1.0 ABSTRACT AND OVERVIEW:

A multi-line property-casualty insurance company, such as Hartford Financial Services, not surprisingly has a complex reporting structure. This must focus on such things as: the types of business the company writes; the issuing agents and marketing programs; and general profiles of and geographic locations of the company’s customers. Reporting, even within a specialized area like the Reserving Actuarial area (which I am part of), requires on-demand turn-around, and this reporting takes place at many different levels. There are several dimensions to it, often within a single consideration, such as “line of business”. Support of such summaries at different levels neatly fits the concept of the “Data Mart”, except that -- as we will see -- the natures of the summaries sometimes tend to defy easy classification in terms of clean hierarchies.

The basis of the discussion is a single-point contact facility we call RESMENU, already documented (for the curious) in a previously published NESUG paper. (See the References.) RESMENU permits Reserving area end-users to directly access various kinds of insurance data at various levels of line structure and other considerations. RESMENU data is refreshed -- actually recreated -- each month, and it is the building of this data that is the context of the paper’s discussion.

This is a paper on SAS macros, and it is also a paper on data design and data modeling. The problem that arises in latter areas leads to a particular solution path in the former area. In sections 2.0 and 3.0, the paper lays out a major hurdle in the data development process: complexities in the line structure vis-à-vis the end-users’ reporting needs. Section 4.0 supplements this by expanding the discussion to touch on other dimensions of the data to be provided, in the restricted context of so-called Loss Development Triangles. In Sections 5.0 and 6.0 we define an implementation strategy for the triangle data mart and decide the manner in which we will build the different levels of line structure to meet reporting requirements. Finally, beginning in section 7.0 we lay out the macro solution path. The emphasis is on the creation of basic “plug-in” components. This is an advanced paper, and as such it does not contain a review of or a discussion of syntax. The focus is on the nature of the macro’s built.

2.0 LINE STRUCTURE COMPLEXITIES:

Figure 1.0 (below) depicts a somewhat simplified subset of the line structure problem referred to in the abstract. It depicts a subdivision of some insurance company’s personal automobile business into various considerations. As the table suggests, there are two basic dimensions:

The vertical dimension is the “Marketing Vehicle” or “Segment”, if you like. Examples of the Segment Code on the table are Agency Favoured Customer (Segment 02-01) and Partnership Zebus Auto (Segment 04-01).

The horizontal dimension represents a type of claim or a type of policy “Coverage” against a claim. Auto Bodily Injury No-Fault (Coverage ident N1) and Auto Physical Damage Non-collision (Coverage D2) are examples. Regard-

---

**A PERSONAL AUTO LINE STRUCTURE**

<table>
<thead>
<tr>
<th>LINE</th>
<th>GUARDIAN</th>
<th>POLYSUB</th>
<th>AUTO PRIMARY</th>
<th>AUTOMOBILE LIABILITY</th>
<th>AUTO P.D.</th>
<th>OTHER LIABILITY</th>
<th>EXCESS LIABILITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>LN-08 DESCRIPTION</td>
<td>S</td>
<td>X1</td>
<td>A</td>
<td>B</td>
<td>B1</td>
<td>B2</td>
<td>C0</td>
</tr>
<tr>
<td>01 01-00 Total Voluntary Personal Auto</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
</tr>
<tr>
<td>01-01 Voluntary Programs</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
</tr>
<tr>
<td>01-02 Voluntary Auto Other</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
</tr>
<tr>
<td>02 02-00 Agency Personal Auto - Total</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
</tr>
<tr>
<td>02-01 Agency Favoured Customer</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
</tr>
<tr>
<td>02-02 Agency Normal Rated</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
</tr>
<tr>
<td>02-03 Agency Non-Standard</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
</tr>
<tr>
<td>02-07 Agency Special Pgmms Auto</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
</tr>
<tr>
<td>02-10 Agency Auto ex-Pgmms</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
</tr>
<tr>
<td>03 03-00 Affinity Pers Auto - Total</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
</tr>
<tr>
<td>03-01 Affinity Pers Auto Basic</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
</tr>
<tr>
<td>03-02 Affinity Pers Auto Pgmms</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
</tr>
<tr>
<td>04 04-00 Partnership Pgmms Auto</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
</tr>
<tr>
<td>04-01 Partnership Zebus Auto</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
</tr>
<tr>
<td>04-02 Partnership Fargo Auto</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
</tr>
<tr>
<td>09 09-00 Total Assigned Risk Auto</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
</tr>
<tr>
<td>10 10-00 Total Personal Automobile</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
<td>55</td>
</tr>
</tbody>
</table>
less of the type of personal auto policy or target market, the various Personal Auto policies have these coverages.

Both dimensions of the line structure are depicted in Figure 1.0 at different levels of aggregation. For example, if we add up all the business across Agency, Affinity and Partnership Programs, we have a total voluntary perspective on the company’s business. Thus the “01-xx” series segments are aggregates (or summations) of contributions from the “02-xx”, “03-xx” and “04-xx” sub-lines. Likewise, in the coverage dimension, Total auto physical damage (coverage “D”) consists of the personal property losses due to collision (“coverage” D1) and non-collision (coverage “D2”). The coverage “K1” is for something called Personal Umbrella in the context of Personal Auto. It provides an excess cover above the Auto Liability limits, when one is sued for a large amount.

In RESMENU SAS Data, the Segment is represented by the Segment Code SEGMT and the Coverage Identifier by the Coverage Suffix Code SUF. Together, the two perspectives form the sub-line identifier which we will call the “Reserving Sub-line” or RSRVLN. Each table entry other than the double dots represents an RSRVLN. Examples are:

- 02-01-B1 Agency Favoured Property Damage Liability
- 02-01-C2 Agency Favoured Uninsured Motorist
- 04-01-K1 Zebus Program Personal Umbrella
- 01-01-B Total Voluntary Auto Liability

3.0 THE “RAGGED” NATURE OF LINE STRUCTURES:

Figure 2.0 shows, as a pair of hierarchies the various levels of the marketing perspective (the SEGMT Segment code) and the relationships between levels -- i.e. what is a subset of what. The lowest levels at the right-hand end of the diagrams are the “kernels”. These are shaded and constitute the “n-way” contributions of the marketing dimension.

In the same manner, Figure 3.0 shows, again as a hierarchy diagram, the various levels of the coverage perspective (the Coverage Identifier SUF) and the relationships between levels. The lowest levels at the right-hand end of the diagrams are the “kernels”, and these are also shaded.

It may be apparent at this point where the term “ragged” is coming from. There is no single clean, hierarchical path -- through the line structure and down to the kernel-level contributions -- which will embrace all roll-ups. And the depths, through the hierarchy, of these roll-up-defining paths vary. These two observations are true, both of the coverage perspective and of the marketing perspective.

The table, Figure 1.0, depicts the “kernel” RSRVLN’s by shaded boxes with the XX’s. All other RSRVLN’s are sub-totals of the kernels, and these have ‘ss’ in the spot defining them on the table. There are 267 of these RSRVLN subsets defined on that table, and only 70 of them are kernels.
The remaining 197 are all roll-ups (or consolidations) of the kernels, formed by adding data vertically or horizontally.

4.0 OTHER DIMENSIONS OF THE INSURANCE DATA:

Figure 4.0 is a typical form of loss (claim) data display called the Loss Development Triangle. In it, the data is grouped by accident period (i.e. the period of loss), and the data’s cumulative totals are tracked at regular valuation intervals since the beginning of the accident period -- i.e. the calendar period in which the event causing the insurance claim occurred. In this Loss Triangle exhibit, the accident period

<table>
<thead>
<tr>
<th>Devel</th>
<th>Accident Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>98.5</td>
</tr>
<tr>
<td>2</td>
<td>136.4</td>
</tr>
<tr>
<td>3</td>
<td>153.0</td>
</tr>
<tr>
<td>4</td>
<td>158.8</td>
</tr>
<tr>
<td>5</td>
<td>162.2</td>
</tr>
<tr>
<td>6</td>
<td>162.7</td>
</tr>
<tr>
<td>7</td>
<td>163.8</td>
</tr>
<tr>
<td>8</td>
<td>165.0</td>
</tr>
<tr>
<td>9</td>
<td>165.1</td>
</tr>
<tr>
<td>10</td>
<td>164.9</td>
</tr>
</tbody>
</table>

Our end-users are interested in:
- All levels and combinations of the line structure.
- Accident period triangles based on year, on month and quarter.
- Development intervals that are similarly flexible in their granularity.
- Loss Items, both those on the basic data and those calculated from the basic items. The former include: paid and incurred losses, paid and incurred loss expenses, and the various claim count statistics. Examples of the latter include: the ratio of paid to incurred losses; the average reported claim cost (incurred dollars divided by reported claims); and the total disposed claims (those settled for payment to the claimant otherwise redundant rollups into accident quarter and year.

On the sample triangle display, the specific instances of these dimensions are highlighted using shading.

5.0 TO SUM OR NOT TO SUM -- AND THAT IS THE QUESTION:

In setting up the data mart to support loss triangle requests from end users, we have some decisions to make.

LOSS ITEM LEVEL OF THE DATA: We can platform triangles for all data items the users want, including the calculated items such as the ratios described above. But once the data is formed into cumulative triangles -- which the end-user interface can do at the point the user makes his request -- such computed items can then be easily calculated as matrix arithmetic combinations of the basic variables.

Decision 1: Only the basic “kernel” loss items will be stored in the Data Mart. The access mechanism will form the “computed” items.

ACCIDENT DATE GRANULARITY: If the data is by month, it certainly could be rolled up to quarter or year on the fly. However, if quarterly and yearly data are frequently used, saving the processing time might be worth setting up the otherwise redundant rollups into accident quarter and year.

Decision 2: Based solely on the profile of “most frequent use”, the coarser levels of granularity will be directly supported in the data mart. It
should be noted that this was simply a choice made among options, not the selection of an absolute “right” way.

LINE STRUCTURE: Sections 2.0 and 3.0 have described the nature of complexities involved in this dimension (or pair of dimensions). The table (Figure 1.0) does depict the kinds of perspectives that the users want. There are three ways we can proceed to make available these perspectives:

♦ We can platform the data at the “n-way” level, these being the 70 “kernel” RSRVLN sub-lines and then force the end-user to request whatever kernels needed to form the subsets he/she wants. That is an easy out for the data support area but it does create a burden for the end-user. If the user selection interface is not to be unduly complicated, it would mean that each roll-up sub-line request will probably be a separate request.

♦ We could platform the data at the “n-way” (“Kernel”) level but instead make the reporting interface “intelligent”, by building into it the dynamic calculation of the sub-lines the user asks for. This now allows the user to directly and to easily request data at different levels in a single request. It does pose a considerable burden on the access mechanism, since each roll-up request has to be translated into the components it needs, with the data then being aggregated (using PROC MEANS).

♦ We could platform the data at all levels of the line structure. This forces the complexity issue back on the construction of the data mart. And it means a lot more data -- referring to our table on the first page -- 267 sub-lines worth, rather than the 70 in an “n-way” or kernel level storage. The approach, however, does have an appeal: it renders the access mechanism that serves up the triangle displays a lot simpler. Now the sub-lines requested can be simply extracted, not formed.

The second option is in fact how the iTM1® cube product (a software product of Applix) apparently works. But the scale of TM1’s functioning is well below the scale of the loss triangle data in the real version of our data mart, and the additional overhead in the data access (absent Teradata servers) moves us toward the third option:

Decision 3: Support the full blown line structure in the data mart itself. All sub-lines depicted on the table will be supported in the data. The recreation or update of the data mart will “fill out” the line structure.

And this decision now drives what we talk about next.

6.0 A SOLUTION PATH TO THE FORMATION OF THE DATA MART:

From ‘nway’ level data, there are two ways one can go to form a Data Mart having a line structure such as that depicted in the table in Figure 1.0.

THE TOP-DOWN “RUBBER MALLET” APPROACH: In this solution path, we express each roll-up in terms of its “kernel” summands, extract “Kernels” for each summand needed, and then PROC MEANS the mess by the defined sub-line structure and the keys used for the other dimensions of the data. That in principle can be done -- and it is a viable approach in those instances where there are few roll-ups relative to the number of “kernel” sub-lines.

However, this top-down solution tends to get out of hand very quickly in the present instance. To directly form all the roll-up RSRVLN’s on the line structure table (Figure 1.0) in a single “sort-and-sum” step means forming repetitions of the kernels to the tune of 1,956 sub-line summands. This works out to 7.3 times the size of the final data mart and 28 times the size of the data at the n-way level. Given the other dimensions of the data, with history going back some 25 years (by month in both time dimensions), the relativities must be appreciated in the context of the order of magnitude of the data itself: millions of observations. We are creeping into the realm of “Impossibly Large Datasets”, to use a term coined by SUGI Author Mike Raithel in a presentation a few years back.

BOTTOM UP - A STAGED FORMATION OF THE SUB-TOTALS: This alternative approach to the consolidations makes use of the hierarchy diagrams (Figures 2.0 and 3.0). From the standpoint of the market segments, regardless as to what coverage is involved, what segments are immediate summands of what other segments? From the standpoint of the coverage codes, what coverages are immediate summands of what other coverages? Instead of doing the job in one pass with an “Impossibly Large Dataset”, we break the task into pieces. We form some subtotals, and then use those in turn to form subtotals at a higher level.

We can translate the “Bottom Up” Process, just described, in terms of “Roll-up Mappings”. The table (Figure 5.0) at the bottom of this page displays just such as set of mappings for the Marketing Perspective (SEGMT code) of the Line structure. At the top of the next page is a similar table (Figure 6.0) defining “Roll-Up Mappings” for the Coverage Identifier perspective (SUF code).

We can handle the consolidation process independently in the Marketing and Coverage perspectives -- except for one twist. There might be a benefit in forming the roll-ups in the Marketing Perspective first. If we do this, we can in the coverage dimension selectively censor (i.e. not form) consolidations that are just not useful. In Personal Auto, the only relevant “Other Liability” coverage is what is called Personal Umbrella, which shows up as Coverage ident K1. The sum of the K-coverages is Total (non-excess) General Liability sub-total, represented by coverage ident “K”. For

<table>
<thead>
<tr>
<th>FORMING THE DATA MART</th>
<th>STAGED ROLL-UPS IN THE SEGMENT DIMENSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUB-LINE</td>
<td>LEVEL</td>
</tr>
<tr>
<td>01 00</td>
<td></td>
</tr>
<tr>
<td>01 01</td>
<td></td>
</tr>
<tr>
<td>01 02</td>
<td></td>
</tr>
<tr>
<td>02 02</td>
<td></td>
</tr>
<tr>
<td>02 03</td>
<td></td>
</tr>
<tr>
<td>02 04</td>
<td></td>
</tr>
<tr>
<td>03 02</td>
<td></td>
</tr>
<tr>
<td>03 03</td>
<td></td>
</tr>
<tr>
<td>04 00</td>
<td></td>
</tr>
<tr>
<td>04 01</td>
<td></td>
</tr>
<tr>
<td>04 02</td>
<td></td>
</tr>
<tr>
<td>05 00</td>
<td></td>
</tr>
<tr>
<td>10 00</td>
<td></td>
</tr>
</tbody>
</table>

[fig. 5.0]
Personal Auto, it would have the same contents as “K1”. Likewise, Total Other Liability, coverage “R” in the table, is the sum of “K” plus Commercial Excess/Umbrella (commercial), which has ident “L”. Again there is no “L” coverage in Personal Auto so that “R” is the same as “K” which is the same as “K1”. We really only need “K1” for the Personal Auto Data.

(As an aside, please remember, for purposes of this discussion, that you are looking at only Personal Automobile insurance. Actual RESMENU data embraces many other areas of business such as Commercial General Liability, which does fill out the K-coverages).

So, the solution path to building the data roll-ups or consolidations will be in two stages. The first stage will work in the vertical dimension of the table in Figure 1.0; the second stage will work in the horizontal dimension:

One final thing you may have noticed, if you compared the two tables on this and the preceding page with the corresponding hierarchy diagrams (Figures 2.0 and 3.0). The Roll-up Mapping tables actually collapse the hierarchies slightly. For example, instead of summing Segments 02-01 thru 02-03 to 02-10 and then adding Segments 02-10 and 02-07 to get 02-00, the rollup process will create 02-00 and 02-10 simultaneously in the same step. This was simply a judgmental shortcut designed to cut down on the number of stages in the summation process.

7.0 THE SEGMENT CONSOLIDATION PROCESS

For purposes of the paper, we now focus our attention on the Marketing Perspective or Segment Dimension. Consider the Segment table shown as Figure 5.0. From that exhibit, we can see a mapping which assigns each Segment to its string of immediate (“next”) roll-ups, and this mapping will specify that:

02-01 rolls into 02-00 and 02-10 and 01-02
02-02 rolls into 02-00 and 02-10 and 01-02
02-03 rolls into 02-00 and 02-10 and 01-02
02-07 rolls into 02-00 and 01-01
03-01 rolls into 03-00 and 01-02

The Segment Dimension has a wrinkle that the Coverage dimension does not have -- a precise two-level hierarchy within its internal structure. This we are going to take advantage of in staging the consolidation process. The first two digits of the Segment Code constitute values of a "LINE". For our Personal Auto data, these LINE's are apparent from Figure 1.0.

01 - Total Voluntary
02 - Agency
03 - Affinity
04 - Partnerships
09 - Assigned Risk
10 - Total, All Personal Auto

As we form the Segment consolidations, we will start at the lowest “level” and fill out the LINE's that exist at that “level”. Then we will generate the data for Segments at the next “level”. These, like Segments 01-01 and 01-02, may in turn be components of other Segments in the same LINE. Indeed that is the case for 01-01 and 01-02 -- the sum of these is Segment 01-00. So we fill out the sub-totals at this next “level”. And so on up the hierarchy, we fill out LINE's at a given “level” and then generate data for LINE's at the next “level”. Clearly for this to work, we should have set up our LINE's so the data within a LINE is at the same “level”.

8.0 REPRESENTING THE SEGMENT CONSOLIDATION MAPPINGS AS SAS FORMATS

For processing purposes, we now represent the two mappings described above as SAS Value Replacement Formats derived from the table depicted as Figure 5.0. Suppose the SAS Name of that table is TABLELIB.TABLEM. Suppose that SEGMT is the Segment Code, LVL is the Level indicator, and ROLLUP1, ROLLUP2 and ROLLUP3 are the values of the mappings. Then the two SAS Formats we will need (and will use in Section 10.0) are:

♦ SPASUML, which will map Segment to the sequence of Roll-up Segments ROLLUP1-ROLLUP3; and
♦ SEGLVL, which will map each Segment to its Roll-up Hierarchy level LVL.
These formats can be created using the following SAS code:

```sas
DATA CNTLFILE (Keep = FMTNAME START LABEL);
Length LABEL $ 40;
Array RULINES(*) ROLLUP1-ROLLUP3;
*.......................................;
Set TABLELIB.TABLEM;
START = SEGMT;
*.......................................;
FMTNAME = "SEGLVL";
LABEL = left(put(LVL, 2.));
Output;
*.......................................;
FMTNAME = "SPASUMD";
LABEL = "*";
Do JJ = 1 to Dim(RULINES);
  If RULINES(JJ) ne "...." Then Do;
    TEMP = Input(RULINES,4.);
    If Floor(SEGMT/100) = Floor(TEMP/100)
    Then LABEL = Left(Trim(LABEL) || "  
      ||RULINES;
  End;
  End;
  If LABEL ne ":*" Then Output;
Run;

PROC SORT Data=CNTLFILE;
  By FMTNAME START;
Run;
PROC FORMAT Cntlin=CNTLFILE Library=FMTLIB;
Run;

Note that SAS Format SPASUM lists the blank-separated values after an asterisk. When the format is used later, the presence or absence of the asterisk will make it easy to see whether or not rollups do exist for the Segment (SEGMT) in question.

9.0 ADDING TO THE TOOL SET -- "SORT-AND-SUM"
MACRO SB0010MS:
Each stage of the data mart formation process involves defining summands for the consolidation and then forming the consolidation. This is nothing more than sort-and-sum, except for one wrinkle: In some instances, the data fed into the sort step may blow out the SAS work-space. So the "sort" step may have break the input file into pieces, sort the pieces and then remerge the data. And a truly useful utility should be intelligent enough to know when to do that.

Here is such a tool, macro SB0010MS:

```sas
/* **************************************** */
/* SB0010MS :  Sort and Summarise .......... */
/* **************************************** */
%macro SB0010MS( infile =,
  outfile =,
  domeans = YES,
  statistic = SUM,
  weight =,
  byvar =,
  idvar =,
  srtsize = 1000000,
  sumvar = ,
  where = );

%put ****************************************;
%put - SB0010MS - Sort Step Begins ..........;
%put ****************************************;
%if &domeans.=YES %then %let sortout=SB0010D1;
%else %let sortout=&outfile;
%put - Determine File Size.;
%let dsid=%sysfunc(open(&infile.));
%if &dsid. lt 1 %then %do;
  %put *** File does not Exist ***;
  %goto endprog;
%end;
%let numrecs=%sysfunc(ATTRN(&dsid.,NOBS));
%let rc=%sysfunc(close(&dsid));
%put - Input Data File ....: &infile.;
%put - Number Observations.: &numrecs.;

%let leftover=&numrecs;
%let numfiles=0;
%do %until(&leftover. le 50000);
  %let numfiles=%eval(&numfiles.+1);
  %let leftover=%eval(&leftover.-&srtsize.);
%end;
%put - Number Recs per Sort.: &srtsize.;
%put - Number Brkout Files.: &numfiles.;
%if &numfiles=1 %then %do;
  %put ****************************************;
  %put - SMALL FILE - Single Step Sort ....;
  %put ****************************************;
  PROC SORT DATA=& infile.
    OUT=& sortout. ;
    BY &byvar.;
    %if &where. ne %then %do;
      WHERE &where.;
    %end;
    RUN;
%end;
%else %do;
  %put ****************************************;
  %put -- LARGE FILE: Sort Using &numfiles Files;
  %put ****************************************;
  PROC SORT DATA=& infile.
    OUT=& sortout. ;
    BY & byvar. ;
    %if & where. ne %then %do;
      WHERE &where.;
    %end;
    RUN;
%end;
%end;
%else %do;
  %put ****************************************;
  %put -- LARGE FILE: Sort Using &numfiles Files;
  %put ****************************************;
  %put -- LARGE FILE Breakout Step;
  DATA %do mm = 1 %to &numfiles.;
    PART:mm
    %end;
  %put ****************************************;
  %put - Number Brkout Files.: &numfiles.;
  %if &numfiles=1 %then %do;
    %put ****************************************;
    %put - SMALL FILE - Single Step Sort ....;
  %else %do;
    %put ****************************************;
    %put -- LARGE FILE: Sort Using &numfiles Files;
  %end;
  %put -- LARGE FILE Breakout Step;
  DATA %do mm = 1 %to &numfiles.;
    PART:mm
    %end;
  %put ****************************************;
  %put - Number Brkout Files.: &numfiles.;
%end;

%end;
```
The macro SB0010MS is intended not only for straight summations but also for consolidations where averages or weighted averages are required, hence the two parameters STATICTIC and WEIGHT. In Reserving Applications, consolidations of average claim costs and loss development time curves (percents) are two applications where SB0010MS is used in this way. Also, SB0010MS can serve as a pure sort utility only, not just in its capacity as a sort-summation utility -- see parameter DOMEANS. Sort Work space is an issue with large SAS Datasets, and parameter SRTSIZE specifies the maximum number of observations permitted in a file to be sorted. There is a 50,000 record fuzz amount so that breakouts do not occur when the number of observations is only slightly over the limit.

Within the sort step, using %SYSFUNC the macro SB0010MS opens the source file (parameter INFILE) and determines the number of observations in that source file, NUMRECS. Based on that file size and the SRTSIZE parameter, the macro calculates the number of Breakout files NUMFILES needed to accomplish the sort. If only one is required, a direct sort of the input takes place. If more than one is required, a DATA step breaks up the source file (parameter INFILE again) using a SELECT group involving iteratively generated WHEN clauses. Each component file is sorted in a set of iteratively generated PROC SORT executions. Then, using the sort criteria, a second DATA-step merges the sorted component files (SET...BY processing).

The sort-break criteria is spelled out by parameter BYVAR.

When SB0010MS is used only as a sort utility (DOMEANS=NO), the output file (parameter OUTFILE) is the output of the PROC SORT ("small file") or the DATA-step merge ("Big file"). Otherwise the output is a temporary file, which now feeds the PROC MEANS summation step. The summation is a strict use of sort-break processing -- using BY (rather than CLASS) criteria. Again, the BY-criteria are specified by parameter BYVAR. The variables to be summed are specified in parameter SUMVAR. Often there are other variables along for the ride which we want to keep. In the RESMENU data, for example, there are separate codes, SEGMT and SUF ("suffix") for respectively the marketing and coverage components of the RSRVLN sub-line designator. These, along with perhaps other data elements, often make up an IDVAR parameter list. Specifying these as ID variables in the PROC MEANS execution ensures that the variables will be passed along to the output file, which is specified by the parameter OUTFILE.

When a Weighted Average is calculated -- such as in the formation of average time curves -- the "weight" variable is specified in parameter WEIGHT. In that instance, the sum of the values of the WEIGHT variable are also calculated and saved as part of the PROC MEANS output - see the use of the SUMWGT= parameter in the PROC MEANS step.

We offer some final observations on SB0010MS. It allows for optional built-in rudimentary data filtration (exclusion) in the sort step. Using parameter WHERE one can set up a simple sub-setting WHERE clause. SB0010MS cleans house; at the end of the execution, it kills off the temporary files it has had to create. And finally, it offers a running commentary on what it is doing (%put's). Because of this, it is normally run with NOTES and MPRINT shut off.
The use of **SB0010MS** is ubiquitous in RESMENU applications. In this capacity it saves much repetitive coding of SAS PROC SORT, DATA-Step and PROC MEANS executions.

### 10.0 BUILDING SEGMENT LEVEL ROLL-UPS -- THE CONSOLIDATION MACRO SB0019MS:

This job falls to another generic macro, **SB0019MS**. Its code is below with a discussion of the code following:

```sas
/* *********************************************/
/* SB0019MS : Make Segment Consolidations ... */
/* *********************************************/
%macro SB0019MS( infile =, outfile =, othrkeys =, byvar =, idvar =, sumvar =, statistic = SUM, weight =, level =, within =);
  %local notes mprint run linemap;
  %local ninp nout dsid rc;
  %put *********************************************;
  %put SB0019MS-1.0  EXTRACT THE DATA ...........;
  %put *********************************************;
  options &notes &mprint;
  %SB0010MS( infile = SB0019D1,
    options nonotes nomprint;
  %put *********************************************;
  %put SB0019MS-2.0 FORM THE SUMMARIES ...........;
  %put *********************************************;
  Run;
  %put *********************************************;
%mend SB0019MS;
```

From a cursory comparison of the `%MACRO` statements, you may pick up on some similarities in the set-ups of **SB0019MS** and our generic sort-sum utility **SB0010MS**. The nature of the summation statistic is flexible and so are the lists constituting the analysis variables, the sort-break By-variables, and the ID-variables. One difference in **SB0019MS** is the special position occupied by the **RSRVLN** sub-line identifier in the BY-list, hence the two separate by-lists for **SB0019MS** -- the first **BYVAR** lists the keys following **RSRVLN** in sort order; and the second list **OTHRKEYS** specifies those key variables preceding **RSRVLN** in the sort order. In some applications, “Risk State” (Policyholder domicile) might be in **OTHRKEYS**.

One very important thing to note about **SB0019MS** is its invocation of sort-and-sum utility **SB0010MS**. **SB0010MS** however, has a more focused purpose in life: to fill out the Segment Values in the sub-line structure. As noted earlier, in forming the Segment roll-ups, we will start at the lowest level and fill out the **LINE**’s that exist at that level. Then we will generate the data for Segments at the
next level. These, like Segments 01-01 and 01-02 may be components of another LINE for which there are subtotals, in this case Segment 01-00. So we fill out the sub-totals at that level. This translates into the following logic in SB0019MS:

1) The LEVEL parameter on the macro specifies the LVL value for the data that will be taken as candidates for summands to the consolidation or roll-up process. The WITHIN parameter of SB0019MS -- which has a values “YES” or “NO” -- drives whether we are filling out a LINE or moving up the hierarchy into the next value of LVL.

2) In the macro, variable RULEVEL holds the current level from which summands are to be taken -- it has as its value the value of the macro parameter LEVEL.

3) OUTSGLVL has the level of the target consolidations. If we are doing rollups within a line, it has the same value as RULEVEL. Otherwise it is the next level up -- i.e. RULEVEL + 1.

4) Data will be selected for possible contribution to the summation process if and only if the Segment has RULEVEL as its LVL value. The level of the Segment is determined using format SEGGLVL, which was derived from table TABLELIB.TABLEM (Figure 5.0).

5) Any rollups that such Segments define will be determined by the list arrived at by using format SPASUML, again from table TABLELIB.TABLEM. The returned string has valid roll-up Segment values if and only if the first character is an asterisk.

6) These candidates will be processed in the application of SB0019MS if and only if the Roll-up Segment has as its value OUTSGLVL. Determining this level is another application of Format SEGGLVL.

The final step in the macro SB0019MS merges the calculated roll-up data with the source data fed to it.

11.0 BUILDING COVERAGE LEVEL ROLL-UPS - THE CONSOLIDATION MACRO SB0018MS:

After the segment consolidations have been filled out, the second stage will fill out the coverage roll-ups. This works in the horizontal dimension of the table shown as Figure 6.0:

♦ Form the “Level 1” Coverage Rollups from the “Level 0” Coverage Data.

♦ Form the “Level 2” Coverage Rollups from the “Level 1” and Level 0 Coverage Data as specified.

♦ Form the “Level 3” Coverage Rollups from the “Level 2” and lower level Coverage Data as specified.

To do this, macro SB0018MS is set up to carry out the steps of this logic. It inputs data at one coverage level and fills out the coverage consolidations at the next level. Unfortunately, space does not permit laying out the details -- i.e. the SAS Code -- of this member of the tool-set. Suffice to say, SB0018MS is similar in flow to SB0019MS and will use a pair mapping formats called $SUFLVL and $SPXSUML, these being very similar in concept respectively to formats SEGGLVL and SPASUML, but with Coverage ident SUF as the argument. These formats will be generated from the SAS Dataset version of the table shown as Figure 6.0.

For the coverage consolidations, a processing difference is that the roll-up process will NOT require the WITHIN parameter. It will still make use of the LEVEL parameter, however. SB0018MS will take Coverage Suffix Values at a given level, create summand data for consolidations at the next level, and form the roll-ups using SB0010MS.

12.0 ASSEMBLING THE TOOL SET AND MAKING THE DATA MART-- MACRO SB0005PC

We now package, into generic macro SB0005PC below, the use of SB0018MS and SB0019MS. SB0005PC will now build the line of business consolidations starting with the “nway” kernel-level data.

/* ************************************************* */
/* SB0005PC : Line-Structure Consolidations . */
/* ************************************************* */

%macro SB0005PC(  infile =, 
    outfile =, 
    otherkeys =, 
    byvar =, 
    idvar =, 
    sumvar =, 
    statistic =, 
    weight =, 
    nsegvls =, 
    ncvovls =);

%local notes mprint run nextlvl;
%local ninp nout dsid rc;
%put **************************************;
%put -------- Begin SB0005PC **************;
%put **************************************;
%let statistic=%upcase(&statistic.);
%if &statistic ne MEAN %then %let weight=;
%put **************************************;
%put SB0005PC-1.0 SEGMENT CONSOLIDATIONS ......;
%put **************************************;
%let nn=%eval(2*(&nseglvls.-1));
%let level=0;
%let srcefile=&infile;
%do jj = 1 to &nn;
%if &jj=%eval((&jj/2)*2) %then %do;
%let within=YES;
%let nextlvl = &level.+1;
%end;
%else %do;
%let within=NO;
%let level = &level.+1;
%let nextlvl = &level.+1;
%end;
%let nextfile=SB0019&jj.;
%put *****************************************************;
%put SB0019MS(  infile = &srcefile.,
    outfile = &nextfile.,
    otherkeys = &otherkeys.,
    byvar = &byvar.,
    idvar = &idvar.,
    sumvar = &sumvar.,
    statistic = &statistic.,
    weight = &weight.,
    level = &level.,
    within = &within.);
%let srcefile=&nextfile;
%end;
A sample execution of **SB0005PC** to build loss triangle data in the line structure depicted schematically in the table shown as Figure 1.0 might look like the following:

```sas
%put ******************************************;
%put SB0005PC-2.0 COVERAGE CONSOLIDATIONS....;
%put ******************************************;
%let mm=eval(&nn.+1&ncovlvls.);
%let level=0;
%do jj = 1 %to &mm;
%let nextlvl = eval(&level.+1);
%if jj lt &mm
  %then %let nextfile=SB0019&jj.;
%else %let nextfile=&outfile.;
%put ******************************************;
%put SB0005PC-2.&jj. Rollups from Level &level;
%put ******** to Level &nextfile. ....;
%put ******************************************;
%let level=0;
%do jj = &nn.+1 %to &mm;
%let level=0;
%put ******************************************;
%let srcefile=&nextfile.;
%put SB0005PC-2.0  COVERAGE CONSOLIDATIONS.....;

A sample execution of **SB0005PC** to build loss triangle data in the line structure depicted schematically in the table shown as Figure 1.0 might look like the following:

**SB0005PC**

```sas
 infile = MYLIB.PAUTOSRC,
  outfile = MYLIB.PAUTODM,
  othkeys = RSTATE,
  byvar = ACCDATE PRODATE CLMTYPE,
  idvar = ACCYEAR,
  sumvar = INCDIND PAIDIND PAIDALE
 CRTDCLMS SETLCLMS,
  statistic = SUM,
  weight =, nseglvls = 3, ncovlvls = 3);
```

where the parameter values translate as follows:

- **MYLIB.PAUTOSRC** is source data at the "nway" level.
- **MYLIB.PAUTODM** is the data mart of consolidations.
- **RSTATE** is an additional key for “Risk State”.
- **CLMTYPE** is another key, for “Claim Category”.
- **ACCDATE** is the Accident Date (where the granularity is by month).
- **PRODATE** is the Calendar Date of the activity (where the granularity is also by month).
- **INCDIND, PAIDIND, CRTDCLMS, etc.** are the claim variates. Read the list as “Incurred Losses”, “Paid Losses”, “Paid Loss Expenses”, “Created (Reported) Claim Counts” and “Settled Claim Counts”.

What **SB0005PC** does is use first **SB0018MS** and then **SB0019MS** over and over again until the consolidations are built up at all levels of first the Marketing Perspective (Segment **SEGMT**) and then the Coverage Perspective (Coverage Suffix code **SUF**) to parameterise names of the SAS Formats used by **SB0019MS** and by **SB0018MS**. This modification would permit the same software to deal with multiple line structures of the same form but having different granularities.

After forming the above line of business consolidations at the accident month level, one can use utility **SB0010MS** to effect further roll-ups of the data to Accident Quarter or to Accident Year, thus completing the Data Mart as specified in Section 5.0.

### 13.0 Concluding Remarks:

What you have seen are simplified versions of key software underlying our RESMENU data mart formation process. The use of these SAS macros is ubiquitous in the RESMENU data production cycles. Setting up a single set of macros such as these means there is one-stop shopping when it comes to maintenance on the processing software.

A lesson that can be culled from an experience such as this is that it often pays to break down a complex process into pieces, not only for reasons of data volume, but also from the point of view of process simplification. Breaking down something into steps will often reveal that the same activity is occurring over and over again. And if the activity is the “same” in some sense, then perhaps the pieces of software performing the processing should also be the same. There is something to be said for interchangeable parts.

**Acknowledgments and References**

SAS is a Registered Trademark of the SAS Institute, Inc. of Cary, North Carolina.


iTM1 is a registered trademark of Applix, Inc.

The formation of Loss Development Triangles from calendar insurance claim data is described in the last section of the paper **Uses and Handy Abuses of Proc Transpose**, published elsewhere in these Proceedings as well as in those for NESUG 2000.

A general discussion of RESMENU and its various reporting and production dimensions is in the paper **Housekeeping Revisited: Managing the Applications Environment** in the NESUG 2000 Proceedings.

The author wishes to thank Cathy Agogliati, Rob Russell and William Spigener for reviewing the paper.

Ralph may be contacted at rleighton@thehartford.com.