common columns to be found dynamically by exploring metadata stored by SAS



Column Name	Member Name	Column Type	Column Length
ACTUAL	PRDSAL2	num	8
ACTUAL	PRDSALE	num	8
ACTUAL	PRDSAL3	num	8
ALIAS_CITY	ZIPCODE	char	300
ALIAS_CITY	ZIPMIL	char	300
ALIAS_CITYN	ZIPMIL	char	300
ALIAS_CITYN	ZIPCODE	char	300
AMOUNT	NVST2	num	8
AMOUNT	RENT	num	8
AMOUNT	NVST1	num	8
AMOUNT	ROCKPIT	num	8
AMOUNT	NVST3	num	8
AMOUNT	NVST5	num	8
AMOUNT	NVST4	num	8
AMOUNT	BUY	num	8
APPLNAME	EISRG	char	8
APPLNAME	SASAPPL	char	8

1. JOIN TABLES DYNAMICALLY

Know your dictionary table

```
proc sql;
        describe table dictionary.columns;
NOTE: SQL table DICTIONARY.COLUMNS was created like:
create table DICTIONARY.COLUMNS
  (
   libname char(8) label='Library Name',
   memname char(32) label='Member Name',
   memtype char(8) label='Member Type',
   name char(32) label='Column Name',
   type char(4) label='Column Type',
   length num label='Column Length',
   npos num label='Column Position',
   varnum num label='Column Number in Table',
   label char(256) label='Column Label',
   format char(49) label='Column Format',
   informat char(49) label='Column Informat',
   idxusage char(9) label='Column Index Type',
   sortedby num label='Order in Key Sequence',
   xtype char(12) label='Extended Type',
   notnull char(3) label='Not NULL?',
   precision num label='Precision',
   scale num label='Scale',
   transcode char(3) label='Transcoded?'
  );
```

1. JOIN TABLES DYNAMICALLY

Investigate dictionary tables with PROC SQL

```
proc sql;
       select libname, memname, name, type, length
              from dictionary.columns
                     where upcase(name) contains 'ID'
                            and libname='SASHELP' and
type='num';
quit;
```

1. JOIN TABLES DYNAMICALLY

Check the nice output with PROC SQL

Library Name	Member Name	Column Name	Column Type	Column Length
SASHELP	ADSMSG	MSGID	num	8
SASHELP	AFMSG	MSGID	num	8
SASHELP	BURROWS	ID	num	8
SASHELP	CLNMSG	MSGID	num	8
SASHELP	DEMOGRAPHICS	ID	num	8
SASHELP	EISMKIL	ID	num	8
SASHELP	EISMKMX	ID	num	8
SASHELP	EISMKNA	ID	num	8
SASHELP	EISMKUS	ID	num	8
SASHELP	EISMSG	MSGID	num	8
SASHELP	ETSMSG	MSGID	num	8
SASHELP	FISH	Width	num	8
SASHELP	GEOEXS	TLID	num	8
SASHELP	GNGMSG	MSGID	num	8
SASHELP	HUMID	Humidity	num	8
SASHELP	IMGMSG	msgid	num	8
SASHELP	IRIS	SepalWidth	num	8
SASHELP	IRIS	PetalWidth	num	8

1. JOIN TABLES DYNAMICALLY

Check efficiency of PROC SQL

```
options fullstimer;
proc sql;
        select libname, memname, name, type, length
                from dictionary.columns
                        where upcase(name) contains 'ID'
                                and libname='SASHELP' and type='num';
quit;
NOTE: PROCEDURE SQL used (Total process time):
      real time           0.73 seconds
      user cpu time       0.42 seconds
      system cpu time     0.29 seconds
      memory              5584.18k
      OS Memory           24672.00k
      Timestamp           05/22/2018 01:52:52 PM
      Step Count                        4  Switch Count  36
```

1. JOIN TABLES DYNAMICALLY

What do these stats mean?

Statistic	Description
Real Time	the amount of real time (clock time) spent to process the SAS job. Real time is also referred to as elapsed time.
User CPU Time	the CPU time that is spent in the user program.
System CPU Time	the CPU time that is spent to perform operating system tasks (system overhead tasks) that support the execution of your SAS code.
Memory	the amount of memory required to run a step.
OS Memory	the largest amount of operating system memory that is available to SAS during the step.
Timestamp	the date and time that a step was executed.
Step Count	the count of DATA steps or procedures that run in a SAS program.
Switch Count	a count of task switches within a step—that is, within a DATA step or procedure—in a SAS program. A task switch occurs when a step requests service from another process. Another task switch occurs when the step resumes. The number reported is for the last step that runs.

1. JOIN TABLES DYNAMICALLY

Can I use PROC PRINT instead?

```
options fullstimer;
proc print data=sashelp.vcolumn;
       var libname memname name type length;
       where upcase(name) contains 'ID' and libname='SASHELP'
              and type='num';
run;
NOTE: There were 34 observations read from the data set SASHELP.VCOLUMN.
      WHERE UPCASE(name) contains 'ID' and (libname='SASHELP') and
(type='num');
NOTE: PROCEDURE PRINT used (Total process time):
      real time           2.19 seconds
      user cpu time       0.92 seconds
      system cpu time     1.18 seconds
      memory              6738.81k
      OS Memory           25440.00k
      Timestamp           05/22/2018 02:22:29 PM
      Step Count                        10  Switch Count  44
```

1. JOIN TABLES DYNAMICALLY

How can I make this join dynamic?

```
proc sql;
   select name, memname, type, length from dictionary.columns
      where libname ='SASHELP'
         group by name
            having count(name) > 1
               order by name;
quit;
```

1. JOIN TABLES DYNAMICALLY

Now you know the common cols.
Joins are a breeze

Column Name	Member Name	Column Type	Column Length
ACTUAL	PRDSAL2	num	8
ACTUAL	PRDSALE	num	8
ACTUAL	PRDSAL3	num	8
ALIAS_CITY	ZIPCODE	char	300
ALIAS_CITY	ZIPMIL	char	300
ALIAS_CITYN	ZIPMIL	char	300
ALIAS_CITYN	ZIPCODE	char	300
AMOUNT	NVST2	num	8
AMOUNT	RENT	num	8
AMOUNT	NVST1	num	8
AMOUNT	ROCKPIT	num	8
AMOUNT	NVST3	num	8
AMOUNT	NVST5	num	8
AMOUNT	NVST4	num	8
AMOUNT	BUY	num	8
APPLNAME	EISRG	char	8
APPLNAME	SASAPPL	char	8

2. CREATE INLINE VIEWS FOR JOINS

Use in-line views to simplify coding a complex query.

2. CREATE INLINE VIEWS FOR JOINS

Business Scenario

List all active Sales Department employees who have annual salaries significantly lower (less than 95%) than the average salary for everyone with the same job title.

Employees with Salaries less than
95% of the Average for their Job

Employee_Name	Employee Job Title	Employee Annual Salary	Job_Avg
Ould, Tulsidas	Sales Rep. I	22,710	26,576
Polky, Asishana	Sales Rep. I	25,110	26,576
Voron, Tachaun	Sales Rep. I	25,125	26,576

2. CREATE INLINE VIEWS FOR JOINS

Step 1 Calculate the average salaries for active employees in the Sales Department, grouped by job title.

```
title  'Sales Department Average Salary';
title2 'By Job Title';

proc sql;
   select Job_Title, avg(Salary) as Job_Avg
   from   orion.employee_payroll as p,
          orion.employee_organization as o
   where p.Employee_ID=o.Employee_ID
      and Employee_Term_Date is missing
      and Department="Sales"
   group by Job_Title;
```

s105d06

2. CREATE INLINE VIEWS FOR JOINS

Viewing the Output

Sales Department Average Salary
By Job Title

Job_Title	Job_Avg
Sales Rep. I	26575.76
Sales Rep. II	27347.97
Sales Rep. III	29213.62
Sales Rep. IV	31588.5

2. CREATE INLINE VIEWS FOR JOINS

Can this SELECT statement be used as a subquery?

```
proc sql;
   select Job_Title, avg(Salary) as Job_Avg
   from   orion.employee_payroll as p,
          orion.employee_organization as o
   where p.Employee_ID=o.Employee_ID
      and Employee_Term_Date is missing
      and Department="Sales"
   group by Job_Title;
```

2. CREATE INLINE VIEWS FOR JOINS

In-Line Views

An *in-line view* is a query expression (SELECT statement) that resides in a FROM clause. It acts as a virtual table, used in place of a physical table in a query.

```
PROC SQL;
SELECT *
   FROM
    (in-line view query expression)
     …;
QUIT;
```

In-line views are often useful when you build complex SQL queries.

2. CREATE INLINE VIEWS FOR JOINS

Step 2 Match each employee to a job title group and compare the employee's salary to the group's average to determine whether it is less than 95% of the group average.

```
proc sql;
   select Employee_Name, emp.Job_Title, Salary format=comma7.,
             Job_Avg format=comma7.
   from(select Job_Title, avg(Salary)as Job_Avg format=comma7.
         from orion.employee_payroll as p,
              orion.employee_organization as o
          where p.Employee_ID=o.Employee_ID
          and Employee_Term_Date is missing
          and Department="Sales"
          group by Job_Title) as job, orion.salesstaff as emp
   where emp.Job_Title=job.Job_Title
      and Salary<Job_Avg*.95 and Emp_Term_Date is missing
      order by Job_Title, Employee_Name;
```

s105d07

2. CREATE INLINE VIEWS FOR JOINS

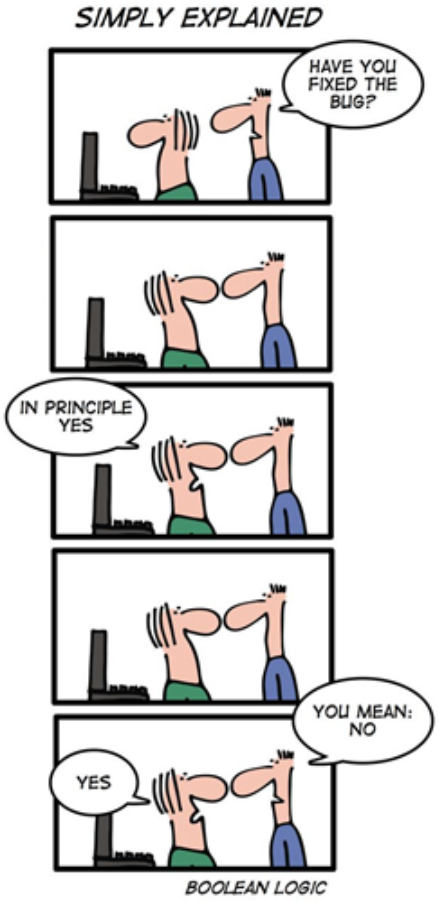
Viewing the Output

Sales Department Average Salary
By Job Title

Employee_Name	Employee Job Title	Employee Annual Salary	Job_Avg
Ould, Tulsidas	Sales Rep. I	22,710	26,576
Polky, Asishana	Sales Rep. I	25,110	26,576
Voron, Tachaun	Sales Rep. I	25,125	26,576

3. PIVOT LIKE A DANCER WITH THE BOOLEAN

THE BOOLEAN GATE



3. PIVOT LIKE A DANCER WITH THE BOOLEAN

Sashelp.bweight --- Infant Birth Weight

The CONTENTS Procedure

Variables in Creation Order

#	Variable	Type	Len	Label
1	Weight	Num	8	Infant Birth Weight
2	Black	Num	8	Black Mother
3	Married	Num	8	Married Mother
4	Boy	Num	8	Baby Boy
5	MomAge	Num	8	Mother's Age
6	MomSmoke	Num	8	Smoking Mother
7	CigsPerDay	Num	8	Cigarettes Per Day
8	MomWtGain	Num	8	Mother's Pregnancy Weight Gain
9	Visit	Num	8	Prenatal Visit
10	MomEdLevel	Num	8	Mother's Education Level

SASHELP.BWEIGHT DATASET

The Sashelp.BWeight data set provides 1997 birth weight data from National Center for Health Statistics (Koenker and Hallock 2001; Abreveya 2001). The data record live, singleton births to mothers between the ages of 18 and 45 in the United States who were classified as black or white. The data set contains 50,000 observations.

3. PIVOT LIKE A DANCER WITH THE BOOLEAN

Over Average weight and under average infant birth weight

Prenatal visit	Infant over average weight	Infant under average weight	Infant Overavg weight/ underavg weight ratio
0	3	8	37.50%
1	17	44	38.64%
2	2	5	40.00%
3	187	248	75.40%

HAVES

HAVE NOTS

3. PIVOT LIKE A DANCER WITH THE BOOLEAN

```
proc sql flow=4 8;
   Title 'Over Average weight and under average infant birth
weight';
   select visit 'Prenatal visit',
       sum(weight > 4000 and married=1 and momsmoke=1) as
           wgt4000 'Infant over average weight',
       sum(weight <=2500 and married=1 and momsmoke=1) as
           wle2500 'Infant under average weight',
       calculated wgt4000/ calculated wle2500 format=percent8.2
           'Infant Overavg weight/ underavg weight ratio'
   from sashelp.bweight
   group by visit;
quit;
```

3. PIVOT LIKE A DANCER WITH THE BOOLEAN

That's not a dance that was a statue.. Let's pivot like a dancer

```
proc sql;
    select married,
        sum(visit >= 1 and momsmoke=0) as visitnonsmokermoms
        'non smoker moms with visits >=1',
        sum(visit >=1 and momsmoke=1) as visitsmokermoms
        'smoker moms with visits >=1'
            from sashelp.bweight
                group by married;
quit;
```

3. PIVOT LIKE A DANCER WITH THE BOOLEAN

That's not a dance that was a statue.. Let's pivot like a dancer

Over Average weight and under average infant birth weight

Married Mother	non smoker moms with visits >=1	smoker moms with visits >=1
0	10987	3130
1	32198	3282

3. PIVOT LIKE A DANCER WITH THE BOOLEAN

Over Average weight and under average infant birth weight

Prenatal visit	Infant over average weight	Infant under average weight	Infant Overavg weight/ underavg weight ratio
0	3	8	37.50%
1	17	44	38.64%
2	2	5	40.00%
3	187	248	75.40%

HAVES

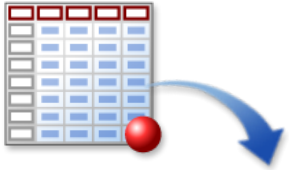
HAVE NOTS

4. RECOGNIZE PATTERNS IN DATA

4. RECOGNIZE PATTERNS IN DATA

Business Scenario

Find data that matches a pattern. HS10_ column has a series of any 10 or 6 digit numbers. An additional challenge- this series never appears in the same position”.



Sample of the data

Sticks or profile shapes of subheading 3916.10

Reproduction proofs for the production of printing plates, rolls, of tariff item No. 8442.50.20

Microcopies of tariff item No. 4903.00.10, 4905.91.00, 4911.10.10 or 4911.10.20

4. RECOGNIZE PATTERNS IN DATA

```
proc sql;
    select * from pattern
        where prxmatch(("/\d{4}\./"),HS10_TSCHED_EDESC)> 0;
```

4. RECOGNIZE PATTERNS IN DATA

HS10_CODE	HS10_TSCHED_EDESC
0810101100	If the aggregate quantity of the goods of this tariff item and of tariff item No. 0810.10.92 imported during the period specified in an order of the Governor in Council specifying limits on the aggregate quantity of goods of this tariff item and of tariff
1701911000	If the aggregate quantity of goods of tariff item Nos. 1701.91.10, 1701.99.10, 1702.90.21, 1702.90.61,1702.90.70 and 1702.90.81 imported during the period specified in an order of the Governor in Council specifying limits on the aggregate quantity of such
1701991000	If the aggregate quantity of goods of tariff item Nos. 1701.91.10, 1701.99.10, 1702.90.21, 1702.90.61,1702.90.70 and 1702.90.81 imported during the period specified in an order of the Governor in Council specifying limits on the aggregate quantity of such
1702902100	If the aggregate quantity of goods of tariff item Nos. 1701.91.10, 1701.99.10, 1702.90.21, 1702.90.61,1702.90.70 and 1702.90.81 imported during the period specified in an order of the Governor in Council specifying limits on the aggregate quantity of such

5. CREATE RUNNING TOTALS

```
data shoes;
    set sashelp.shoes;
    /* Give an index for each row in the shoes dataset*/
    uniq=_n_;
run;

proc sql;
    create table sqlruntot as
        select region, product, sales,
        (select sum(a.sales) from shoes as a
            where a.uniq <= b.uniq) as Running_total
                from shoes as b;
quit;
```

6. HANDY LINKS

- Fullstimer system option
- Top 10 SQL tricks in SAS
- SAS Help datasets description
- Find your data pattern with PERL
- PERL Regular Expressions cheatsheet
- Using The Boolean operation in PROC SQL

Thank you

Charu shankar

SAS Institute Toronto

Email Charu.shankar@sas.com

Blog https://blogs.sas.com/content/author/charushankar/

Twitter CharuYogaCan

Linkedin https://www.linkedin.com/in/charushankar/

THE POWER TO KNOW.