Paper 162-2010

The Quest for the Holy Grail: OLAP Approaches for Calculating Actuarial Measures

Heather Seeno, Highmark Inc., Camp Hill, PA
Alan Leach, Qualex Consulting, Cincinnati, OH

ABSTRACT

At Highmark Inc. Online Analytical Processing (OLAP) is used to provide access to large amounts of data stored in Highmark's data storage system, the "Enterprise Data Warehouse." A common actuarial metric is the calculation of Per Member Per Month cost, or PMPM. The numerator in this ratio is total claims cost in dollars. The denominator is total members enrolled, including members who have no claim costs and are not represented in the numerator. This metric is a challenge in standard OLAP reporting tools because the numerator and denominator represent different populations at different levels of summation. If a user wants to look at the data across various levels of claims dimensions, such as CLAIM TYPE, the numerator will change depending on whether the user is looking at Inpatient, Outpatient, Professional or Drug claims. However, the denominator, the total population of members remains constant across these levels. The purpose of this paper is to explore the development steps required to generate PMPM using SAS® OLAP Server and Futrix®.

DISCLAIMER

The study is not intended to be an exhaustive study of the subject matter. This presentation is a general study done by Highmark to better understand OLAP. It is not intended to provide instructions on how a company may analyze and use its data or how a company would use the software and other materials referenced in the presentation. This presentation is a summary of a study performed by Highmark and was done independent of the individuals and companies referenced in the presentation. The study does not represent an endorsement of any of the software and other materials referenced in the presentation nor is the study an endorsement of the individuals or companies referenced in the presentation.

INTRODUCTION

In this paper we discuss our quest to provide users in the Actuarial department at Highmark the Per Member Per Month (PMPM) cost. We will discuss two solutions that we developed. Our first solution uses Futrix®, a product of Futrix Limited, and leverages its customization capabilities. The second solution utilizes SAS® OLAP Server technology. We'll begin with a brief review of OLAP terminology, followed by a discussion of the problem of calculating PMPM within OLAP. This will lay the groundwork for discussing cubes we developed under Futrix® and SAS® V9.2.

BACKGROUND

WHAT IS OLAP?

Online Analytical Processing (OLAP) is a technology that provides end users with access to large amounts of data in an intuitive and rapid manner. It enables users to analyze measures across different views of multidimensional data. Highmark Inc. is a health insurer covering over 4 million members in Pennsylvania and West Virginia. At Highmark, OLAP is one of an arsenal of tools available to analysts, providing quick access to large amounts of data stored in the Enterprise Data Warehouse. The Actuarial department at Highmark maintains a number of OLAP Cubes, centered on various insurance products.

What is OLAP? OLAP is the process of aggregating calculations in advance and surfacing those calculations in a drillable interface. In 2009 Rupinder Dhillon and Harry Droogendyk likened OLAP to "A Pivot Table on Steroids." Pivot tables are summarized values of detail data. Similarly, OLAP cubes are summarized values of detail data, but the underlying data can be much, much larger than what a spreadsheet can handle.

The source data structures from which an OLAP cube is built can vary. ROLAP (Relational OLAP) uses relational tables that are joined appropriately based on the users' query. MOLAP (Multidimensional OLAP) relies on pre-computation and storage of information in the cube. HOLAP (Hybrid OLAP) uses a combination of ROLAP and MOLAP. The SAS® OLAP Server supports all of these types of cubes. In SAS® OLAP server, data drill paths are predetermined based on the anticipated approach an analyst might take to addressing business problems.

OLAP TOOLS AT HIGHMARK

There are several tools in the SAS® Enterprise Intelligence Platform licensed by Highmark to develop and surface OLAP cubes. To develop cubes in SAS® we employ SAS® OLAP Cube Studio and the cube designer wizard included with SAS® Data Integration Studio. These tools assist the developer in defining the structure of the cube. The developer defines dimensions and hierarchies, or drill paths, based on the anticipated approach an analyst might take to addressing business problems. The analysis variables or measures are also defined. OLAP Cube Studio can optionally generate the OLAP procedure statements used to build the cube. To surface cubes to users there are several tools available – SAS® Web Report Studio, SAS® Enterprise Guide, and SAS® Add-in for Microsoft Office. Each of these allows the user to analyze the data by examining tuples, or slices, of the cube quickly and easily.

In addition to the SAS tools mentioned above, Highmark also uses Futrix®, an Enterprise Business Intelligence suite written in SAS® and licensed through Futrix® Limited of Wellington, New Zealand. It is a pre-packaged solution that provides business users with the information they need to make more effective decisions. It is a compliment to SAS® Enterprise Intelligence Platform and can optionally integrate into the SAS® Enterprise Intelligence Platform environment of your site. Like SAS® Business Intelligence, Futrix® offers a set of tools for designing and building cubes as well a function rich end user interface for analyzing cube data. Futrix® can build cubes using a proprietary structure based on indexed summary tables as well as with PROC OLAP. In fact, if you use Futrix® to develop cubes using PROC OLAP you can push the cubes to your SAS® Business Intelligence environment and surface them in any of the end user reporting tools mentioned above. However, cubes built in SAS® OLAP Cube Studio cannot be surfaced in the Futrix® end user interface. SAS® cubes do not support the Futrix® “Drill Anywhere” capability. For our solution we use Futrix® V6.1.

OLAP TERMINOLOGY

In addition to the technological differences mentioned above, SAS® Business Intelligence and Futrix® use slightly different terms when describing cube structure. We will discuss this briefly to help lay the ground work for later discussions of our solutions. SAS® Business Intelligence tools, such as SAS® OLAP Cube Studio, use the conventional OLAP terms. In conventional OLAP terms the order in which information may be retrieved, from the highest summary level down to the detailed data, is specified by a hierarchy. Dimensions are collections of related hierarchies. For example, a Time dimension might have several hierarchies, such as “Year->Month->Day”, “Year->Quarter->Day” and “Year->Week->Day”. The variables that combine to create a hierarchy are known as levels. In the example Year, Month, Quarter, Week and Day are all levels.

Futrix® uses slightly different terms for these objects. In designing cubes the developer does not define specific drill paths or hierarchies. This permits users to drill anywhere. In Futrix® any single grouping variable is referred to as a dimension. These are organized logically, not functionally, into dimension groups. For example, in Futrix® the dimensions of Year, Month, Quarter, Week and Day might be organized under a dimension group called Time. Dimensions may or may not be hierarchically related inside dimension groups. For example, in a dimension group called Demographics one might find the dimensions of Race, Age, and Gender. This would not represent a particular drill path just a logical collection of dimensions.

The table below summarizes the terminology differences between these two OLAP purveyors:

Example	SAS® OLAP Server	Futrix®
Time	Dimension	Dimension Group
Year/Month/Day	Hierarchy	Not Applicable
Year	Level	Dimension
Month	Level	Dimension
Day	Level	Dimension

Table 1. OLAP Terminology

THE CHALLENGE: CALCULATING PMPM

Typically we have two major components to our data - a claims dataset that contains claims filed by members and an enrollment dataset containing all members who were eligible to file claims. For example,

MONTH	INSURANCE PRODUCT	NUMBER OF MEMBERS ENROLLED
Jan 2009	HDHPPO	100
Jan 2009	DRGBLU	200
Feb 2009	HDHPPO	300
Feb 2009	DRGBLU	150

Table 2. Sample Enrollment Data

CLAIM NUMBER	MONTH	INSURANCE PRODUCT	DRUG NAME	PAID AMOUNT
001	Jan 2009	HDHPPO	DRUG A	$500
002	Jan 2009	HDHPPO	DRUG A	$200
003	Jan 2009	HDHPPO	DRUG B	$100
004	Jan 2009	DRGBLU	DRUG B	$200
005	Feb 2009	HDHPPO	DRUG B	$1000
006	Feb 2009	HDHPPO	DRUG A	$300

Table 3. Sample Claims Data

CLAIMS and ENROLLMENT have dimensions in common – MONTH and INSURANCE PRODUCT. However, CLAIMS has additional information about the claims – DRUG NAME and PAID AMOUNT.

We can join them and we would get:

MONTH	INSURANCE PRODUCT	DRUG NAME	PAID AMOUNT	NUMBER OF MEMBERS ENROLLED
Jan 2009	HDHPPO	DRUG A	$500	100
Jan 2009	HDHPPO	DRUG A	$200	100
Jan 2009	HDHPPO	DRUG B	$100	100
Jan 2009	DRGBLU	DRUG B	$200	200
Feb 2009	HDHPPO	DRUG B	$1000	300
Feb 2009	HDHPPO	DRUG A	$300	300
Feb 2009	DRGBLU			150

Table 4. Combined Data

The metric we are interested in is per member per month calculation (PMPM) which is the total amount paid divided by the total number of members in that slice of data.

For example, the PMPM for Jan 2009 for HDHPPO, DRUG A is $500/100 = $5.

At issue is when we try and calculate that measure across DRUG NAME. For example if want to calculate PMPM for Jan 2009, HDHPPO the correct calculation is

($500 + $200 + $100) / 100 = $8.

However, using standard aggregations the calculation will come out as

($500 + $200 +$100) / (100 + 100 + 100) = $2.60

In OLAP terminology, MEMBERS is a *semi-additive* measure. It is additive across some dimensions, such as MONTH and INSURANCE PRODUCT, but not additive across other dimensions, such as DRUG NAME. We need a way to stop the MEMBERS from summing across the claims dimensions, but instead honor the unique combination of enrollment dimensions.

In standard static reporting this challenge is easily overcome by summarizing claims and enrollment separately and then joining to the results based on the specific report of interest. Indeed, our actuaries have done this previously. In the dynamic world of OLAP, where users can create ad-hoc reports summarizing by claim specific columns and / or member specific columns on the fly – delivering PMPM has become our quest.

THE DATA

To illustrate our solutions in the paper we generated sample claims and membership data. We began with the universe of Highmark contract holders with drug coverage in 2009 and randomly selected a subset of 50,000. For each contract holder we captured the region, product, client and group under which the contract holder is insured. We then acquired the pharmacy claims incurred in 2009 for that subset of contract holders. For each claim we extracted the dollar amount Highmark paid out as well as key information about the claim such as the date the claim was incurred and the drug name. Drug claims are classified by AHFS Pharmacologic-Therapeutic Classification©, the categories established in the American Hospital Formulary Service published by the American Society of Health-System Pharmacists®. Throughout this paper we refer to the AHFS Pharmacologic-Therapeutic Classification© as Therapeutic Class.

The claims paid amounts included in this paper are for illustrative purposes only and do not reflect any real Highmark data. All claims dollars have been modified using randomly assigned algorithms.

APPROACH #1: FUTRIX® CUSTOM CODE

The Futrix® approach requires two separate meta-items – one for claims and one for enrollment. (In Futrix® a meta-item is a registered data source. It is akin to a table registration in SAS® Business Intelligence but also has elements of SAS® Information Maps.) While the user is using the Futrix® interface to drill into the claims cube, there is custom code working behinds the scenes that will query the enrollment cube based on whatever enrollment specific dimensions or filtering the user has selected in the claims cube. The code will then take the results of the enrollment query and join it back to the results of the claims query the user just performed. Futrix® then displays these combined results back to the user.

To set up our meta-data we start with a claims base table and an enrollment base table and register them as separate meta-items in Futrix®. We align the common dimensions between the two meta-items.

DIMENSION GROUP	DIMENSION	Claims Meta-item	Members Meta-item
Time			
	Year	YEAR	YEAR
	Quarter	QTR	QTR
	Month	MONTH	MONTH
Formulary			
	Therapeutic Class	THERAPEUTIC_CATEGORY_1	
	Formulary 2	THERAPEUTIC_CATEGORY_2	
	Formulary 3	THERAPEUTIC_CATEGORY_3	
	Drug	DRUG	
Product			
	Region	REGION	REGION
	Product	PRODUCT	PRODUCT
	Client	CLIENT	CLIENT
	Group	GROUP	GROUP
Member			
	Member Gender	GENDER	GENDER
	Member Relation Code	RELATION	RELATION
	Member Zip Code	MEMBER_ZIP_CODE	MEMBER_ZIP_CODE
Pharmacy			
	Pharmacy State	PHARMACY_STATE	
	Pharmacy Chain	PHARMACY_CHAIN	
	Pharmacy	PHARMACY	
Prescriber			
	Prescriber ID	PRESCRIBER_ID	
Enrollment			
	Plan Area	PLAN_AREA	PLAN_AREA
	Drug Type		
Drug Type			
	Mail Order or Retail	MAIL_RETAIL	
	Brand or Generic	BRAND_GENERIC	

Table 5. Futrix® Metadata Matrix Dimension View

The alignment of these dimensions is critical to this solution. Doing this allows us to leverage meta-data within Futrix® to generate the code to query the enrollment meta-item. Furthermore, to simplify the code, we require that common columns between claims and enrollment have the same name. For example, in the above meta-data matrix you can see that the Member Zip Code dimension is sourced from the same column in both claims and enrollment – MEMBER_ZIP_CODE.

For the MEMBERS measure for CLAIMS we use a custom measure in Futrix®. Custom measures are measures that do not exist in the base table. Instead, it is the developer's responsibility to supply these values via his/her own code which is executed during report time. Custom measures are used when there is a requirement for nonstandard calculations.

In our case, there is no MEMBERS source variable in CLAIMS. Instead, our custom code queries the enrollment meta-item at report time to return the appropriate membership value. This is supplied as the value for the custom measure in the claims. Total Paid Amount is a source measure. Total Paid Amount and Total Members are used to calculate Paid PMPM.

Description	Type	Claims Meta-item	Members Meta-Item
Total Paid Amount	Source	PAID_AMOUNT	
Total Members	Custom	MEMBERS	
Paid PMPM	Calculated	Calculated	
Total Members	Source		MEMBERS

Table 6. Futrix® Metadata Matrix Measure View

The code for the custom measure is executed in the Query: (Post) handle - one of the many user exits available in which Futrix® allows developers to execute code during end user processing. The Query (Post) handle runs just after Futrix® has queried the data source and returned a summary dataset, but before the results are displayed to the user. Futrix® custom code is written in SAS® Component Language.

To support our custom code we implement two custom attributes in Futrix®. The first is a custom attribute on the meta-item. This attribute is entered for the CLAIMS meta-item and contains the custom ID of the MEMBERS meta-item. The second custom attribute is the internal ID of the Total Members measure for the MEMBERS meta-item.

If these custom attributes are not null then our custom code runs a routine that queries the current Futrix® state. The custom code determines what dimensions and filters the user is using on the claims.

Our custom code uses these pieces of information - 1) the ID of the MEMBERS meta-item 2) the ID of the Total Members measure and 3) the current dimensions and filters being used by the user - to build a query against the MEMBERS meta-item. Futrix® has an API (Application Programmers Interface) that allows developers to run methods against a Futrix® object. One of the methods available is a QUERY method in which you can pass Futrix® parameters and have it query a meta-item and return a summary dataset. Right after the CLAIMS meta-item is queried by Futrix® our code makes a query of the MEMBERS meta-item and summarizes it to the appropriate level. This summarized members data then joins to the summarized claims data available in the Query (Post) method. PMPM – entered as a calculated measure – is then calculated by Futrix®.

Below are two viewpoints from the finished cube. In Figure 1, the data is summarized for the entire year of 2009. Total Paid Amount is $5,727,210. Total Members are 937,800. (Note that at the annual summary level the value of Total Members is actually the total number of member months whereby each member enrolled is counted for every month of enrollment. Members enrolled for the entire year are counted as twelve member months.) Paid PMPM is $6.11. Drug claims averaged $6.11 per member per month in 2009.

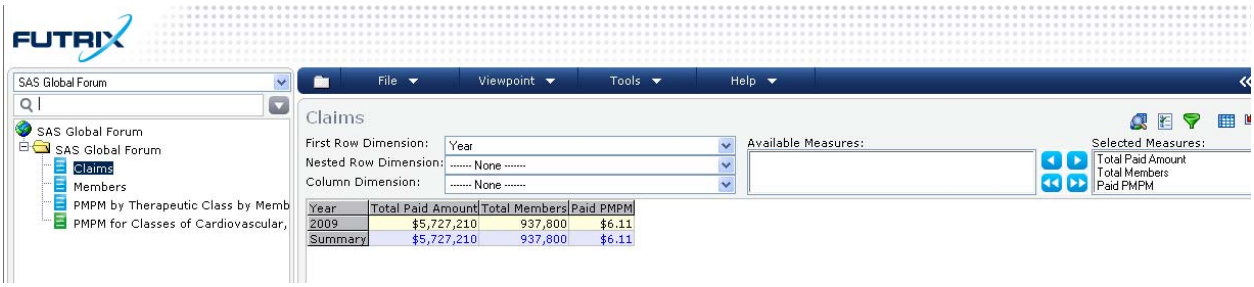
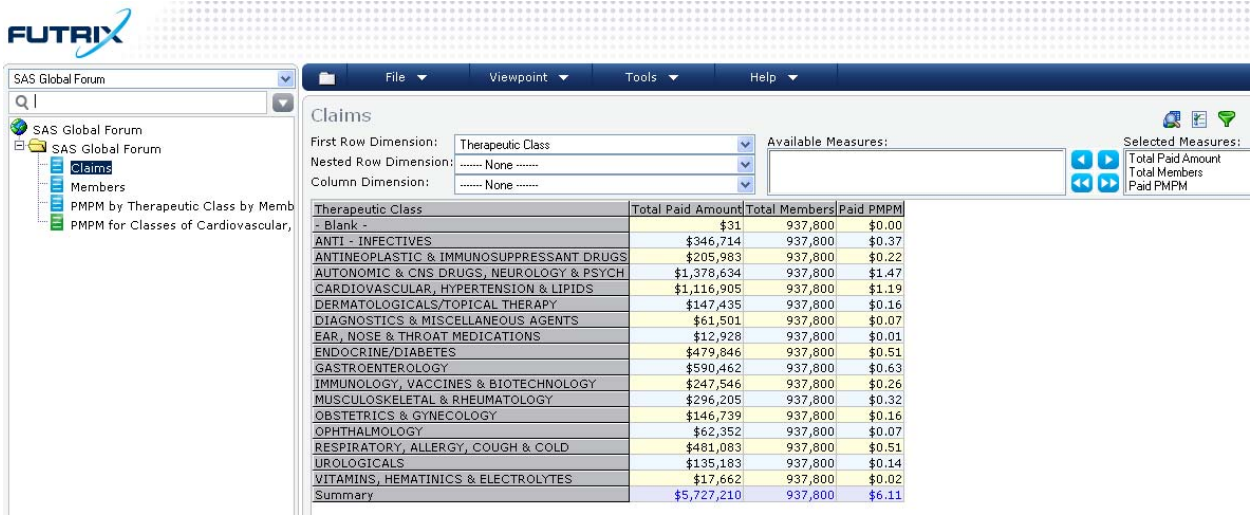


Figure 1. Futrix® Viewpoint of 2009 Drug Claims, Members and PMPM

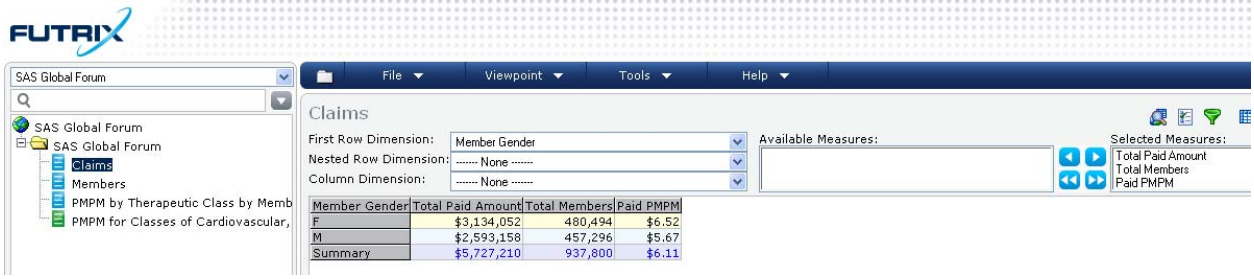
Classifying by Therapeutic Class provides an illustration of the semi-additive nature of Total Members. Total Members remains constant at 937,800 across all Therapeutic Classes of drugs. Across Therapeutic Class, Total Members is non-additive. In contast, Total Paid Amount is additive by Therapeutic Class.



Therapeutic Class	Total Paid Amount	Total Members	Paid PMPM
- Blank -	$31	937,800	$0.00
ANTI - INFECTIVES	$346,714	937,800	$0.37
ANTINEOPLASTIC & IMMUNOSUPPRESSANT DRUGS	$205,983	937,800	$0.22
AUTONOMIC & CNS DRUGS, NEUROLOGY & PSYCH	$1,378,634	937,800	$1.47
CARDIOVASCULAR, HYPERTENSION & LIPIDS	$1,116,905	937,800	$1.19
DERMATOLOGICALS/TOPICAL THERAPY	$147,435	937,800	$0.16
DIAGNOSTICS & MISCELLANEOUS AGENTS	$61,501	937,800	$0.07
EAR, NOSE & THROAT MEDICATIONS	$12,928	937,800	$0.01
ENDOCRINE/DIABETES	$479,846	937,800	$0.51
GASTROENTEROLOGY	$590,462	937,800	$0.63
IMMUNOLOGY, VACCINES & BIOTECHNOLOGY	$247,546	937,800	$0.26
MUSCULOSKELETAL & RHEUMATOLOGY	$296,205	937,800	$0.32
OBSTETRICS & GYNECOLOGY	$146,739	937,800	$0.16
OPHTHALMOLOGY	$62,352	937,800	$0.07
RESPIRATORY, ALLERGY, COUGH & COLD	$481,083	937,800	$0.51
UROLOGICALS	$135,183	937,800	$0.14
VITAMINS, HEMATINICS & ELECTROLYTES	$17,662	937,800	$0.02
Summary	$5,727,210	937,800	$6.11

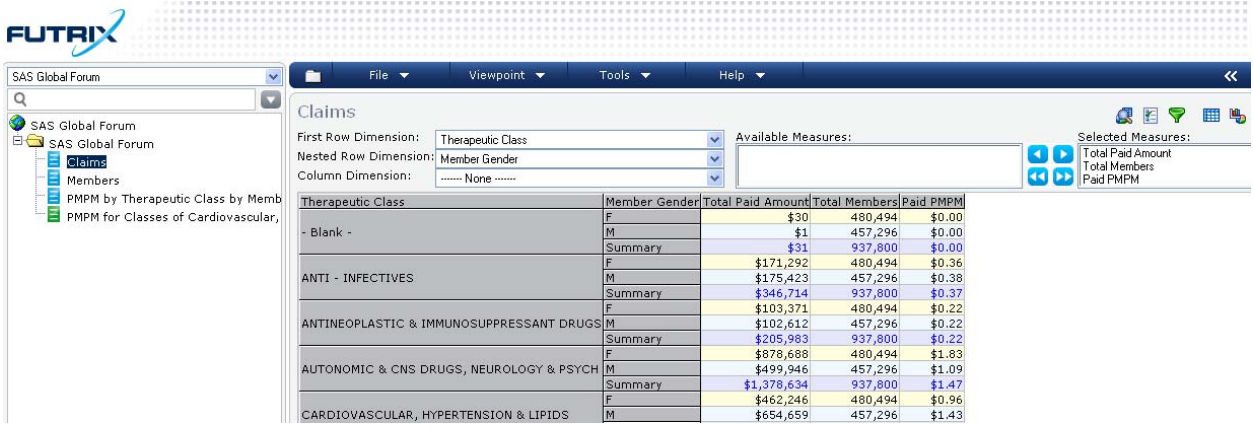
Figure 2. Futrix® Viewpoint of 2009 Drug Claims, Members and PMPM by Therapeutic Class

Drilling down by member gender provides an illustration of when the measure Total Members is additive. Total Female Members is 480,494 and Total Male Members is 457,296. Added together, Total Members is 937,800. PMPM averages $6.52 for Females and $5.67 for Males, for an overall average of $6.11.



Member Gender	Total Paid Amount	Total Members	Paid PMPM
F	$3,134,052	480,494	$6.52
M	$2,593,158	457,296	$5.67
Summary	$5,727,210	937,800	$6.11

Figure 3. Futrix® Viewpoint of 2009 Drug Claims, Members and PMPM by Member Gender



Therapeutic Class	Member Gender	Total Paid Amount	Total Members	Paid PMPM
- Blank -	F	$30	480,494	$0.00
	M	$1	457,296	$0.00
	Summary	$31	937,800	$0.00
ANTI - INFECTIVES	F	$171,292	480,494	$0.36
	M	$175,423	457,296	$0.38
	Summary	$346,714	937,800	$0.37
ANTINEOPLASTIC & IMMUNOSUPPRESSANT DRUGS	F	$103,371	480,494	$0.22
	M	$102,612	457,296	$0.22
	Summary	$205,983	937,800	$0.22
AUTONOMIC & CNS DRUGS, NEUROLOGY & PSYCH	F	$878,688	480,494	$1.83
	M	$499,946	457,296	$1.09
	Summary	$1,378,634	937,800	$1.47
CARDIOVASCULAR, HYPERTENSION & LIPIDS	F	$462,246	480,494	$0.96
	M	$654,659	457,296	$1.43

Figure 4. Futrix® Viewpoint of 2009 Drug Claims, Members and PMPM by Therapeutic Class and Gender

Behind the scenes in the above example, our custom code queries the enrollment meta-item asking for membership totals by Member Gender. It then joins these results to the claims summary that it just calculated and then displays the resulting table to the user.

In our custom code approach the join between claims and membership is a left join. In the final results there are only dimension values that exist in claims. Viewpoints based on this do not show enrollment or PMPM across levels of dimensions that experienced no claims. For instance, if a group had five healthy members who submitted no claims for a given month, their enrollment is not reflected in the cube. The workaround is to pad the claims data with any combination of membership values that do not exist in the claims data. These padded records have zero dollar amounts but the existence of the value in the data enables the value to be displayed in the report with the proper enrollment. The number of records is not enough to degrade performance significantly. We also create a new dimension or flag for these dummied claims records, allowing users to include or exclude them as desired.

APPROACH #1: FUTRIX® CUSTOM CODE PROS/CONS

PROS:

No restrictions on drill paths

No concern about secondary data structure.

CONS:

Requires knowledge of SAS® Component Language.

Requires workaround to pad claims.

While Futrix® cubes can be viewed in end user tools such as the SAS® Add-In for Microsoft Office, SAS® Enterprise Guide and SAS® Web Report Studio, the calculated measure of PMPM cannot be surfaced outside of Futrix® without additional SAS coding, such as a Stored Process.

APPROACH #2: FUTRIX® HEALTH

Futrix® has recently announced the release of Futrix® Health. With Futrix® as its core component, Futrix® Health is a solution geared specifically to the health care industry. A key feature of this solution is “linked measures.” Linked measures allow you to implement semi-additive measures such as MEMBERS on a CLAIMS data source. With this new product, measures like PMPM can be implemented out of the box. This eliminates the need for custom code and simplifies the cube development process. As with earlier versions, the solution requires two data sources, one for CLAIMS and one for MEMBERS. However, instead of using a custom measure for membership in CLAIMS one adds a ‘Linked Measure’ via the interface.

In a recent release of Futrix® Health, linked measures can be calculated using a variety of join options, including full outer joins. Users can optionally display enrollment totals even when there are no associated claims for the dimension value.

APPROACH #2: FUTRIX® HEALTH PROS/CONS

PROS:

Flexible join criteria eliminates the need for workaround to pad claims.

Easy to implement via the interface.

No custom code required.

CONS:

While Futrix® cubes can be viewed in end user tools such as the SAS® Add-In for Microsoft Office, SAS® Enterprise Guide and SAS® Web Report Studio, the calculated measure of PMPM cannot be surfaced outside of Futrix® without additional SAS coding, such as a Stored Process.

APPROACH #3: SAS® OLAP SERVER

Part of the SAS® Enterprise Intelligence Platform, the SAS® OLAP Server is a standards-compliant OLAP data source that retrieves results for Multidimensional eXpression (MDX) queries. The cubes can be designed and built using PROC OLAP or SAS® OLAP Cube Studio. SAS® OLAP Cube Studio provides a wizard to walk you through the design of a cube.

In the SAS® Enterprise Intelligence Platform, cubes are arranged with hierarchies, or drill paths, organized into dimensions. Measures are the analysis variables and can be source variables in the data, or defined using MDX logic.

The tables below list the Dimensions, Hierarchies, Levels and Measures for our cube.

Dimension	**Hierarchy**	**Levels**
Time	Time	YEAR QUARTER MONTH
Formulary1	Formulary1	THERAPEUTIC_CATEGORY_1 THERAPEUTIC_CATEGORY_2 THERAPEUTIC_CATEGORY_3 DRUG
Formulary2	Formulary2	THERAPEUTIC_CATEGORY_1 DRUG
Product1	Product1	REGION PRODUCT CLIENT GROUP
Product2	Product2	CLIENT GROUP
Member1	Member1	RELATION GENDER MEMBER_ZIP_CODE
Member2	Member2	PLAN_AREA MEMBER_ZIP_CODE
Pharmacy1	Pharmacy1	PHARMACY_STATE PHARMACY_CHAIN PHARMACY
Pharmacy2	Pharmacy2	PHARMACY_STATE PHARMACY
ProductType	ProductType	MAIL_RETAIL BRAND_GENERIC

Table 7. SAS® Dimensions, Hierarchies, and Levels

Measures
TOTAL PAID AMOUNT
__MEMBERS
MEMBERS
PMPM

Table 8. SAS® Measures

Note that there is only one hierarchy per dimension. Typically, a dimension can have a number of hierarchies. However, our solution requires only one hierarchy per dimension, which we will discuss later. A requirement of PROC OLAP is that, when there is only one hierarchy in a dimension, the hierarchy has to have the same name as the dimension itself. SAS® PROC OLAP informed us with the error message below:

```
ERROR: The hierarchy name "YrQtMoHier" is invalid for the "TimeDim1" dimension;
A hierarchy must have the same name as the dimension to which it belongs when
the dimension is defined with only one hierarchy.
```

Calculating PMPM is THE HOLY GRAIL of which we speak in the title of this paper. How were we to accomplish that using SAS® OLAP Server? If PMPM were calculated as Total Paid Amount divided by Total Claimants, the calculation would be simple. Total Claimants could be determined by selecting the distinct count of member ID's from the population of claims. At each level of claims it is possible to determine this number. However, PMPM is calculated as Total Paid Amount divided by All Possible Claimants. How are we to "look up" the number of possible claimants, or enrollees, in such a manner that does not double count when we begin to summarize?

Stuart Levine from the SAS® OLAP Research and Development team provided us with a solution: Build the cube on the low level N-way claims table, but then create a series of separate "non-leaf" aggregate tables for the membership counts. These aggregate tables contain the members count properly rolled up to the enrollment levels. In the claims cube, we define two measures: MEMBERS and PMPM. These are defined in such a way as to look up the correct member count from the appropriate aggregate table.

The PROC OLAP statement that identifies the name of the non-leaf table is the AGGREGATION statement. This is where the real magic for this solution takes place. In our solution we use the AGGREGATION statement with a list of enrollment levels to point to the specific table that contains the membership summarized for a particular hierarchy or combination of hierarchies.

Here is a sample AGGREGATION statement:

```
AGGREGATION REGION PRODUCT CLIENT GROUP  /  table = SGF2010.AGG_8 ;
```

To support this aggregation statement we build a table - SGF2010.AGG_8 - that has membership summarized by the combination of CLIENT GROUP PRODUCT and REGION. The table has a column for membership which we have called __MEMBERS.

In our PROC OLAP code we include a statement for the measure __MEMBERS as follows:

```
MEASURE __members column=__members  stat=sum aggr_column=__members;
```

The AGGR_COLUMN option tells SAS® OLAP Server to use the column __MEMBERS to supply the values for the measure __MEMBERS.

The SAS® OLAP Server automatically references data from the smallest available table. Since the non-leaf tables are always smaller than the N-way, whenever the server receives a request for an aggregation or summarized level, it will always pull from the table named in the AGGREGATION statement rather than the base N-way table. Thus, when the SAS® OLAP Server determines it should use the SGF2010.AGG_8 table to satisfy a query it retrieves the column __MEMBERS from that table. Note, that this table does not contain claims measures such as PAID. Instead, it must retrieve that from another source – either the NWAY or another aggregation.

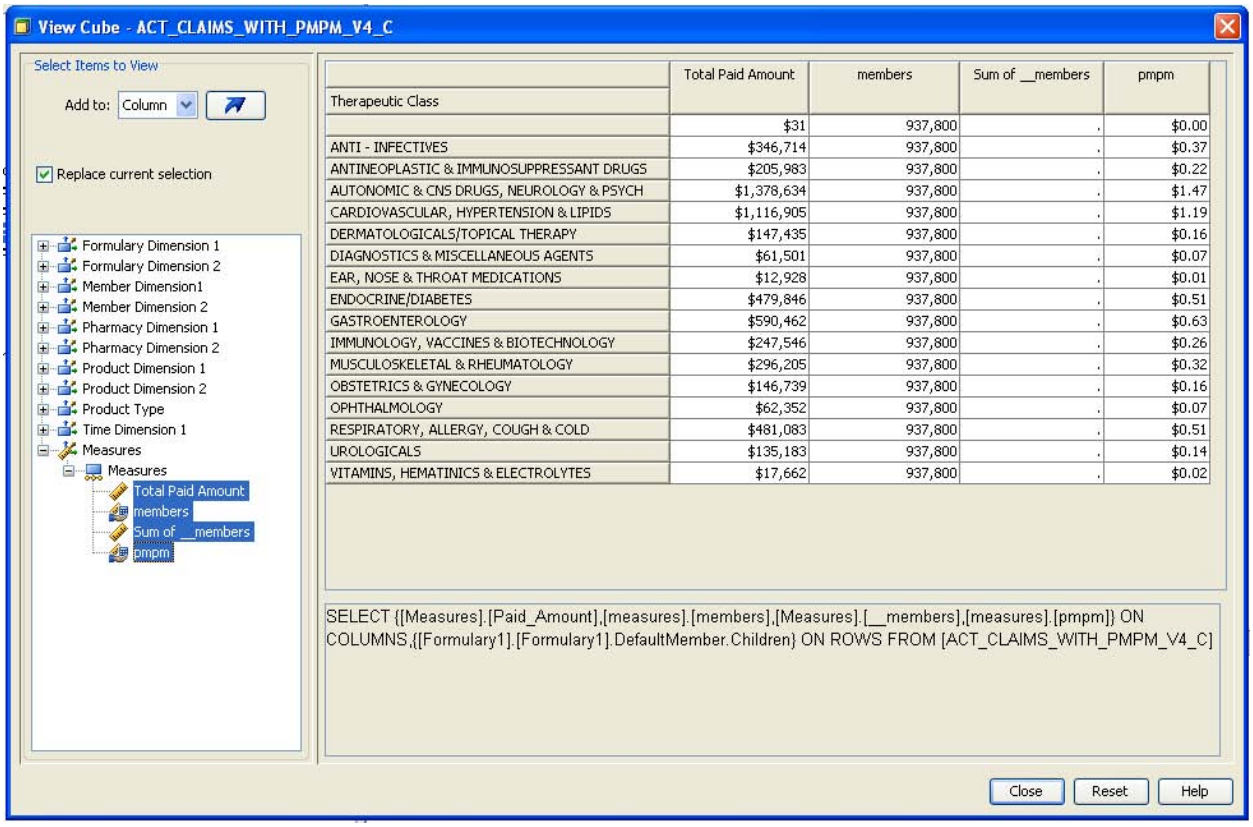
Later on we reference this measure in the DEFINE statement for the measure MEMBERS.

```
DEFINE MEMBER "[ACT_CLAIMS_WITH_PMPM_V4_C].[measures].[members]" as
"([measures].[__members], [Formulary1].[All Formulary1],
                          [Formulary2].[All Formulary2],
                          [Pharmacy1].[All Pharmacy1],
                          [Pharmacy2].[All Pharmacy2],
                          [ProductType].[All ProductType]
                          ) , FORMAT_STRING = 'COMMA32.' "  ;
```

In this statement we specify that the measure MEMBERS is defined by the set of the measure __MEMBERS and all levels of the claims specific dimensions. In other words, MEMBERS is defined as the value of the __MEMBERS measure without regard to the current level of any of the claims hierarchies. Behind the scenes the MDX code is telling SAS to only consider the membership dimensions when determining the correct aggregation for __MEMBERS.

To support this approach we need multiple aggregation tables - a minimum of one table for every combination of membership hierarchies. We automated the aggregation table creation through the use of a macro that determines all the combinations of hierarchies and creates the associated aggregation tables. The macro also produces the code for the AGGREGATION statement which we then drop into the PROC OLAP code generated by SAS® OLAP Cube Studio. The aggregation tables can also be added through the wizard. To enhance performance developers can also build aggregation tables for claims hierarchies as well as the combination of claims hierarchies and enrollment hierarchies. The tables for these aggregates will not contain the actual MEMBERS column. The SAS® OLAP Server will supply actual MEMBERS from the proper aggregate table of membership levels.

This approach requires the creation of a dummy measure, __MEMBERS in the claims base table. This measure is used simply in the definition of MEMBERS and PMPM and forces SAS® OLAP Server to look-up the proper value from the aggregation table. The dummy measure remains as part of the cube definition and is visible in cube viewers. Figure 5 below shows the dummy calculated measure, Sum of __MEMBERS. The values of the dummy measure will all be blank, but the measure can be hidden from users via an information map in map-based tools such as SAS® Web Report Studio.



Therapeutic Class	Total Paid Amount	members	Sum of __members	pmpm
	$31	937,800	.	$0.00
ANTI - INFECTIVES	$346,714	937,800	.	$0.37
ANTINEOPLASTIC & IMMUNOSUPPRESSANT DRUGS	$205,983	937,800	.	$0.22
AUTONOMIC & CNS DRUGS, NEUROLOGY & PSYCH	$1,378,634	937,800	.	$1.47
CARDIOVASCULAR, HYPERTENSION & LIPIDS	$1,116,905	937,800	.	$1.19
DERMATOLOGICALS/TOPICAL THERAPY	$147,435	937,800	.	$0.16
DIAGNOSTICS & MISCELLANEOUS AGENTS	$61,501	937,800	.	$0.07
EAR, NOSE & THROAT MEDICATIONS	$12,928	937,800	.	$0.01
ENDOCRINE/DIABETES	$479,846	937,800	.	$0.51
GASTROENTEROLOGY	$590,462	937,800	.	$0.63
IMMUNOLOGY, VACCINES & BIOTECHNOLOGY	$247,546	937,800	.	$0.26
MUSCULOSKELETAL & RHEUMATOLOGY	$296,205	937,800	.	$0.32
OBSTETRICS & GYNECOLOGY	$146,739	937,800	.	$0.16
OPHTHALMOLOGY	$62,352	937,800	.	$0.07
RESPIRATORY, ALLERGY, COUGH & COLD	$481,083	937,800	.	$0.51
UROLOGICALS	$135,183	937,800	.	$0.14
VITAMINS, HEMATINICS & ELECTROLYTES	$17,662	937,800	.	$0.02

Figure 5. SAS® OLAP Cube Studio view of 2009 Drug Claims, Members, the dummy measure (Sum of _members), and PMPM by Therapeutic Class

One requirement of this approach is that each hierarchy must be placed in a separate dimension. This is to allow us to use the DEFINE statement above to specify all the claim hierarchies. Earlier we gave the example of Time as a dimension that had several hierarchies, such as Year/Month/Day or Year/Quarter/Day or Year/Week/Day. SAS® OLAP permits multiple hierarchies within a dimension; but in order for this solution to work it is necessary to separate each hierarchy out into its own dimension. Further, in Version 9.1.3 SAS® OLAP did not allow a "level" (variable) to occur in more than one dimension. Thus this solution requires Version 9.2 SAS® OLAP Server to implement in anything but a very simple cube. In addition, if you use a level in more than one dimension, you have to give the level a different name even though it is still referring to the same source variable. In the code you will see these two dimension statements:

```
DIMENSION Product1
      CAPTION          = 'Product Dimension 1'
      SORT_ORDER       = ASCENDING
      HIERARCHIES      = (
         Product1
         ) /* HIERARCHIES */;

      HIERARCHY Product1
         ALL_MEMBER = 'All Product1'
         CAPTION    = 'Region --> Product --> Client --> Group'
         LEVELS     = (
            REGION PRODUCT CLIENT GROUP
            ) /* LEVELS */
         DEFAULT;

      LEVEL REGION
         CAPTION        =  'Region'
```

```
      SORT_ORDER     =  ASCENDING;

   LEVEL PRODUCT
      CAPTION        =  'Product'
      SORT_ORDER     =  ASCENDING;

   LEVEL CLIENT
      CAPTION        =  'Client'
      SORT_ORDER     =  ASCENDING;

   LEVEL GROUP
      CAPTION        =  'Group'
      SORT_ORDER     =  ASCENDING;


DIMENSION Product2
   CAPTION          = 'Product Dimension 2'
   SORT_ORDER       = ASCENDING
   HIERARCHIES      = (
      Product2
      ) /* HIERARCHIES */;

   HIERARCHY Product2
      ALL_MEMBER = 'All Product2'
      CAPTION    = 'Client --> Group'
      LEVELS     = (
         CLIENT_A  GROUP_A
         ) /* LEVELS */
      DEFAULT;

   LEVEL CLIENT_A
      COLUMN         =  CLIENT
      CAPTION        =  'Client'
      SORT_ORDER     =  ASCENDING;

   LEVEL GROUP_A
      COLUMN         =  GROUP
      CAPTION        =  'Group'
      SORT_ORDER     =  ASCENDING;
```

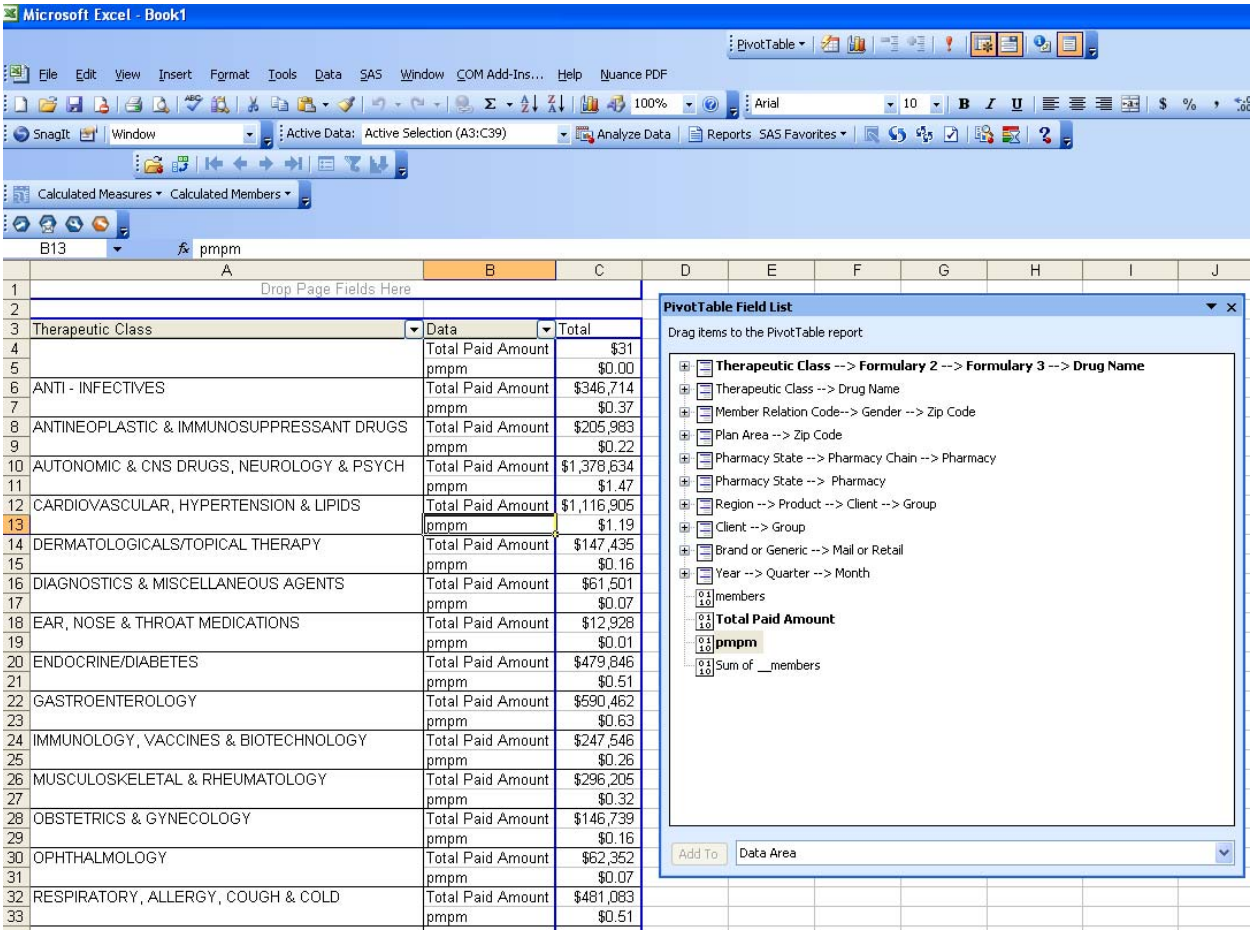
In the second dimension we have to give the levels a different name – CLIENT_A and GROUP_A – even though they use the same source column, specified by the COLUMN = option.

While building our cubes we also encountered the following error:

```
ERROR: CUBES CANNOT HAVE MORE THAN ONE TIME DIMENSION.
```

Thus we were not able to create all of the hierarchies that we wanted and had to settle for only one hierarchy for TIME, which we chose to be the YrQtMoHier. Allowing users to drill directly from Year to Month without going through Quarters was not an option.

OLAP cubes built in SAS can be surfaced to users in the myriad of tools available in the SAS® Enterprise Intelligence Platform – SAS® Enterprise Guide, SAS® Web Report Studio and SAS® Add-in to Microsoft® Office – allowing users to exploit cubes in the familiar world of a spreadsheet.



Therapeutic Class	Data	Total
	Total Paid Amount	$31
	pmpm	$0.00
ANTI - INFECTIVES	Total Paid Amount	$346,714
	pmpm	$0.37
ANTINEOPLASTIC & IMMUNOSUPPRESSANT DRUGS	Total Paid Amount	$205,983
	pmpm	$0.22
AUTONOMIC & CNS DRUGS, NEUROLOGY & PSYCH	Total Paid Amount	$1,378,634
	pmpm	$1.47
CARDIOVASCULAR, HYPERTENSION & LIPIDS	Total Paid Amount	$1,116,905
	pmpm	$1.19
DERMATOLOGICALS/TOPICAL THERAPY	Total Paid Amount	$147,435
	pmpm	$0.16
DIAGNOSTICS & MISCELLANEOUS AGENTS	Total Paid Amount	$61,501
	pmpm	$0.07
EAR, NOSE & THROAT MEDICATIONS	Total Paid Amount	$12,928
	pmpm	$0.01
ENDOCRINE/DIABETES	Total Paid Amount	$479,846
	pmpm	$0.51
GASTROENTEROLOGY	Total Paid Amount	$590,462
	pmpm	$0.63
IMMUNOLOGY, VACCINES & BIOTECHNOLOGY	Total Paid Amount	$247,546
	pmpm	$0.26
MUSCULOSKELETAL & RHEUMATOLOGY	Total Paid Amount	$296,205
	pmpm	$0.32
OBSTETRICS & GYNECOLOGY	Total Paid Amount	$146,739
	pmpm	$0.16
OPHTHALMOLOGY	Total Paid Amount	$62,352
	pmpm	$0.07
RESPIRATORY, ALLERGY, COUGH & COLD	Total Paid Amount	$481,083
	pmpm	$0.51

Figure 6. SAS® Add-In for Microsoft® Office view of 2009 Drug Claims, Members and PMPM by Therapeutic Class

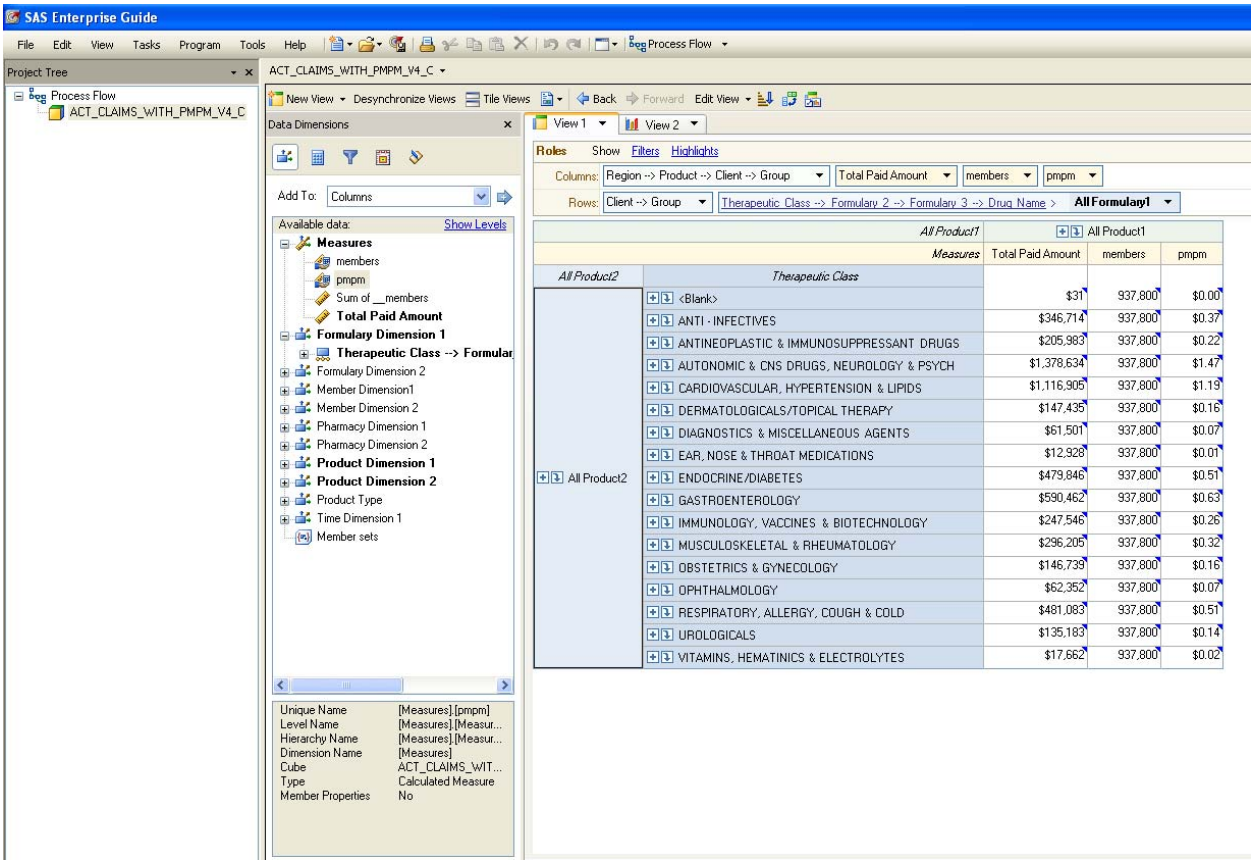


Figure 7. SAS® Enterprise Guide view of 2009 Drug Claims, Members and PMPM by Therapeutic Class

APPROACH #3: SAS® OLAP SERVER PROS/CONS

PROS:

PMPM calculations can be surfaced in a variety of tools included in SAS® Business Intelligence.

Easy to implement using PROC OLAP syntax.

CONS:

Requires extra coding outside of the PROC OLAP to create the secondary aggregate tables.

Requires workaround to pad claims.

Creates a dummy measure.

DATA CONSIDERATIONS

Working with Healthcare data offers its own unique challenges. Client renewals depend on various dimensions that are of interest to actuaries, such as the Client's Rating Category Code, which indicates whether the client is currently rated as small group or experience rated. In Pennsylvania clients with 50 or fewer members are rated as small group. Some groups cross back and forth between small group and large group, as employees are gained and then lost from year to year.

Claims are processed with the information available to the claims processing system at the time the claim is received. In contrast, enrollment data is corrected retroactively, as customers inform the insurer that certain employees are no longer with the company and their coverage is being dropped, for instance. It is possible to encounter the situation where there are claims for a given month but no enrollment for that month. Stakeholders must make the business decision on how to deal with these situations. One option is to decide that enrollment has the most recent information and update the claims accordingly. Another option is to report on discrepancies and address each individually.

CONCLUSION

PMPM is a key metric in the health care industry providing an average cost per member that can be used to spot trends in health care costs. While actuaries have a wealth of OLAP tools available to them now to analyze the data in their organization, this important statistic is not available out of the box. However, there are customizations that developers can perform to bring PMPM into the analysts' arsenal of tools. For those using Futrix®, the use of custom attributes, custom measures, and the supporting custom code makes PMPM available in the interface. Futrix® Health, which offers linked measures out of the box, is also an option. With SAS® OLAP Server, PMPM is available by using options available in PROC OLAP, with some supporting summary tables. This solution makes the PMPM metric available in any of the SAS® Business Intelligence tools that use cubes. So, if you are questing for the Holy Grail of PMPM it is very much obtainable in OLAP.

REFERENCES

- Rupinder Dhillon and Harry Droogendyk "Manipulating OLAP Cubes: Advanced Techniques for SAS® Programmers", http://support.sas.com/resources/papers/proceedings09/042-2009.pdf
- Ben Cochran,"Using PROC OLAP to Build Cubes with NON-Additive Measures", http://www2.sas.com/proceedings/mwsug/A02-2009.pdf-
- Creating and Exploiting OLAP Using the SAS® System Course Notes, 2007 SAS® Institute Inc., Cary, NC
- Futrix® 6.1 - Customization Reference – February 2010, Futrix® Limited , Wellington New Zealand

ACKNOWLEDGMENTS

Special thanks go to Maryann Hiscock and Stuart Levine at SAS®. Also, we would like to express appreciation to Syuzanna Zakharova at Futrix® Limited for her support. We also gratefully acknowledge Sarah Mitchell of Qualex Consulting Services for her work in developing custom code in Futrix.

RECOMMENDED READING

- Clare Nicklin, "Identifying Complications and Overcoming Problems Using SAS® Enterprise BI Server" http://support.sas.com/resources/papers/proceedings09/046-2009.pdf
- Ben Zenick and Brian Miles, "Beyond the Basics: Advanced OLAP Techniques" http://www2.sas.com/proceedings/sugi31/219-31.pdf
- Brian Miles and Ben Zenick, "Custom Calculations Are a Snap When Using MDX and OLAP" http://support.sas.com/resources/papers/proceedings09/020-2009.pdf

CONTACT INFORMATION

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

Name: Heather Y. Seeno
Enterprise: Highmark, Inc.
Address: 1800 Center Street SP3W
City, State ZIP: Camp Hill, PA 17089-0089
Work Phone: 717.302.2361
Fax: 717.302.3502
E-mail: Heather.Seeno@highmark.com

Name: Alan Leach
Enterprise: Qualex Consulting Services
Address: 2217 Cleneay Ave
City, State ZIP: Norwood, OH 45212
Work Phone: 513-731-8831
Fax: 603-807-5671
E-mail: Alan.Leach@qlx.com
Web: www.qlx.com