MindStream Analytics

Tips and Tricks to get more out of Essbase

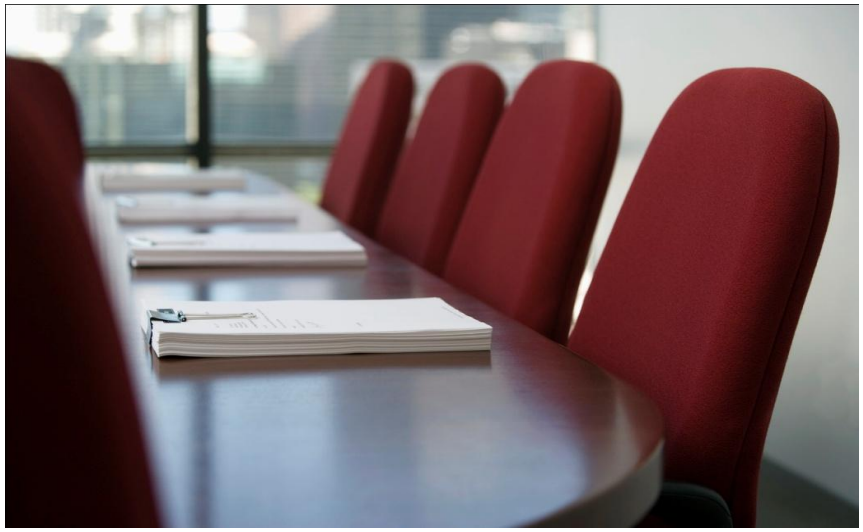
MINDSTREAM® ANALYTICS

Webinar will begin at 12:05pm

03.29.13

Agenda

- Introduction MindStream Analytics – Phil Sable
- Overview of topics covered
- Tips and Tricks to get more out of Essbase
- Summary
- Questions



About MindStream Analytics

WHEN YOU'RE LOOKING FOR THE BEST…		
MINDSTREAM IS A LEADER IN…	IS A MEMBER OF THE…	HAS EXPERIENCE THROUGH…
- EPM Solutions Focused Around Oracle Hyperion Product Suite - Practices - Planning - Financial Close Process - BI / Data Integrations - Infrastructure - Strategy / Training - Managed Services	-Oracle Partner Advisory Board - Hyperion SIG - Board Member - Domain Lead - Oracle Beta Development & User Experience Team	- Hundreds Of Customer Implementations - Leading Product & Industry Experts/Analysts - More Than A Decade Of Execution Experience

ORACLE Certified Professional

OAUG Hyperion SIG THE HOME FOR HYPERION

affiliate of OAUG oracle applications users group

ORACLE CERTIFIED PARTNER Specialized Oracle Hyperion

ORACLE Platinum Partner

About Phil Sable

Phil Sable

- 13 years of Planning/Essbase Experience
- 16 years of BI Experience
- Oracle Certified Expert in Hyperion Planning & Essbase
- Sr. Consultant , Planning & Essbase Practice - **MindStream Analytics**

Contact Information

Email: psable@mindstreamanalytics.com

Telephone: 734-377-7718

Topics Covered

- Nested Fixes – limit passes through the database
- Conditional Fix – Using SET EMPTYMEMBERSETS
- Dense Fixes/Calculation Blocks
- Creating Blocks – Dealing with Block Creation issues
- Substitution Variables – Using them as Macros
- Cache settings – Increasing database performance
- Data export using Calculation scripts
- @Xwrite – Moving data between applications
- Varying Attributes – tracking slowly changing dimensions
- Typed Measures
- Smartview Trick - Converting #Missing to Zeros

Nested Fixes – limit passes through the database

Nested fixes are used for a number of tasks. Fixes are used to limit the portion of the database on which calculations are performed. They can also be useful in optimizing calculations. These are the tasks I will be covering.

- Fixes used to limit passes through the database
- Fixes used to handle exceptions in the calculation process
- Fixes used to control conditional execution of code segments

Example – Handling Leap Year

MINDSTREAM ANALYTICS

Not using nested fixes causes two passes thru the database

```
FIX(("FY2012","FY2016",@RELATIVE"AccountingMethod",0),@RELATIVE("YearTotal",0),@RELATIVE("
    Maturity",0),@RELATIVE("BusinessUnit",0),@RELATIVE("CreditRatings",0),@RELATIVE("Counterpa
    rty",0),@RELATIVE("SYS-REG",0))

"ACTIVITY"(
IF("Term" <= 366 and "Term" != #MISSING)
    "Total Book Value of short-term bonds"=  "CurBookValue“;
ENDIF
) ENDFIX

FIX("FY2011","FY2013","FY2014","FY2015",RELATIVE(“AccountingMethod",0),@RELATIVE("YearTotal"
    ,0),@RELATIVE("Maturity",0),@RELATIVE("BusinessUnit",0),@RELATIVE("CreditRatings",0),@REL
    ATIVE("Counterparty",0),@RELATIVE("SYS-REG",0))

"ACTIVITY"(
IF("Term" <= 365 and "Term" != #MISSING)
    "Total Book Value of short-term bonds"=  "CurBookValue";
ENDIF
)
ENDFIX
```

Nested Fixes

MINDSTREAM ANALYTICS

Using nested fixes limits the number of passes through the database

```
FIX(@RELATIVE("AccountingMethod",0),@RELATIVE("YearTotal",0),@RELATIVE("Maturity",0),
@RELATIVE("BusinessUnit",0),@RELATIVE("CreditRatings",0),@RELATIVE("Counterparty",0),@RELA
    TIVE("SYS-REG",0))
     FIX( "FY2011","FY2013","FY2014","FY2015")   *** FIRST NESTED FIX STARTS HERE ***

"ACTIVITY"(
IF("Term" <= 365 and "Term" != #MISSING)
    "Total Book Value of short-term bonds"=  "CurBookValue";
ENDIF
)
ENDFIX  *** FIRST NESTED FIX ENDS HERE ***
    FIX("FY2012","FY2016")  *** SECOND NESTED FIX STARTS HERE ***

"ACTIVITY"(
IF("Term" <= 366 and "Term" != #MISSING)
    "Total Book Value of short-term bonds"=  "CurBookValue";
ENDIF
)
    ENDFIX  *** FIRST NESTED FIX ENDS HERE ***
ENDFIX
```

ORACLE Platinum Partner

Conditional Statements within a FIX

By using a combination of Fix, Set, Substitution variables, and Calculation functions, you can make a fix statement that is in fact a conditional statement. This is useful in calculation scripts that need to be time sensitive. Forecasting applications are a good example. We will use the following structures to build this conditional fix statement.

- EMPTYMEMBERSETS: This set statement tells Essbase to ignore any fix statements that have no members
- @REMOVE: This Calculation command allows you to remove members from a list of members
- FIX: This statement contains the members on which we wish to fix
- SUBSTITUTION VARIABLE: A global variable used to store a member name
- The following example uses these to process forecast months; it ignores all actual months

Creating a Conditional Statement Using a FIX

If the “CurActMonth” is set to BegBalance, the code in the fix will be run, since removing Beg_Balance from Jan has no effect.

If the CurActMonth is set to Feb, the remove statement will include BegBlance, Jan, Feb. Since Jan is in this list, it is removed from the fix. Now the fix is empty and its code is skipped.

```
SET EMPTYMEMBERSETS ON;

FIX (@Remove(Jan , Begbalance: &CurActMonth))
%PROCESS_FORECAST();
ENDFIX
```

Dense Fixes/Calculation Blocks

Calculation Blocks are also known as dense fixes.

- Calculation blocks should be based on a dense member whenever possible. Otherwise many blocks may be called, dramatically slowing calculations.
- Using a sparse member calls all blocks with that sparse member, resulting in much slower execution.
- A Calculation block member does not necessarily limit the calculation to a single member. You may assign other members in the block.
- Calculation blocks are required when using an if statement in a calculation script.
- You may select any dense member from the appropriate dimension to set your calc block. It does not need to be one you are actually calculating.

Calculation Block Example

```
/* HERE WE ARE FIXING ON THE SPARSE MEMBER WE WANT TO CALCULATE */
FIX(@RELATIVE("AccountingMethod",0),@RELATIVE("YearTotal",0),@RELATIVE("Maturity",
    0),@RELATIVE("BusinessUnit",0),@RELATIVE("CreditRatings",0),@RELATIVE("Counterp
    arty",0),@RELATIVE("SYS-REG",0))

"Federal_Unemployment"(   /* OPENING THE CALCULATION BLOCK WITH A DENSE MEMBER */
    IF(("Monthly_Total_Salary" + "Bonus Total" + "LTI") * "Fed_UEMP_Rate"->"BegBalance"-
    >"NO_EMPLOYEE"->" E00403-NO_DEPT " >= "Fed_UEMP_Max"->"BegBalance"-
    >"NO_EMPLOYEE"->“E00403-NO_DEPT")
    "Federal_Unemployment" = "Fed_UEMP_Max"->"BegBalance"->"C00403-NO_DEPT";
     ELSE
     "Federal_Unemployment" = ("Monthly_Total_Salary" + "Bonus Total" + "LTI") *
    "Fed_UEMP_Rate"->"NO_EMPLOYEE"->”E00403-NO_DEPT ";
    ENDIF /* is >= Max
ENDIF;
                              )          /* CLOSING THE CALCULATION BLOCK*/
```

Creating Blocks & Creation issues

- Often in Essbase you will write calculations that won't seem to work. This is generally a block creation issue. There are a number of ways to create blocks.
- DataCopy:
  - This method is best used when you really need to copy data from a sparse to a sparse dimension example Version to Version.
  - If all you need do is create blocks, consider writing a "0" to a single dense member intersection.
- CREATEBLOCKONEQ: Creates blocks if a sparse member has its value set by something other than a constant. Default is off. I would set this to "on" either in the database or most scripts.
- CREATENONMISSINGBLK This command is much more sensitive.
  - It processes all potential blocks and creates any where the formula returns a non-missing value.
  - It is very slow! Use only within Fix commands on small portions of the database.
  - Use SET CREATENONMISSINGBLK "off" to stop processing blocks when not needed.

Creating Blocks Examples

```
SET CREATENONMISSINGBLK ON;

FIX(@RELATIVE("AccountingMethod",0),@RELATIVE("YearTotal",0),@RELAT
   IVE("Maturity",0),@RELATIVE("BusinessUnit",0),@RELATIVE("CreditRatin
   gs",0),@RELATIVE("Counterparty",0),@RELATIVE("SYS-REG",0))
            FIX( "FY2011","FY2013","FY2014","FY2015")
"ACTIVITY"(
IF("Term" <= 365 and "Term" != #MISSING)
"Total Book Value of short-term bonds"=  "CurBookValue";
ENDIF
)
            ENDFIX
SET CREATENONMISSINGBLK Off;

CREATEBLOCKONEQ ON;
Working = Actual * 1.05;
```

Substitution Variables – How They Work

- A substitution variable is set up by an administrator to simplify maintenance of things like calc scripts and retrieves
- The question I always get is: what can I put in a variable?
- Are these valid in a substitution variable?
  1. Jan
  2. “Jan”
  3. Jan:Feb
  4. “Jan:Feb”
  5. Jan,Feb
  6. “Jan”,”Feb”
- The answer is yes all of these can be put in a variable

Substitution Variables – How They Work (continued)

- Before a calc script (or whatever it is) runs, it replaces the variable with the value the variable is saved to. It then validates and runs the calc script (or whatever it is).

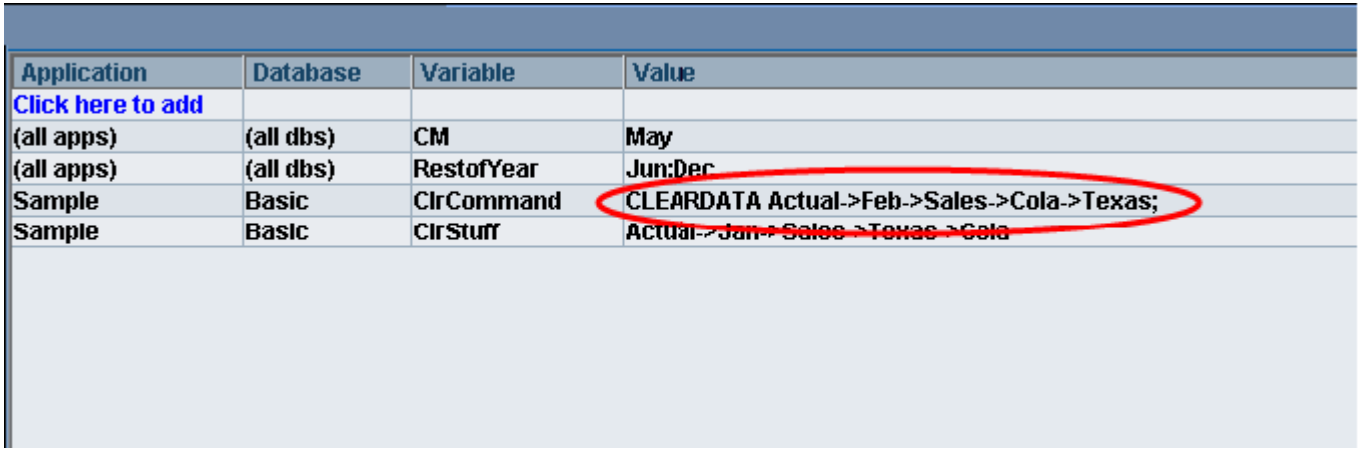
- Do you need double-quotes around a variable value like Actual->Sales?
- No, because you wouldn't normally put the term "Actual->Sales" in double-quotes in a calc script

- You would just say Actual->Sales

- In other words, put quotes around something in a variable if you would want quotes around them in a calc script

Substitution Variables – Using them as Macros

- You can actually set a variable to anything you want.
- •Check out this substitution variable:



Application	Database	Variable	Value
Click here to add			
(all apps)	(all dbs)	CM	May
(all apps)	(all dbs)	RestofYear	Jun:Dec
Sample	Basic	ClrCommand	CLEARDATA Actual->Feb->Sales->Cola->Texas;
Sample	Basic	ClrStuff	Actual->Jan->Sales->Texas->Cola

&ClrCommand= CLEARDATA Actual->Feb->Sales->Cola->Texas;

Substitution Variables – Calc Lines

- Yes, that's a full line from a calc script (including a semi-colon).
- Watch how you can call it and then run it in a calc script:

```
SET UPDATECALC OFF;

&ClrCommand
```

  - And yes, it actually runs.

Substitution Variables – Whole Scripts

- You can even take this a step further and put a series of lines into a script like this:

Application	Database	Variable	Value
Click here to add			
(all apps)	(all dbs)	CM	May
(all apps)	(all dbs)	Header	SET UpdateCalc Off; SET CalcParallel 4; SET Msg Summary;
(all apps)	(all dbs)	RestofYear	Jun:Dec
Sample	Basic	ClrCommand	CLEARDATA Actual->Feb->Sales->Cola->Texas;
Sample	Basic	ClrStuff	Actual->Jan->Sales->Texas->Cola

- And then call it like this:

Script

```
&Header

&ClrCommand
```

Substitution Variables – What's The Point?

- You can put the common lines from all of your calc scripts into a substitution variable and call them from all your scripts

- No more repeating "SET UpdateCalcOff;" at the top of every script
- Just put &Header at the top of each script

- And if your header needs to be changed, just change the variable
- Basically, you're writing your own macros… using variables

Cache settings – Increasing database performance

- In Essbase cache size settings can significantly affect database and general server performance.

- Essbase uses five memory caches to coordinate memory usage, they are
  - Index cache
  - Data file cache
  - Data cache
  - Calculator cache
  - Dynamic calculator cache

- Because of our limited time and the complexity of the last two items we will cover only the first three

Cache settings – (Continued)

- Essbase provides default size settings for each cache but you can adjust the sizes as needed for each database to optimize Essbase database performance

- Appropriate cache size is affected by many factors, including:
  - Database size
  - Block size
  - Index size
  - Available server memory

- The settings that you should use for each of the caches that you can configure depend on data distribution and the dense/sparse configuration of the database

Sizing the Index Cache

- The index is stored in index files . When a database is active, the most recently accessed index pages are held in the index cache. How much of the index can be held in memory simultaneously depends on the amount of memory that you allocate to the index cache.
- The size of index pages is fixed at 8 K to reduce input-output overhead, as well as to simplify database migration.
- The effectiveness of the index cache size depends on the nature of the calculation.
- For example, if you were reloading and recalculating an entire database (such as a database that is refreshed each month), a high index cache size is not helpful, because Essbase is creating blocks rather than searching the index cache for existing blocks during calculation.

Sizing the Index Cache – (Continued)

- The size of the Index Cache should be set to the combined size of all essn.ind files, if possible; otherwise, as large as possible.
- Do not set this cache size higher than the total index size, because no performance improvement results
- The calculation for the space required to store the index files (essxxxxx.ind) uses the following factors:
  - Number of existing blocks
  - 112 bytes—the size of an index entry
- To calculate the size of a database index, including all index files, perform the following calculation:
    - number of existing blocks * 112 bytes = the size of database index

Example

- Assume a database with 15,000,000 blocks.
  - 15,000,000 blocks * 112 = 1,680,000,000 bytes

Data File Cache

- The data file cache holds data files (.pag files) in memory if you are using direct I/O. If you are not using direct I/O, the data file cache is not used. How much of the data within data files can fit into memory at one time depends on the amount of memory you allocate to the data file cache.

- In general, if you must choose whether to allocate memory to the data cache or to the data file cache, choose the data file cache if you are using direct I/O.

- Data File Cache should be set to the combined size of all essn.pag files, if possible; otherwise, as large as possible

- In general, if you are using direct I/O, make the data file cache as large as system resources allow. Again, if you are using buffered I/O, the data file cache is not used.

Sizing the Data Cache

- Data blocks reside on physical disk and in memory. The number of blocks that can be held in the data cache simultaneously depends on how much memory you allocate to the data cache.
- When a block is requested, Essbase searches the data cache for the block. If Essbase finds the block in the cache, it is accessed immediately.
- If the block is not found in the cache, Essbase searches the index for the appropriate block number and then uses the index entry of the block to retrieve it from the appropriate data file on disk.
- Retrieving a requested block from the data cache is faster and therefore improves performance.

Sizing the Data Cache

- Remember if you must choose whether to allocate memory to the data cache or to the data file cache, choose the data file cache if you are using direct I/O.
- The size of the Data Cache should be set to 0.125 * the value of data file cache size
- Increase value if any of these conditions exist:
  - Many concurrent users are accessing different data blocks.
  - Calculation scripts contain functions on sparse ranges, and the functions require all members of a range to be in memory (for example, when using @RANK and @RANGE).

Data export using Calculation scripts

In the most recent releases of Essbase a new export functionality has been added. It allows you to export data from BSO databases, from within a calculation script.

- There are two associated commands that you need to use.
  - DATAEXPORTOPTIONS – This statement allows you to control how the data will be exported
  - DATAEXPORT – This statement actually does the export and defines information about the file based on the options you selected.
- The export command is best used within Fix statements.

- The export command executes MUCH faster than Essbase report scripts.

- It offers much more flexibility then the existing Export option.

Data export using Calculation scripts (Continued)

Options commands include

- DATAEXPORTENABLEBATCHINSERT – allows batch insert into tables if the ODBC driver supports it.
- DEXPSQLROWSIZE – allows you to control how many rows are inserted at a time. This command does not affect batch size.
- EXPORTFILESIZELIMIT – sets a limit to the export files size. Default 2 GB. It automatically creates additional files as needed.
- DATAEXPORTCOND – allows you to further filter the data, based on data values rather than members.

DATAEXPORTOPTIONS Command

- DataExportLevel - All, Level0, Input.
- DataExportDynamicCalc – Do you want dynamically calculated members exported? This will slow exports if ”on” since the values need to be calculated.
- DataExportNonExistingBlocks – Exports all potential blocks, used with export dynamic calcs to get all data.
- DataExportColFormat – If “on” it exports in column format. I would recommend this, especially for backups and exports to other systems.
- DataExportColHeader – For text outputs, it sets the dense dimension to be used as column headers. Pick a small dense dimension or the number of columns may be excessive.
- DataExportRelationalFile - Formats the file for a relational export.
- DataExportOverwriteFile – Only for text files. It will overwrite existing files if “on”.

DATAEXPORTOPTIONS Example

```
SET DATAEXPORTOPTIONS {
DataExportLevel LEVEL0 ;
DataExportDynamicCalc OFF ;
DataExportNonExistingBlocks OFF ;
DataExportPrecision 9 ;
DataExportColFormat ON ;
DataExportColHeader "Period" ;
DataExportRelationalFile OFF ;
DataExportOverwriteFile ON ;
} ;
```

DATAEXPORT command

- File - exports a text file. You specify file/path, Delimiter, and MissingChar.
- Essbase interprets the path in context to the server. Export files cannot be written to a client.
- Relational output. dsnName, table name, user, and password.
- The table must exactly match the output columns.
- The table is not cleared by this command.
- Substitution variables can be used.
- BIN output. Requires only file name.
- It is used for static backups
- Both the outline and file must have the same date time stamp .

DATAEXPORT Examples

Relational Export

```
FIX(&BY, "WORKING", "BUD_FCST", "BUD_REORG", "BUD", "TARGET_1",
   "TARGET_2", "TARGET_3", @LEVMBRS("CostCenters",0),
   @DESCENDANTS ("YearTotal", 0), @LEVMBRS("Products",0),
   @LEVMBRS("Adjustments",0), @LEVMBRS("Measures",0),
   @LEVMBRS("Projects",0), @LEVMBRS("Subledgers",0));
   DATAEXPORT "DSN" "FDWSTAGE" "HYP_Budget" "ESSBASE" "Z4H2a6tuv6kMIr";
ENDFIX
```

File Export

```
FIX (@DESCENDANTS("NET_INCOME",0),
   "Exact",@LEVMBRS("FundCode",0),@LEVMBRS("Year",0),@LEVMBRS("Operating
   Unit",0),@LEVMBRS("Rate",0),"ACT","ACTIVITY",@DESCENDANTS("C00203",0),@
   LEVMBRS("Subledger",0), @DESCENDANTS("YearTotal",0));
DATAEXPORT "File" "|" "E:\Batch\Level0_ACT.txt" "#MI";
ENDFIX;
```

@Xwrite – Moving data between applications - Example

- Used to send data from one Essbase Database to another
  - Very useful for Planning applications that have workflows in multiple databases

```
//ESS_LOCALE English_UnitedStates.Latin1@Binary
Name: Drivers
Purpose:  To feed the IncStmt database the drivers it needs from this Volume database. It has been determined through testing
      this method creates the least amount of extra blocks and performs the fastest while creating blocks for us.
Created:  09/14/2010
Author:  Alex Ladd

*/
SET UPDATECALC OFF;


FIX ("Actual","Curr","No Entity")
 Fix("Channel","PR_EO")

  /* Gross New EO Accounts Driver */
 "MTD Count Gross Deposit Accounts" (
  @XWRITE("MTD Count Gross Deposit Accounts", _RevCube_, "Value","Gross Account EO");
 )
```

- BEWARE - The Intelligent Calculator essentially shuts off this function
  - Turn UpdateCalc OFF

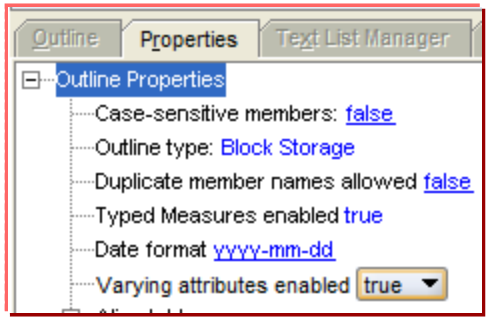
Varying Attributes

- Gives the user the ability to track things by Attribute groupings as they change over time or Perspective.

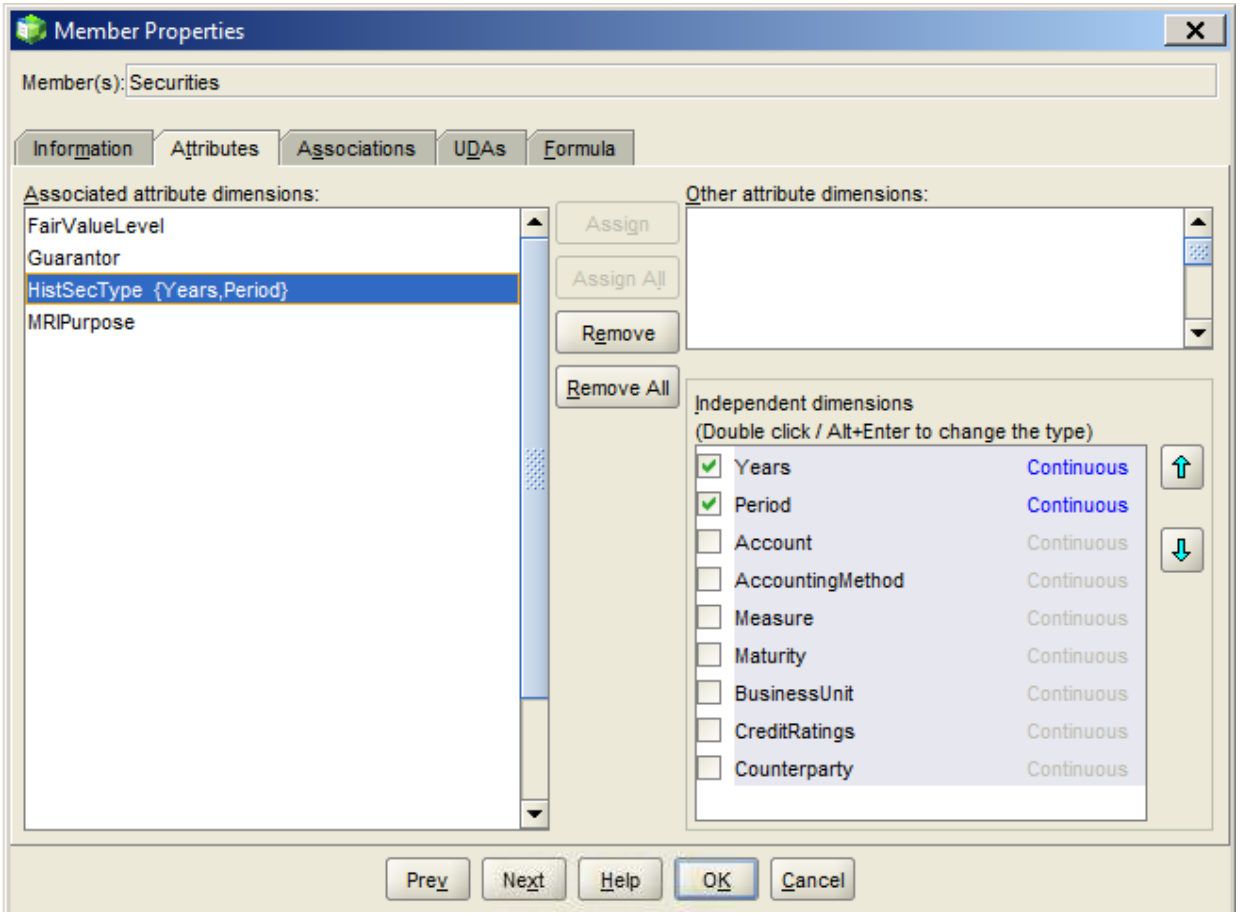
- For instance, a bank that has investments that move from one Security Type to another Security Type wants to see the data where it existed in a given period. Varying attributes allows them to see the data in the Security Type that the investment was in at any point in time.

To Implement

First, Enable Varying Attributes in the Outline



Varying Attributes

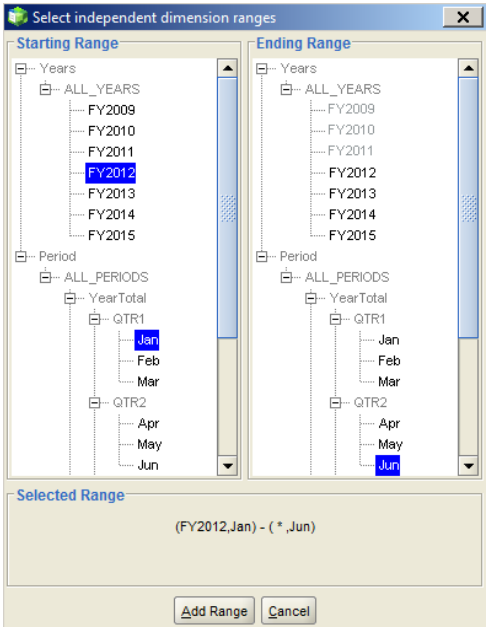


Then edit the member properties of the source dimension of the attribute and choose which dimensions are independent

Two types of Independent Dimensions:

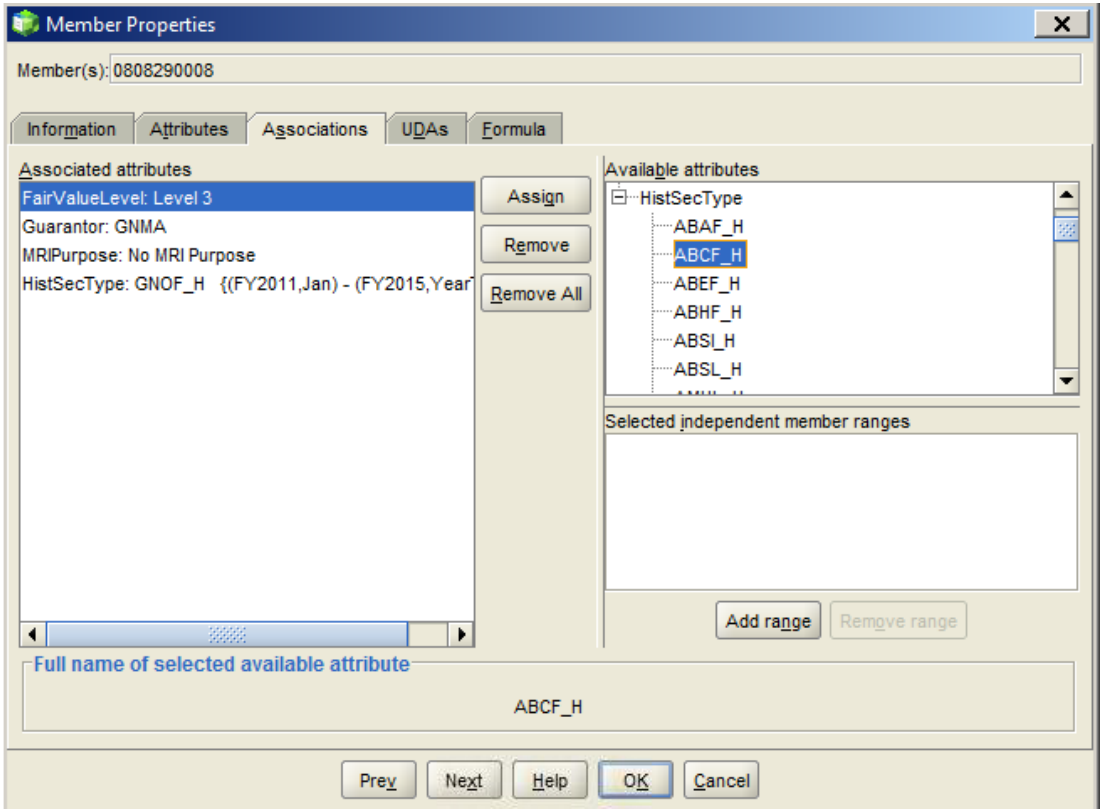
- Continuous
  - Example: Time
  - Example: Years
- Discreet
  - Example: None

Varying Attributes



Add Range Dialog

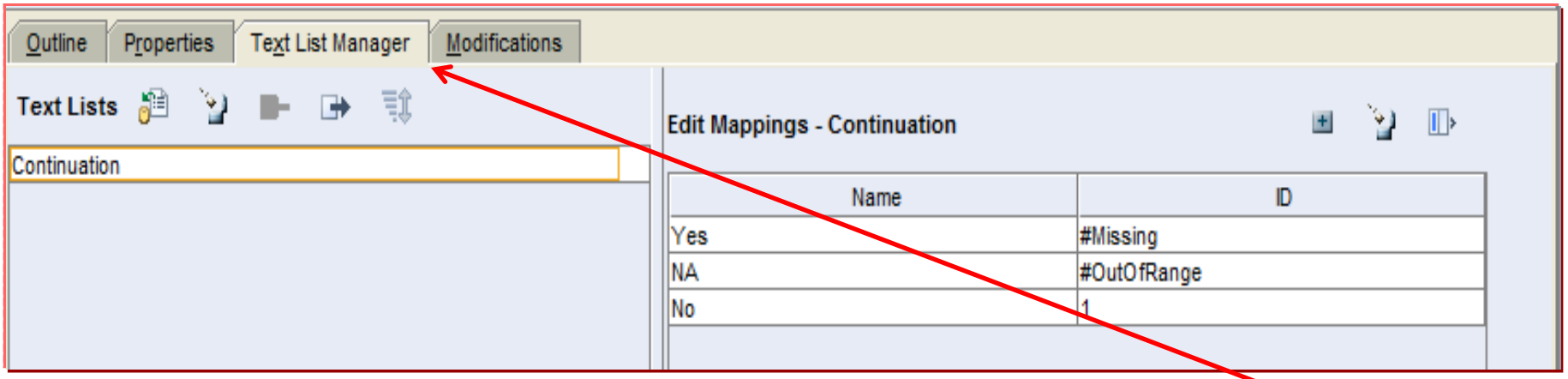
Choose a base member to assign the varying attribute to and select the varying attribute. Once you have selected the varying attribute the “Add Range” button becomes active. Once you add a range you may assign the varying attribute.

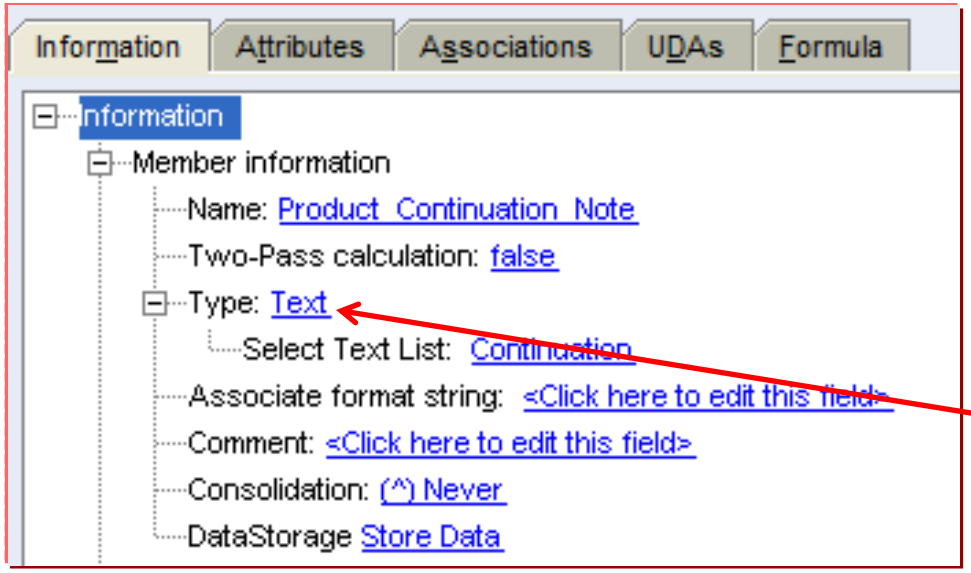


Typed Measures

- Similar to Planning Smart Lists
- Typed Measures store a value in Essbase that references a piece of Text in a Text List
- In order to use it you must enable it for the database

Typed Measures - Member Properties





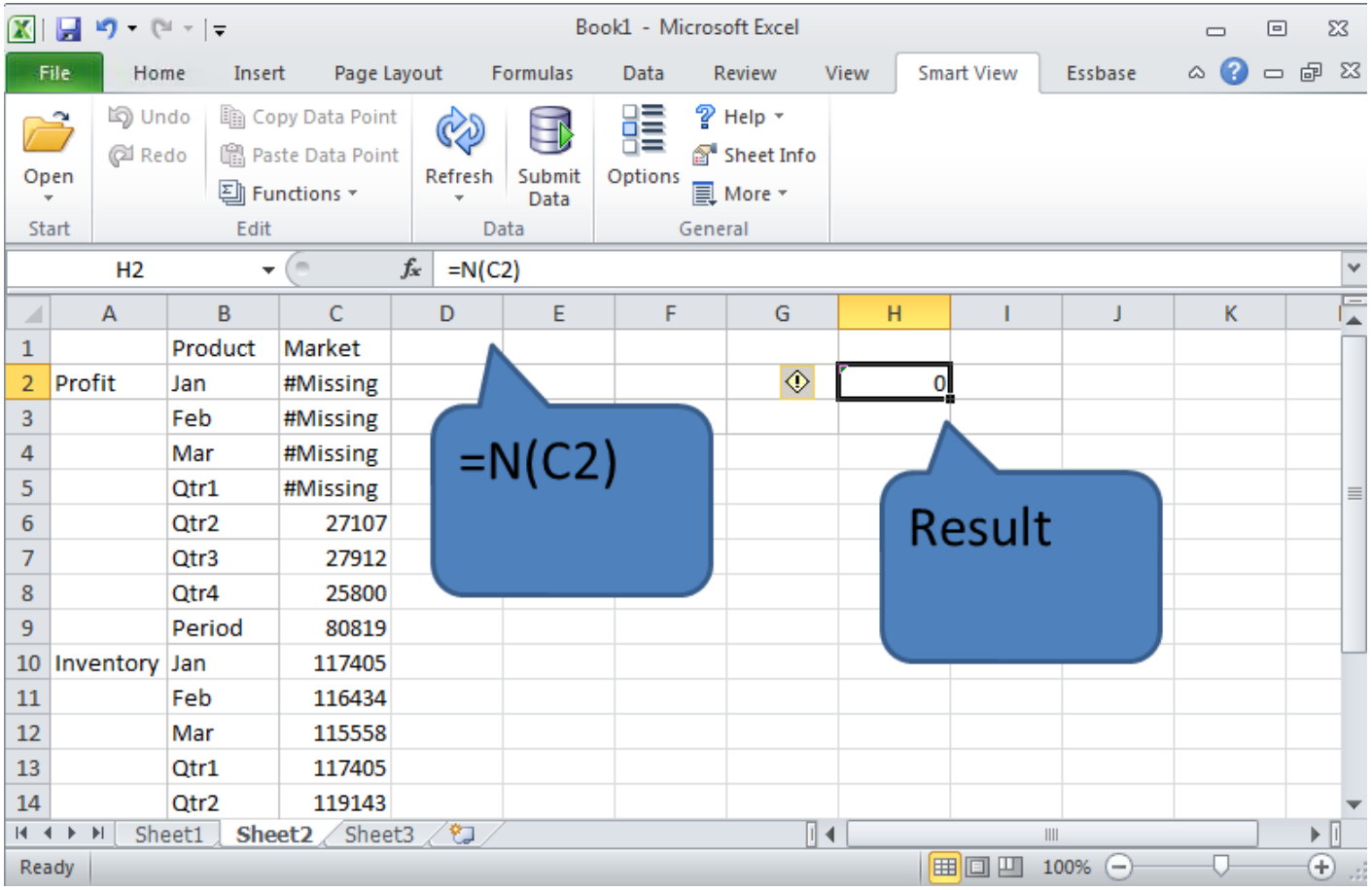
In the Outline Editor you need to build a Text List to relate a Text type measure to a value

Relate the Text Member to the Text List in the Member Properties Window

Smartview Trick – Converting #Missing to Zeros

- If you try to do something like subtract #missing (or another non numeric value), you can get the #Invalid in your calculated cell.
- To get around this, you can modify the formula
- Just add N() around the cell references.

Smartview Trick

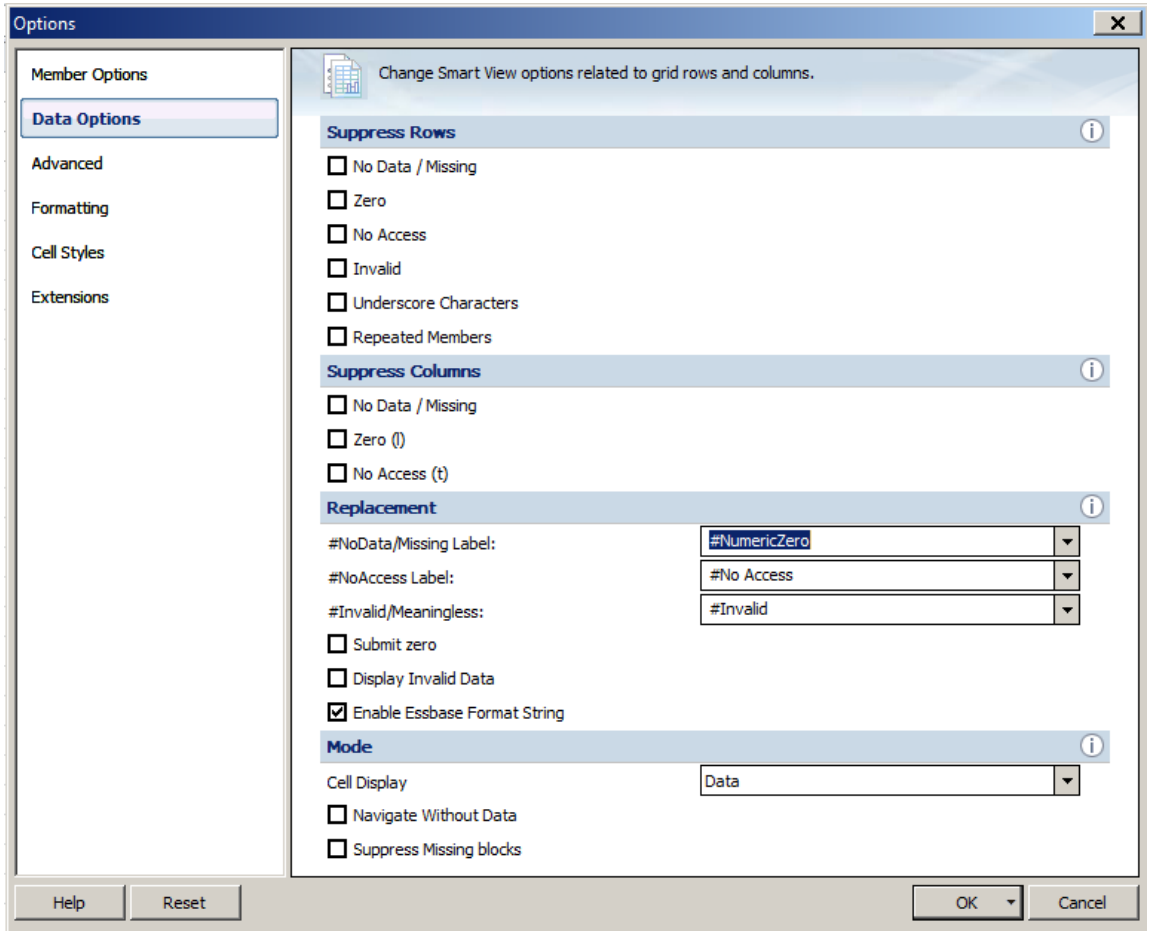


	A	B	C	D	E	F	G	H	I	J	K
1		Product	Market								
2	Profit	Jan	#Missing					0			
3		Feb	#Missing								
4		Mar	#Missing								
5		Qtr1	#Missing								
6		Qtr2	27107								
7		Qtr3	27912								
8		Qtr4	25800								
9		Period	80819								
10	Inventory	Jan	117405								
11		Feb	116434								
12		Mar	115558								
13		Qtr1	117405								
14		Qtr2	119143								

Smartview Trick – (Continued)

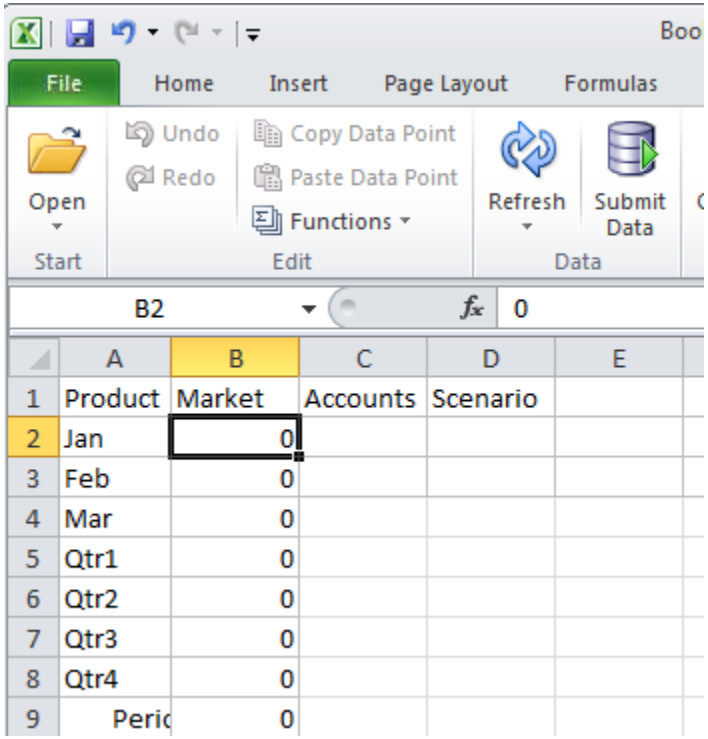
- Say we want a numeric zero to be returned from Essbase to complete some offline calculation
- This is also easy!
- We simply go into the SmartView and under OPTIONS\DATA OPTIONS in the Replacement section select the value #NumericZero in the #NoData/Missing Label dropdown list

Smartview Trick – (Continued)



Smartview Trick – (Continued)

- Now we have numeric zeros in our spreadsheet



Smartview Trick – (Continued)

- **Disclaimer** –This should not be used if you are sending data back to Essbase
- It will cause the database size to grow since you would now be storing data instead of #MISSING
- Empty consumes no space to store whereas a zero consumes the same space as 1 billion. Think of this as a grocery bag. If you fill a grocery bag with nothing, it takes up no space. If you fill it with empty cans (a zero), it consumes the same amount of space as if those cans were full (1 billion)
- So be careful using this trick

Summary

- Nested Fixes – limit passes through the database
- Conditional Fix – Using SET EMPTYMEMBERSETS
- Dense Fixes/Calculation Blocks
- Creating Blocks – Dealing with Block Creation issues
- Substitution Variables – Using them as Macros
- Cache settings – Increasing database performance
- Data export using Calculation scripts
- @Xwrite – Moving data between applications
- Varying Attributes - tracking slowly changing dimensions
- Typed Measures
- Smartview Trick - Converting #Missing to Zeros

Questions?

Thank You !

Philip Sable
Manager, Planning & Essbase Practice

75 Arlington St Suite 500 , Boston, MA 02116
Cell: (734) 377-7718
psable@MindStreamanalytics.com
www. MindStreamanalytics.com