interRel CONSULTING
Relevant Analysis. Relevant Solutions.

- **Oracle BI & EPM Tracks**
- Essbase
- Hyperion Planning & HFM
- Beginner to Guru Content
- OBIEE 11g
- Hyperion Developers On-Site
- **CONFERENCE HIGHLIGHTS**
- 250+ Sessions
- Hands-on Training
- Six full-day Symposiums

- **CHECK OUT OUR PRESENTATIONS**
- *Advanced Tips and Tricks Smart View*
- *Writing ASO Calc Scripts*
- *Advanced Tips and Tricks 11.1.2 Essbase Studio*
- *Hacking Essbase (2 hour session)*
- *Plus Seven More!*

- **USE THE SPECIAL interRel CODE IRC TO RECEIVE A $100 DISCOUNT ON REGISTRATION**

- **KSCOPE11.COM/BIEPM**

interRel CONSULTING

RELEVANT ANALYSIS. RELEVANT SOLUTIONS.

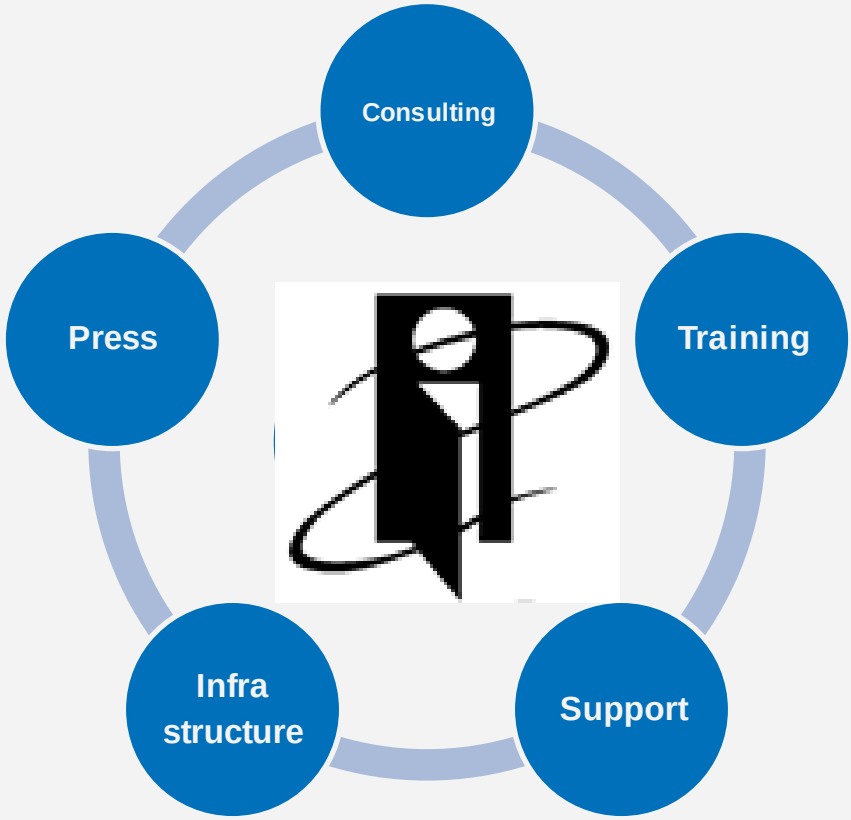
Top 10 Essbase Optimization Tips that Give You 99+% Improvements

Edward Roske
eroske@interrel.com
BLOG: LookSmarter.blogspot.com
WEBSITE: www.interrel.com
TWITTER: Eroske

About interRel

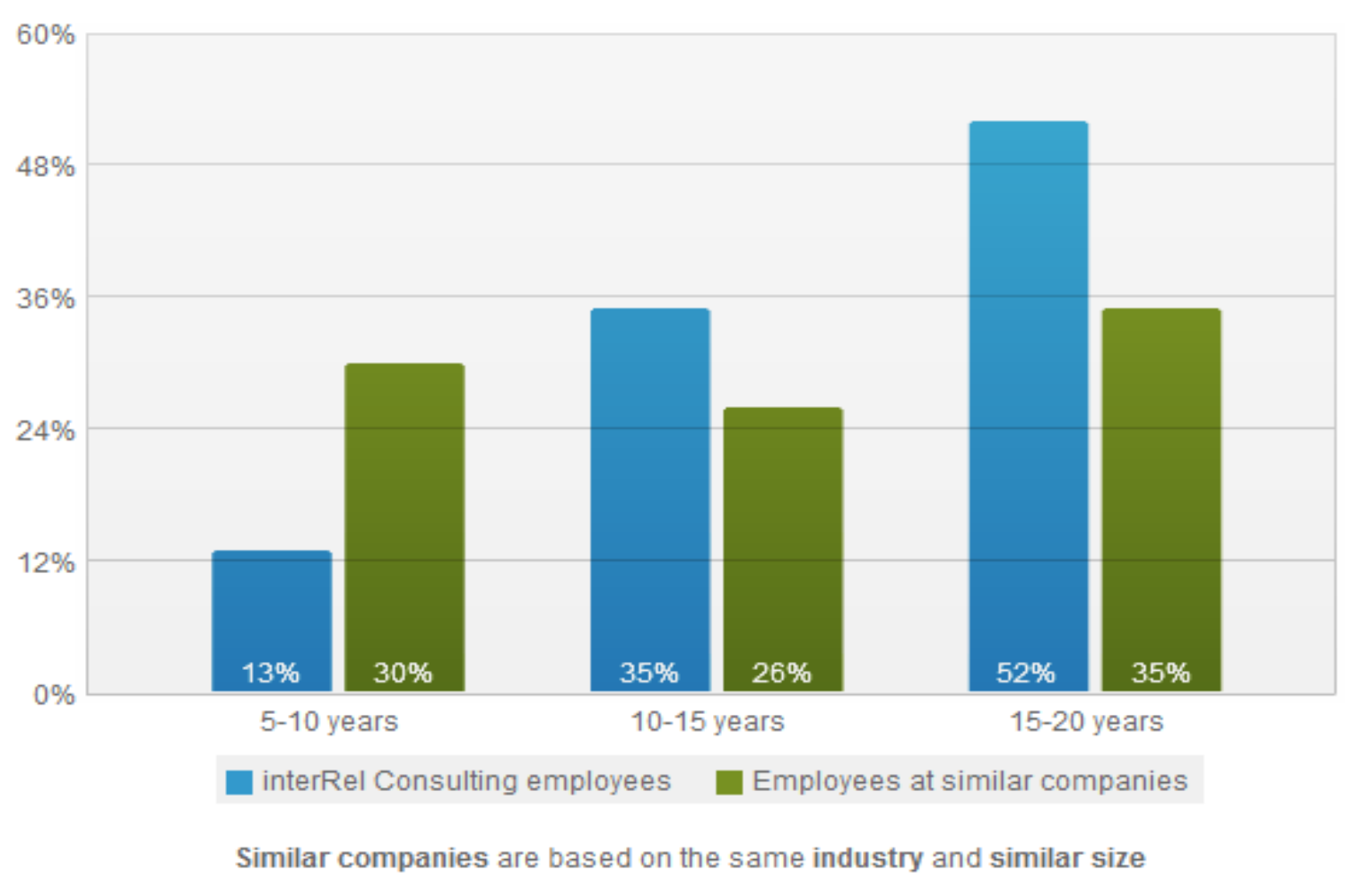
- **Reigning Oracle Award winner EPM & BI Solution of the year**
- ***Three*** Oracle ACE Directors for Hyperion
- Oracle Platinum Partner
- One of the 100 fastest growing tech companies in the USA (CRN Magazine, 2007-2010)

- Authors of the 8 Best Selling books on Hyperion & Essbase
- Essbase Studio book to be released April 11, 2011
- All available on LuLu.com

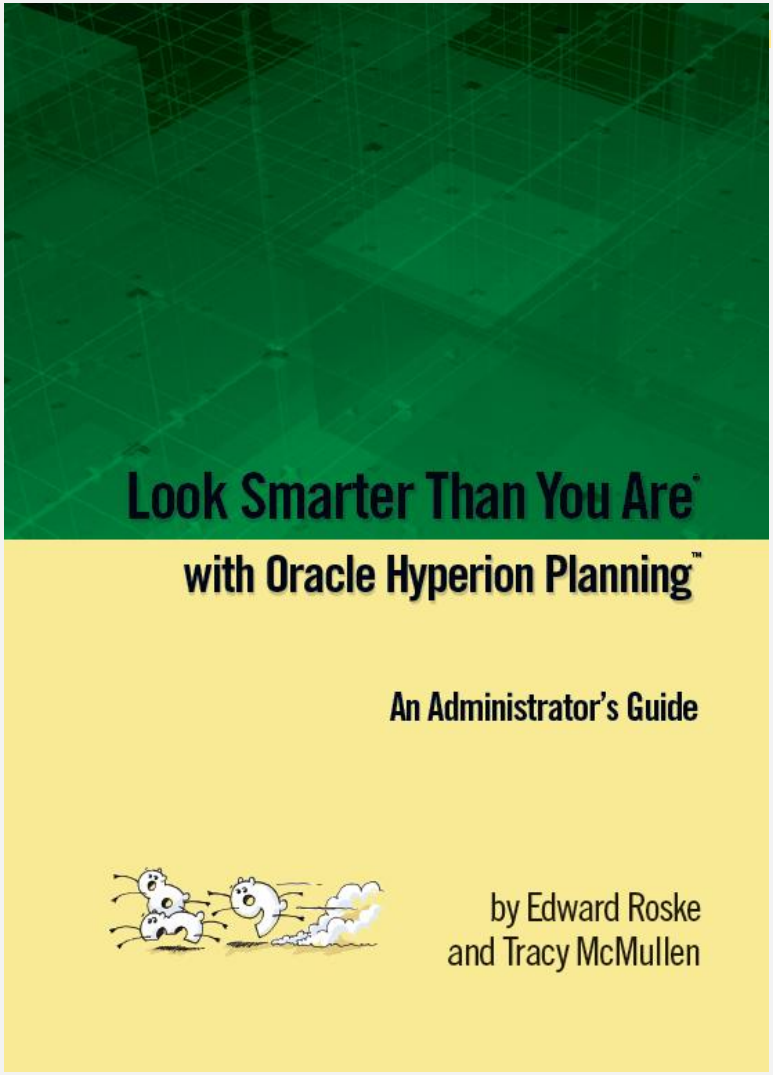


Focused exclusively on Oracle EPM & BI

2011 LinkedIn.Com Analysis of Years of Work Experience



Look for yourself at http://www.linkedin.com/company/interrel-consulting/statistics



8 Hyperion Books Available:

- Essbase (7): Complete Guide
- Essbase System 9: Complete Guide
- Essbase System 9: End User Guide
- Essbase 11: Admin Guide
- ***Essbase Studio 11***
- Smart View 11: End User Guide
- Planning: End Users Guide
- Planning: Administrators

To order, check out www.LuLu.com

Disclaimer

- These slides represent the work and opinions of the presenter and do not constitute official positions of Oracle or any other organization.
- This material has not been peer reviewed and is presented here with the permission of the presenter.
- This material should not be reproduced without the written permission of interRel Consulting

Top 10 Essbase Optimization Tips

1. Effective settings of Dense and sparse for BSO (BSO)
2. Order the dimensions in the outline (BSO)
3. Effective settings for Stored and Dynamic hierarchies (ASO)
4. Set data and index caches (BSO)
5. Set aggregate storage cache - 32MB for every 2 GB of data size (ASO)
6. Calc scripts - only calculate the dimensions and members required (BSO)
7. Defrag your BSO Databases
8. Effective design of member stores (stored, dynamic calc, label only) (BSO)
9. Use aggregations - with query hints, hard and soft restrictions (ASO)
10. Faster loads with load buffers, concurrent loads, incremental loads and partial clears (ASO)

Top 10 Essbase Optimization Tips

10. Faster loads with load buffers, concurrent loads, incremental loads and partial clears (ASO)

ASO Load Process

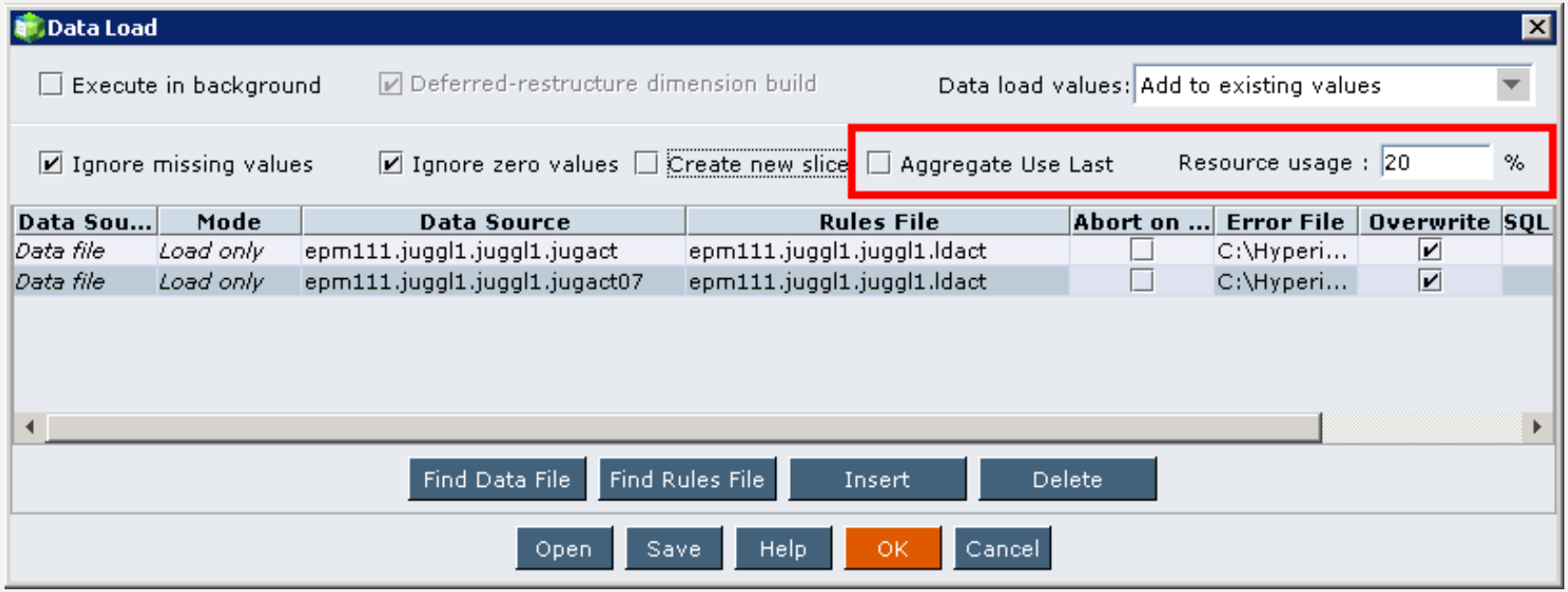
- Behind the scenes, the ASO data load process is a bit different from the loading of block storage databases
- Data sources include text file or relational database
- Can load one or more data sources
- Can load with or without rules files
- If multiple sources are used, choose
  - Overwrite existing values
  - Add to existing values
  - Subtract from existing values
- Because the data files are potentially very large, a temporary load buffer is used
  - Loading a single load file does not involve the buffer

Prepare for the Data Load

- Don't include fields that are applicable to BSO (they will be ignored)
- Load level zero data only
- If #Missing is specified, the cell will be removed from the database
- Don't worry about sorting*
  - The load buffer will sort and accumulate values
- Currency name and currency category are not supported

*Presorting multiple data source loads will improve performance even though the buffer always sorts: A sort-sort-merge-sort will be faster than a nonsort-nonsort-merge-sort. Not as dramatic a difference as BSO, but still important on very large loads and time critical incrementals.

Use Load Buffer with Multiple Files



- Aggregate Use Last option
- Resource Usage

Load Data with MaxL – Multiple Data Sources

- Using the load buffer during incremental data loads improves performance
- Initialize the load buffer to accumulate the data
  - alter database AsoSamp.Sample initialize load_buffer with buffer_id 1;
- Read the data sources into the load buffer
  - import database ASOSamp.Sample data from server data_file 'file_1' to load_buffer with buffer_id 1 on error abort;
  - import database ASOSamp.Sample data from server data_file 'file_2' to load_buffer with buffer_id 1 on error abort;
  - import database ASOSamp.Sample data from server data_file 'file_3' to load_buffer with buffer_id 1 on error abort;
- Load data from the buffer into the database
  - import database ASOSamp.Sample data from load_buffer with buffer_id 1;

Load Data with MaxL – Multiple Data Sources

- Import statements do not need to be continuous
- As long as the buffer exists, the database is locked from queries, aggregations and data loads by other means

Concurrent Loads

- Multiple load buffers can exist on an ASO database
- Load data simultaneously using multiple data buffers
- Commit multiple data load buffers in the same operation
  - Faster than committing each buffer by itself
- Must use separate sessions in MaxL

Concurrent Data Loads

- MaxL Session 1

```
alter database AsoSamp.Sample
initialize load_buffer with buffer_id 1 resource_usage 0.5;
import database AsoSamp.Sample data
from data_file "dataload1.txt"
to load_buffer with buffer_id 1
on error abort;
```

- MaxL Session 2

```
alter database AsoSamp.Sample
initialize load_buffer with buffer_id 2 resource_usage 0.5;
import database AsoSamp.Sample data
from data_file "dataload2.txt"
to load_buffer with buffer_id 2
on error abort;
```

- When data fully loaded, one commit statement

```
import database AsoSamp.Sample data
from load_buffer with buffer_id 1, 2;
```

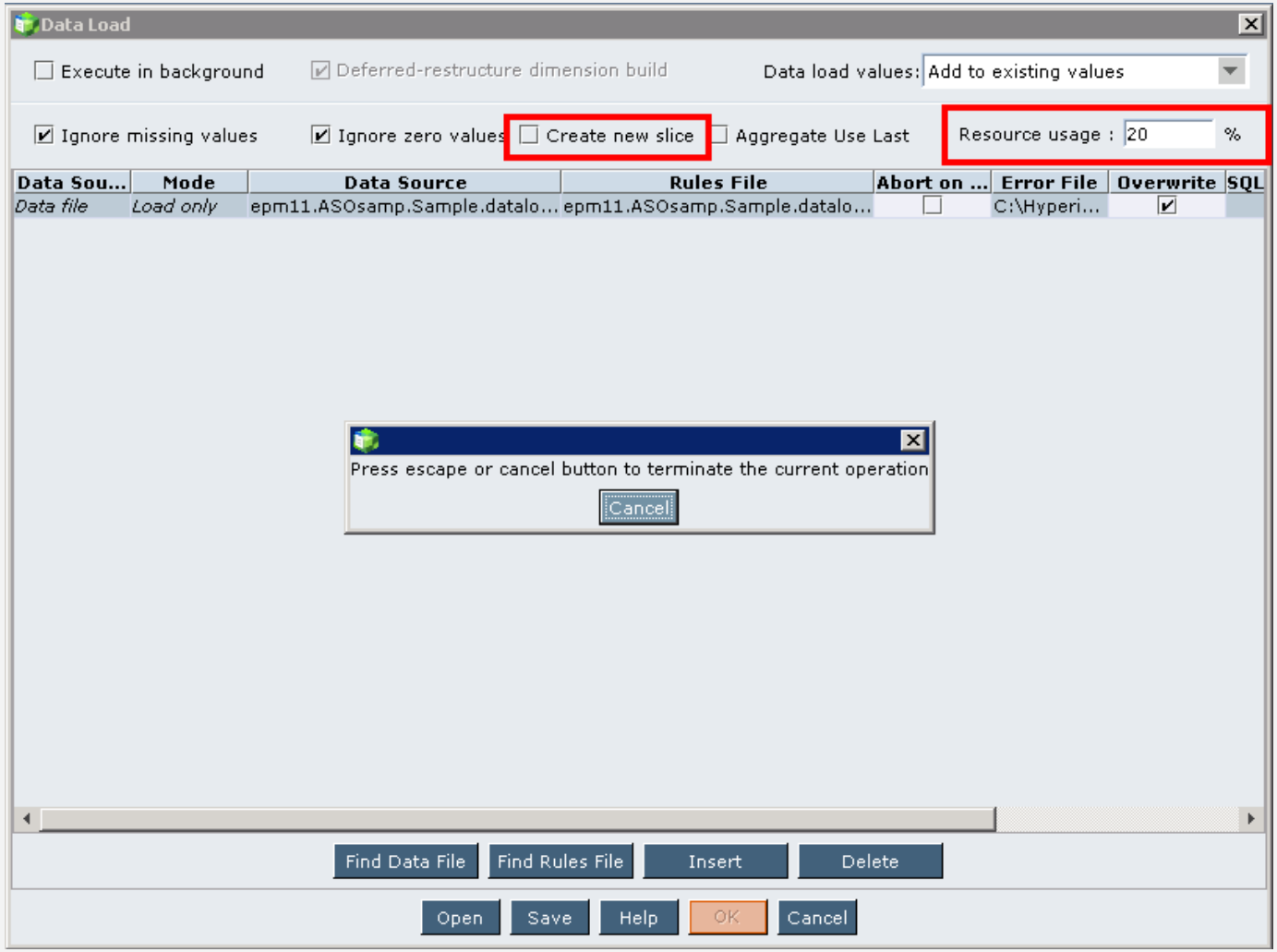
Slice Loading = Incremental Loads

- Enables “trickle feed” functionality
- Issue - Users perform retrievals when the database was being loaded and ASO loads can take a while
- Incremental loading creates subcubes or slices along side the primary slice of the database
- Dynamic aggregations are performed across the necessary slices to provide query results
- Different materialized views might exist within a slice as compared to the primary slice of the database

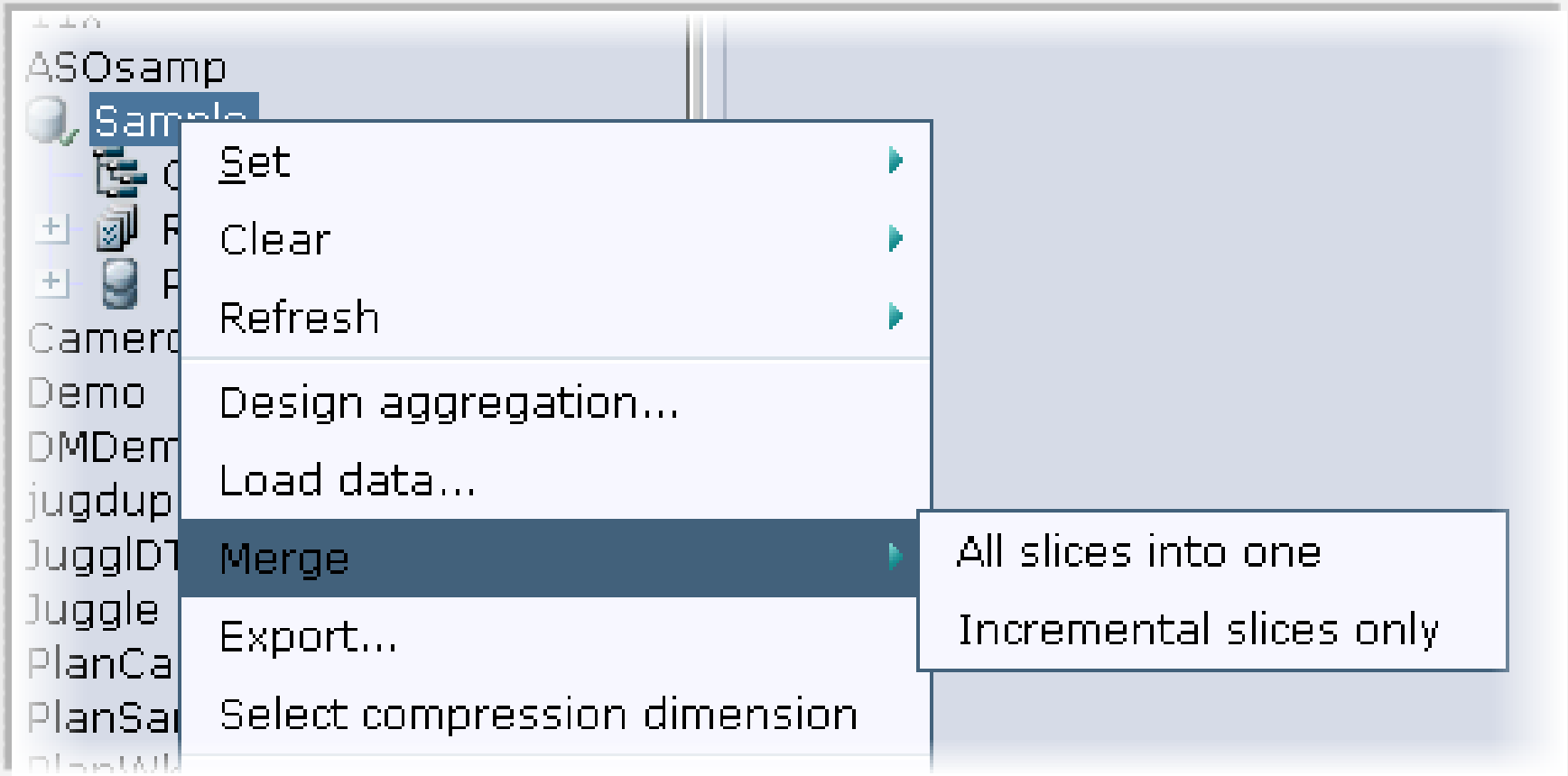
Incremental Data Loads

- Incremental data load time is proportional to the size of the incremental data
- Options
  - Merge all incremental slices into the main database slice or
  - Merge incremental slices into a single data slice, leaving the main db slice unchanged
- Load data into multiple data load buffers at the same time

Incremental Loads



Merging Slices



Partial Data Clear

- New feature in 11x
- Why?
  - Need to clear and modify selected portions of the cube
  - Refresh actuals but budget is static
- Physical clear completely removes cells
  - Longer to clear
  - Faster retrievals after aggregation
- Logical clear removes cells by creating compensating cells in a new slice
  - Faster to clear

Top 10 Essbase Optimization Tips

9. Use aggregations - with query hints, hard and soft restrictions (ASO)

10. Faster loads with load buffers, concurrent loads, incremental loads and partial clears (ASO)

ASO Aggregations

- Data doesn't have to be aggregated but aggregations can speed up retrievals
- Maximum of 1024 aggregations
- Aggregates an entire level/slice not specific intersections
- Default
  - Doesn't guarantee best performance

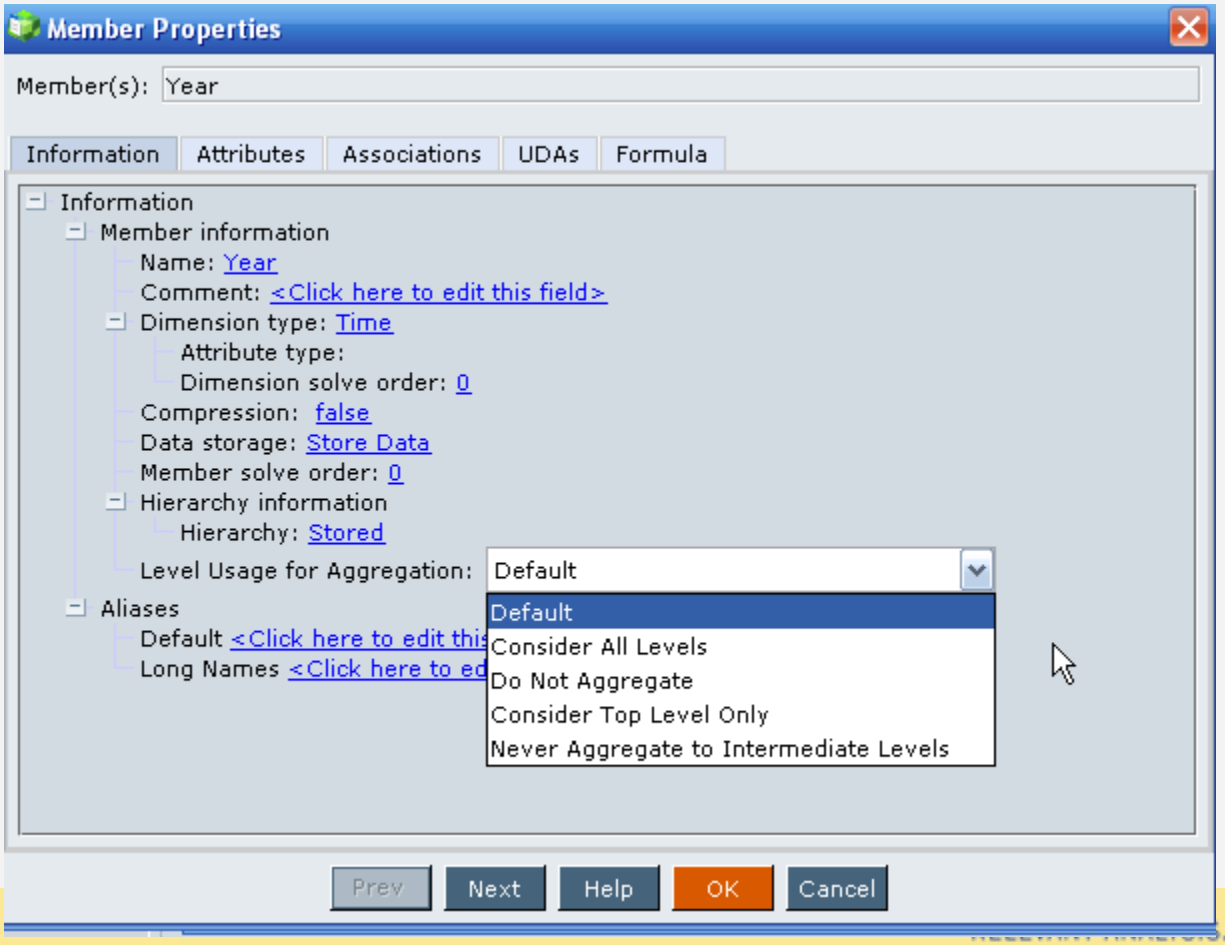
- Query Tracking
  - To determine specific materializations, turn on query tracking and then do a single query
  - Tracks the requests from users and uses this information to create aggregate views

Intelligent Aggregations=Hard Restrictions

- You can now define Hard restrictions for a dimension
  - Default (no restriction for primary hierarchy, no aggregation for alternate hierarchies)
  - Consider all levels
  - Do not aggregate
  - Consider top level only
    - (you only query top level)
  - Never aggregate to intermediate levels
    - (you only query level zero or top dimension)
- Level based weighting – provide levels to consider

Intelligent Aggregations for ASO

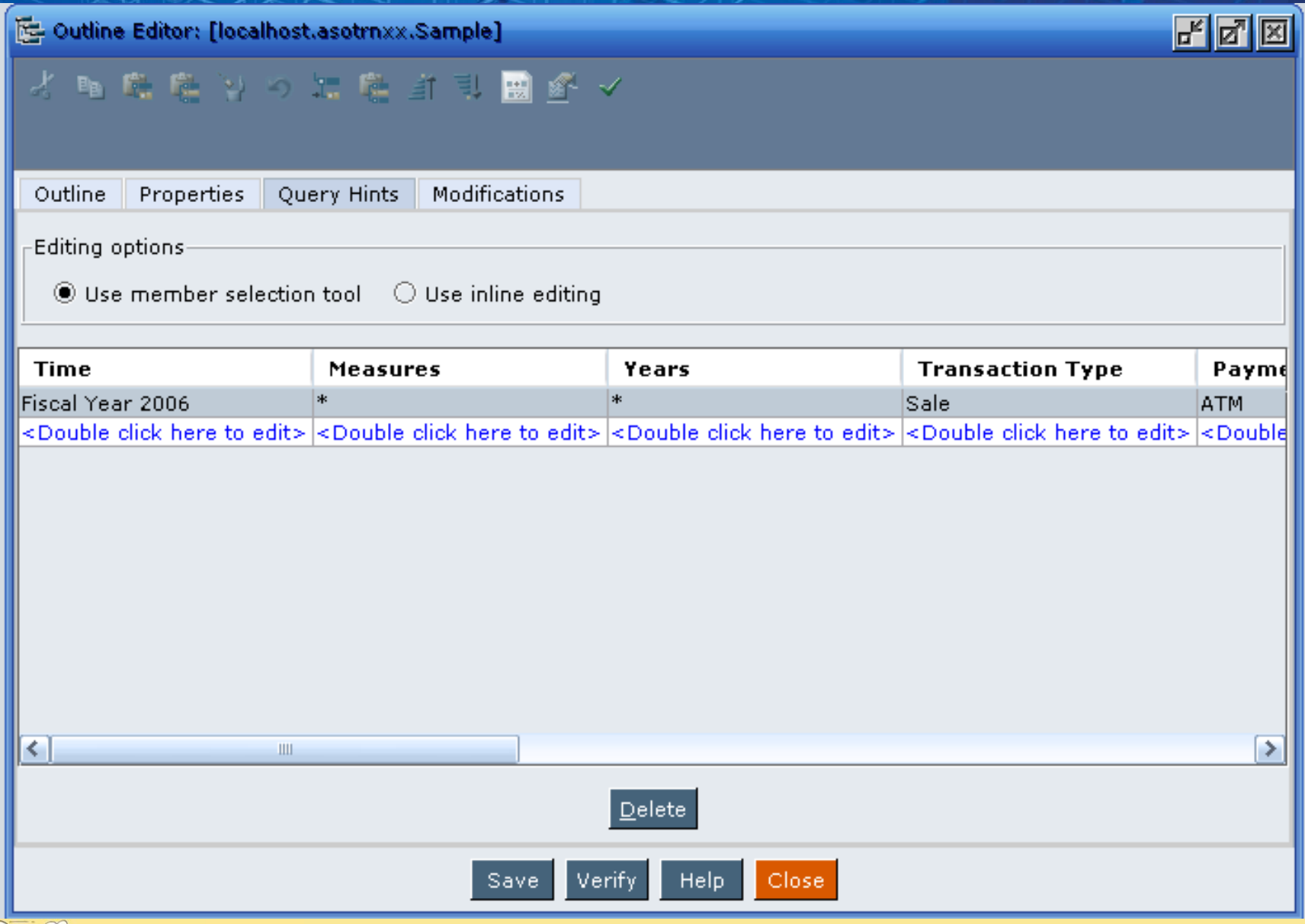
- EAS interface allows user to specify outline constraints:



Query Hints = Soft Restrictions

- You can define “soft restrictions” as a query hint
- Just select a representative member (any member)
- Essbase will take this into consideration when creating aggregation views

Query Hints


Outline Editor: [localhost.asotrnxx.Sample]
Outline
Properties
Query Hints
Modifications
Editing options
Use member selection tool
Use inline editing
Time
Measures
Years
Transaction Type
Payme
Fiscal Year 2006
*
*
Sale
ATM
<Double click here to edit>
<Double click here to edit>
<Double click here to edit>
<Double click here to edit>
<Double
Delete
Save
Verify
Help
Close

Top 10 Essbase Optimization Tips

8. Effective design of member stores (stored, dynamic calc, label only) (BSO)

9. Use aggregations - with query hints, hard and soft restrictions (ASO)

10. Faster loads with load buffers, concurrent loads, incremental loads and partial clears (ASO)

Dynamic Calculations

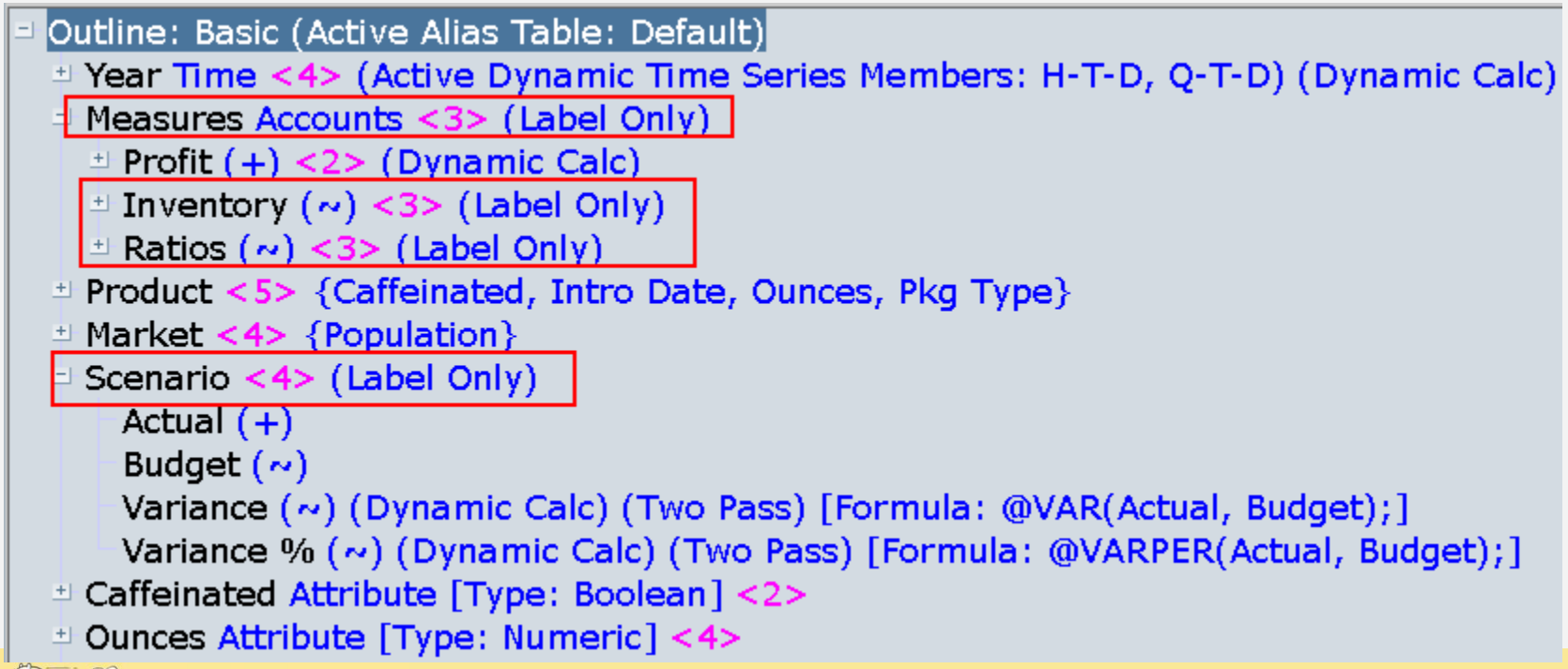
- Dynamic calc members are evaluated during retrievals
- You can reference dynamic calc members
- Watch out for dynamic calc members on different dimensions
  - Sparse will calculate before dense
  - Within Dense, it's outline order
  - Two-Pass is last (sparse before dense)
- Watch out for dynamic calcs that are dependent on other dynamic calcs

Dynamic Calculations
Advantages - Dynamic Calc (Non-Store)

- Use:
  - No impact on pre-calculation window
  - Used to retrieve infrequently accessed information or to save space
  - Used when data changes often and changes need to be reflected instantly
  - Reduce block size (for dynamic calcs on dense dimensions)
- Best Choice for:
  - Relatively simple calculations
  - Small queries that do not span large numbers of data cells
  - Infrequently accessed data

Label Only

- When should I tag a member label only?
  - For all “navigation” members
  - Those members that you use to organize the outline
  - No data is stored



Top 10 Essbase Optimization Tips

7. Defrag your BSO Databases

8. Effective design of member stores (stored, dynamic calc, label only) (BSO)
9. Use aggregations - with query hints, hard and soft restrictions (ASO)
10. Faster loads with load buffers, concurrent loads, incremental loads and partial clears (ASO)

Fragmentation

- Fragmentation can be a potentially crippling side-effect of frequently updated databases that use one of the compression techniques mentioned earlier
- Let's assume that we have a very simple block with only eight cells:

100	#Missing	#Missing	#Missing
#Missing	#Missing	#Missing	#Missing

Fragmentation

- Any one of the compression methods would work well on this, but for the sake of example let's assume Run-Length Encoding is used and is able to compress the data storage component to 32 bytes (8 Bytes for the 100 and 24 Bytes to compress all the #Missing values together)
- Then, a user writes some budget data to this block:

100	150	200	#Missing
#Missing	#Missing	#Missing	#Missing

Fragmentation

- This block will now require 48 Bytes to store (8 Bytes for each number, and 24 Bytes for the #Missing values)
- Fragmentation happens because Essbase can't fit 48 Bytes back into the original 32 Byte location and it is written to the end of the file
- The original block still remains in the file, but there is no corresponding pointer in the index file so it is lost forever but still taking up space

Frequent Causes of Fragmentation

- Read/write databases where users constantly update data
- Execute calcs around the clock
- Frequent updates and recalc's of dense members
- Poorly designed data loads
- Large number of Dynamic Calc and Store members
- Isolation level of uncommitted access with commit block = zero

Remove Fragmentation

- Perform an export of the database, delete all data in the database with CLEARDATA, and reload the export file
- Force a dense restructure of the database
- Also through MaxL
- Also by right-clicking in Admin Services & choosing “Restructure”

Top 10 Essbase Optimization Tips

6. **Calc scripts - only calculate the dimensions and members required (BSO)**
7. Defrag your BSO Databases
8. Effective design of member stores (stored, dynamic calc, label only) (BSO)
9. Use aggregations - with query hints, hard and soft restrictions (ASO)
10. Faster loads with load buffers, concurrent loads, incremental loads and partial clears (ASO)

Only Calc the Necessary Dimensions

- Don't use Calc All (in most cases)
  - Will do all dimensions whether they need to be calced or not
  - Can't control dimension aggregation
- Use
  - Calc Dim
    - Will look for formulas
    - Can be used on dense dimensions, but you shouldn't be calcing dense dimensions anyway
  - Agg
    - Only on sparse and no formulas, which is what you should have
    - May need CALC DIM on deep, sparse dimensions

Only Calc the Necessary Dimensions

- Assume
  - Dimensions: Period, Accounts, Scenario, Products, Customers
  - All upper levels of period are dynamically calculated
  - Accounts has some stored upper level members and formulas
  - Scenario contains 2 stored members (Actual, Budget) and 1 dynamic calc member (Variance)
  - All other dimensions are sparse with no member formulas

```
Set Updatecalc Off;

Calc Dim (Accounts);
Agg (Customer, Products);
```

Top 10 Essbase Optimization Tips

5. **Set aggregate storage cache - 32MB for every 2 GB of data size (ASO)**
6. Calc scripts - only calculate the dimensions and members required (BSO)
7. Defrag your BSO Databases
8. Effective design of member stores (stored, dynamic calc, label only) (BSO)
9. Use aggregations - with query hints, hard and soft restrictions (ASO)
10. Faster loads with load buffers, concurrent loads, incremental loads and partial clears (ASO)

Aggregate Storage Cache

- Memory used to speed up data loads, aggregations and retrievals
- 32 MB default cache size supports 2 GB of level zero data
- Use the size of input level data (level 0 data) to determine cache size
- Cache incrementally increases until the maximum cache size is used or OS denies additional allocations
- Cache size is affected by the number of application threads
- View cache statistics in Administration Services console under Database Properties
- Change Cache settings by Administration Services console or MAXL

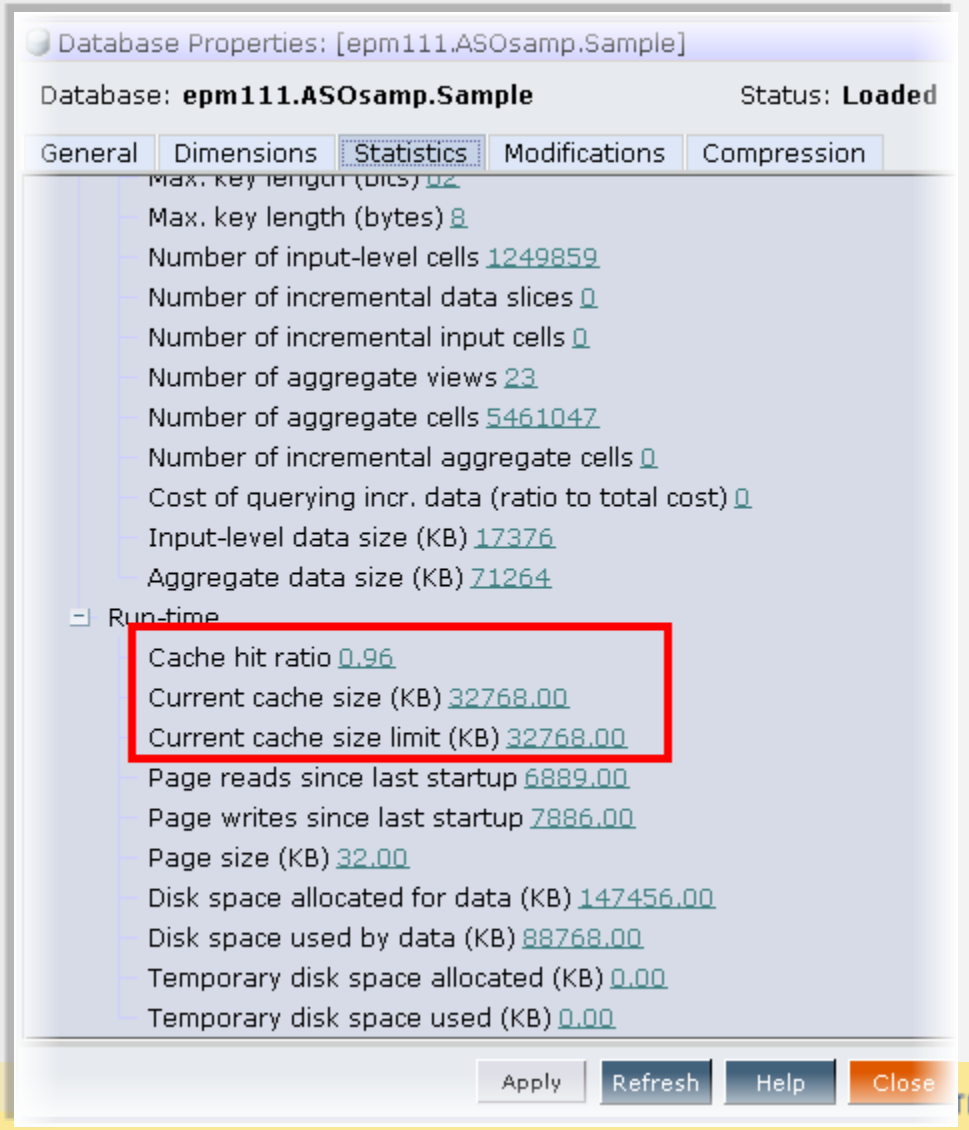
Parallel Calculation Threads

- Essbase uses multiple threads to divide the aggregate storage cache during materialization
- If you increase the number of threads specified in CALCPARALLEL config settings, then increase the aggregate storage cache
  - Increase by the same factor as the threads

Set Aggregate Storage Cache

Aggregate Cache Statistics

- Cache Hit Ratio
- Current Cache size
- Current Cache size limit

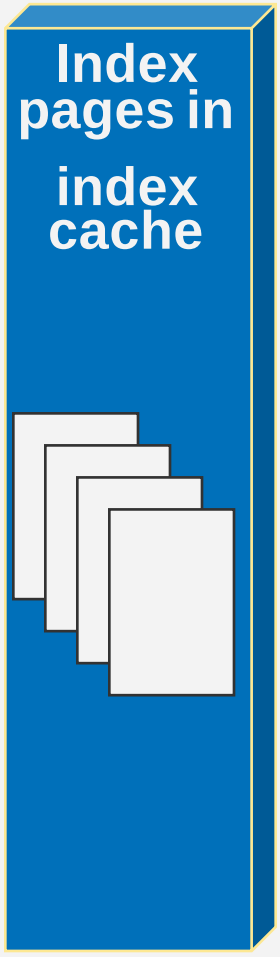


Top 10 Essbase Optimization Tips

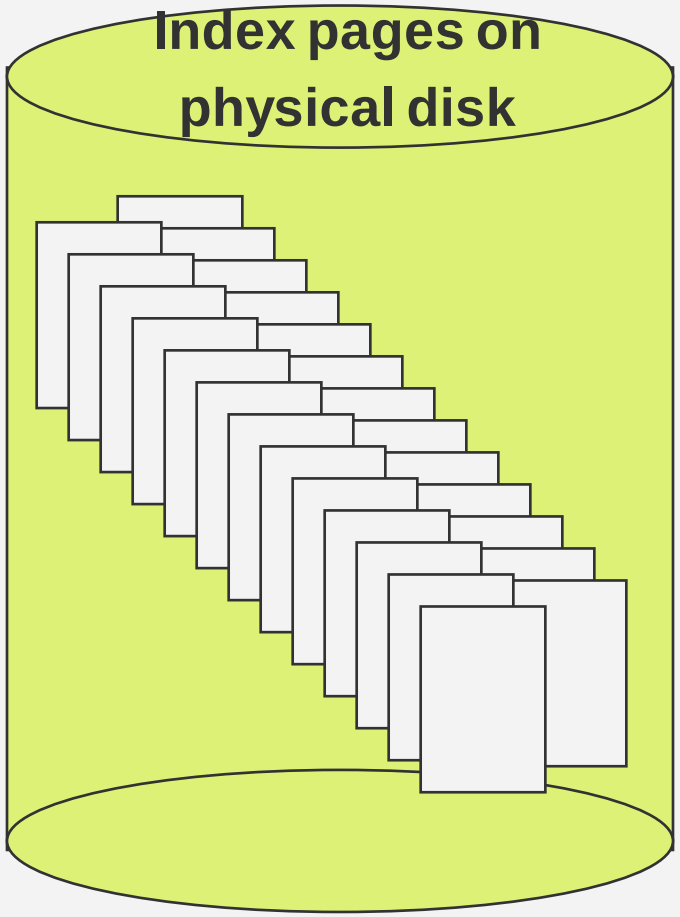
4. Set data and index caches (BSO)

5. Set aggregate storage cache - 32MB for every 2 GB of data size (ASO)
6. Calc scripts - only calculate the dimensions and members required (BSO)
7. Defrag your BSO Databases
8. Effective design of member stores (stored, dynamic calc, label only) (BSO)
9. Use aggregations - with query hints, hard and soft restrictions (ASO)
10. Faster loads with load buffers, concurrent loads, incremental loads and partial clears (ASO)

Index Cache

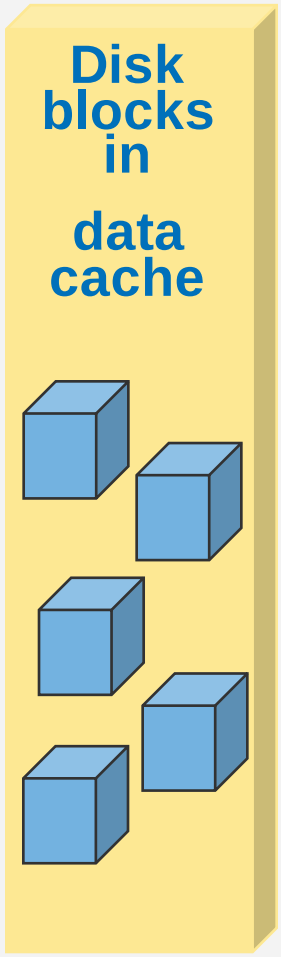


RAM

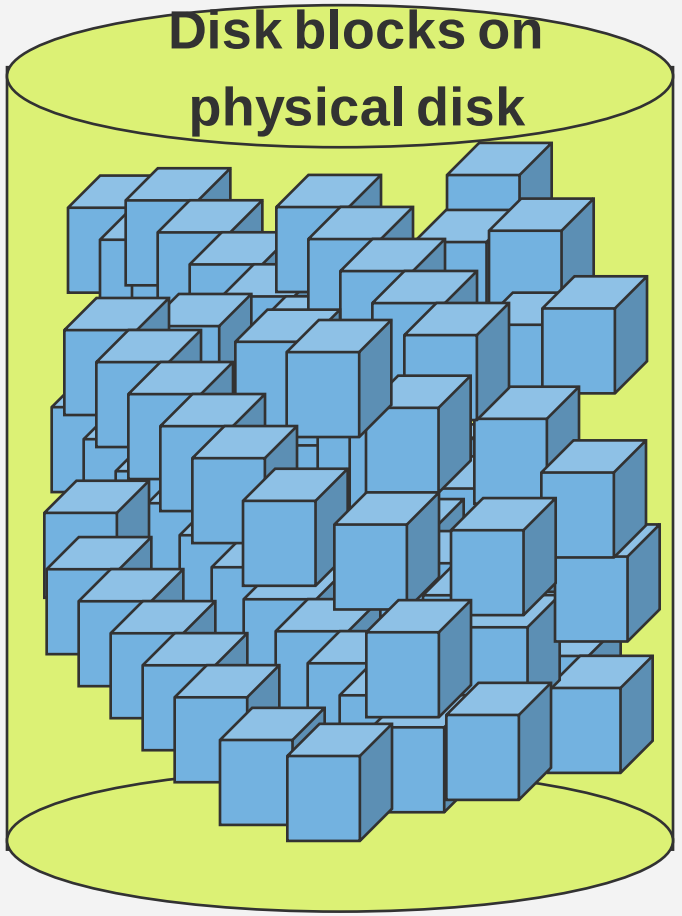


Disk

Data Cache

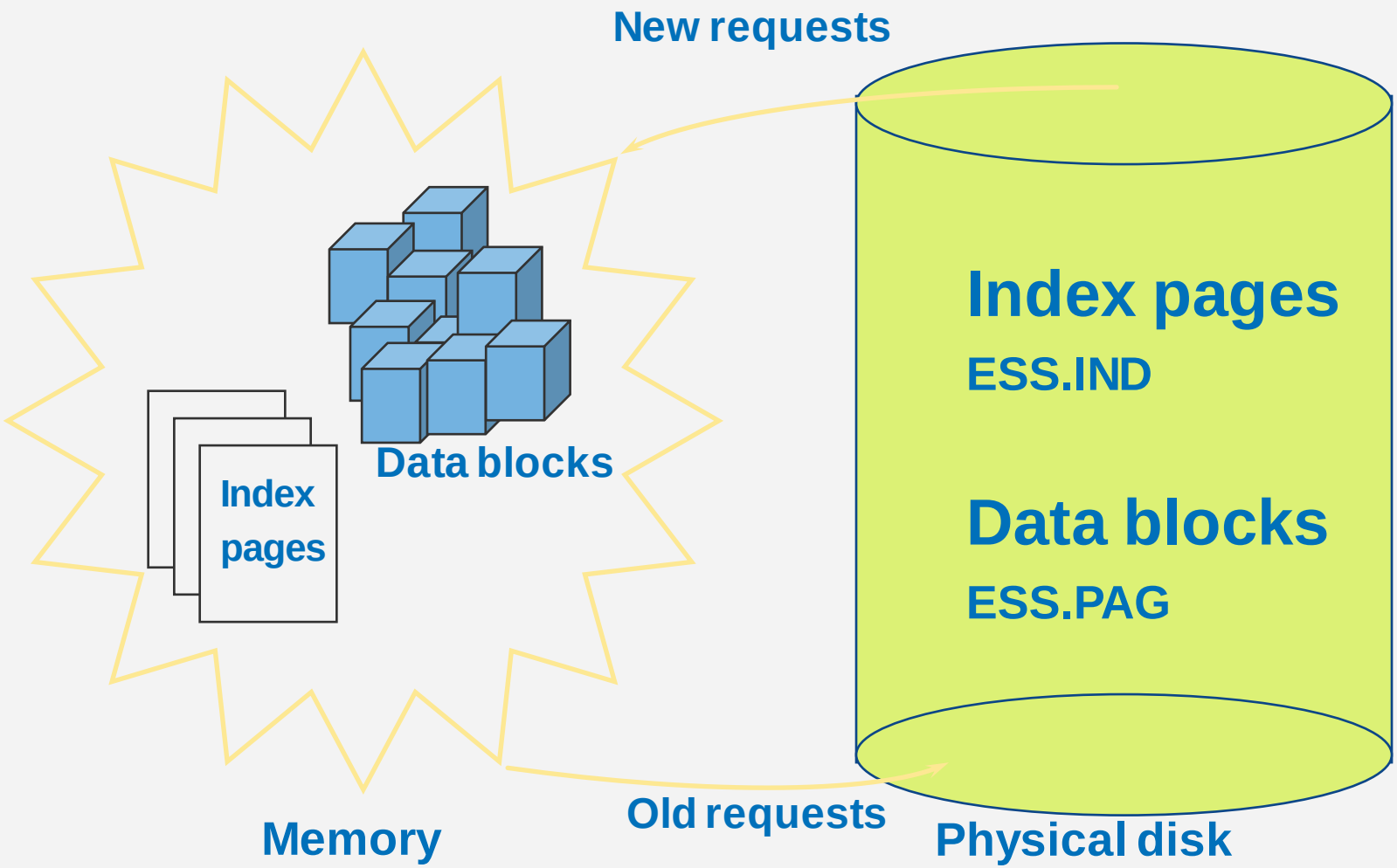


RAM



Disk

Paging


New requests
Index pages
ESS.IND
Data blocks
ESS.PAG
Data blocks
Index
pages
Memory
Old requests
Physical disk

Factors Affecting Cache Sizing

- Database size
- Block size
- Index size
- Available memory
- Data distribution
- Sparse / dense configuration
- Needs of database (e.g. complexity of calculations)

Priority for Memory Allocation

1. Index Cache
2. Data File Cache
3. Data Cache

Guideline for Index Cache

- Default
  - Buffered I/O: 1024 KB (1048576 bytes)
  - Direct I/O: 10240 KB (10485760 bytes)
- Guideline:
  - 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, as no performance improvement results

Guideline for Data Cache

- Default
  - 3072 KB (3145728 bytes)
- Guideline
  - 0.125 * Combined size of all essn.pag files, if possible; otherwise as large as possible
  - 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)
    - For data load, the number of threads specified by the DLTHREADSWRITE setting is very high and the expanded block size is large

Cache Hit Ratios

- Hit Ratios evaluate how well caches are being utilized
- Indicates the percentage of time that a requested piece of information is available in the cache
- Higher the better
- Right click on the Database and select Properties. Navigate to the Statistics tab in Administration Services to view hit ratios
- Index Cache Hit Ratio setting indicates the success rate in locating index information in the index cache without having to retrieve another index page from disk
  - Goal = 1

Top 10 Essbase Optimization Tips

3. Effective settings for Stored and Dynamic hierarchies (ASO)

4. Set data and index caches (BSO)
5. Set aggregate storage cache - 32MB for every 2 GB of data size (ASO)
6. Calc scripts - only calculate the dimensions and members required (BSO)
7. Defrag your BSO Databases
8. Effective design of member stores (stored, dynamic calc, label only) (BSO)
9. Use aggregations - with query hints, hard and soft restrictions (ASO)
10. Faster loads with load buffers, concurrent loads, incremental loads and partial clears (ASO)

Dimension / Hierarchy Types

- Stored
  - Best used for speed of aggregating big dims
  - Only + aggregations
  - No formulas
  - Accounts can't be Stored
- Dynamic
  - Calculated not aggregated
  - All calcs done at retrieval time
  - Multiple consolidation symbols
  - Formulas (MDX)
  - Not part of Aggregate View Selection

Dimension / Hierarchy Types

- Multi-Hierarchy
  - Stored or Dynamic
  - Non shared member must occur in the OTL before shared members
  - First hierarchy in multiple can't contain shared

Accounts Dimension

- Accounts
  - Dynamic dimension
  - Non additive unary operators
  - If you need time balance
  - Expense tags accomplished through UDAs
  - Default compression dimension
    - Compression on this dimension (any dimension allowed beginning in 9.3)

Top 10 Essbase Optimization Tips

2. Order the dimensions in the outline (BSO)

3. Effective settings for Stored and Dynamic hierarchies (ASO)
4. Set data and index caches (BSO)
5. Set aggregate storage cache - 32MB for every 2 GB of data size (ASO)
6. Calc scripts - only calculate the dimensions and members required (BSO)
7. Defrag your BSO Databases
8. Effective design of member stores (stored, dynamic calc, label only) (BSO)
9. Use aggregations - with query hints, hard and soft restrictions (ASO)
10. Faster loads with load buffers, concurrent loads, incremental loads and partial clears (ASO)

Old School

- Hourglass
- Accounts first, Time second
- Dynamic calcs always slow retrievals

Dimension Ordering Guidelines

- Time (turn on RLE compression)
- Accounts
- Largest Dense Dimensions
- Smallest Dense Dimensions
- Smallest Aggregating Sparse Dimensions
- Largest Aggregating Sparse Dimensions
- Non-aggregating Sparse Dimensions

- Just a starting point… test to figure out the most optimal setting for your database
  - May change if you are tuning for calc performance vs. retrievals

Optimized Dimension Order

Typical Hourglass →

Original	Optimized
Accounts (D)	Time Periods (D)
Time Periods (D)	Accounts (D)
Years	Job Code (AS)
Versions	Organization (AS)
Scenarios	Years (NAS)
Job Code	Versions (NAS)
Organization	Scenarios (NAS)
Employee Status	Employee Status (Attr Dim)
Fund Group	Fund Group (Attr Dim)

Modified Hourglass ←

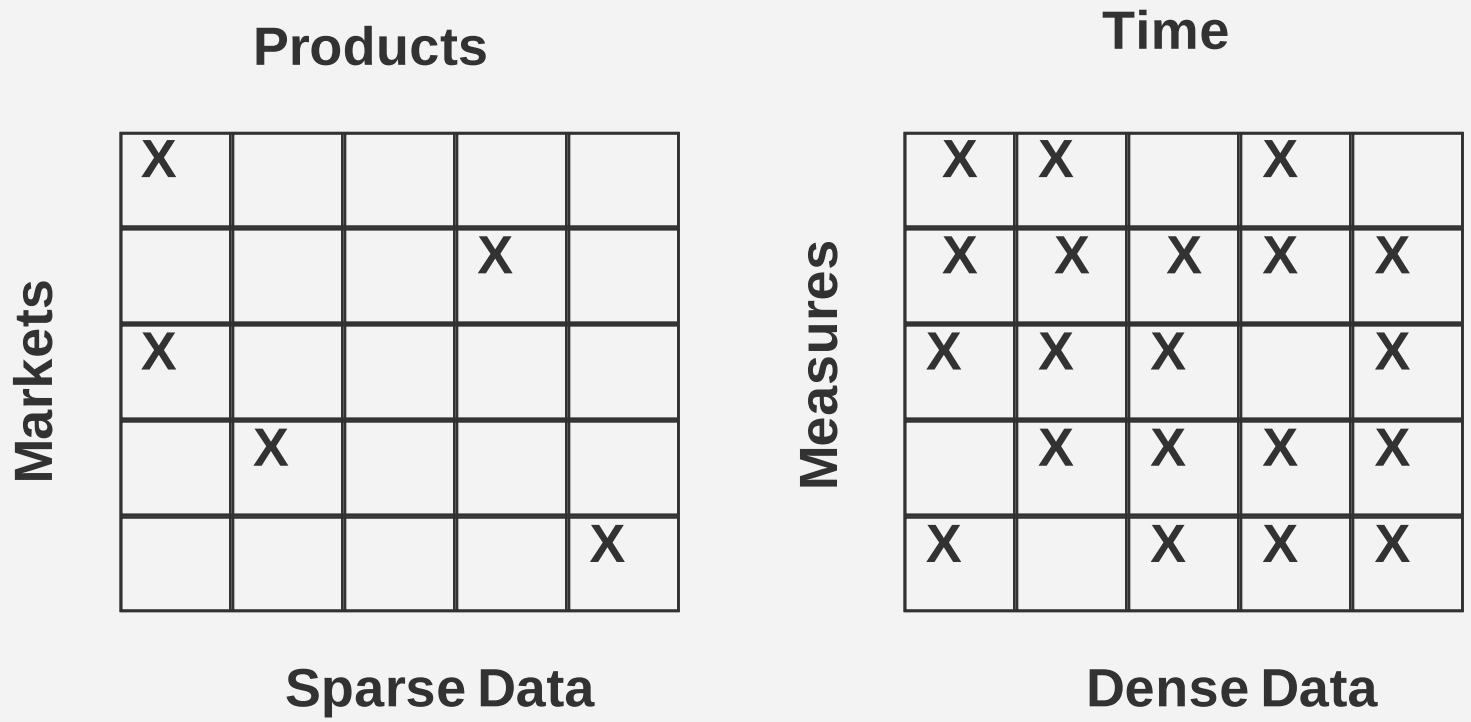
Consider Heavily Queried Dims

- Consider placing heavily queried dimensions up further in the outline

Top 10 Essbase Optimization Tips

1. **Effective settings of Dense and sparse for BSO (BSO)**
2. Order the dimensions in the outline (BSO)
3. Effective settings for Stored and Dynamic hierarchies (ASO)
4. Set data and index caches (BSO)
5. Set aggregate storage cache - 32MB for every 2 GB of data size (ASO)
6. Calc scripts - only calculate the dimensions and members required (BSO)
7. Defrag your BSO Databases
8. Effective design of member stores (stored, dynamic calc, label only) (BSO)
9. Use aggregations - with query hints, hard and soft restrictions (ASO)
10. Faster loads with load buffers, concurrent loads, incremental loads and partial clears (ASO)

Dense and Sparse

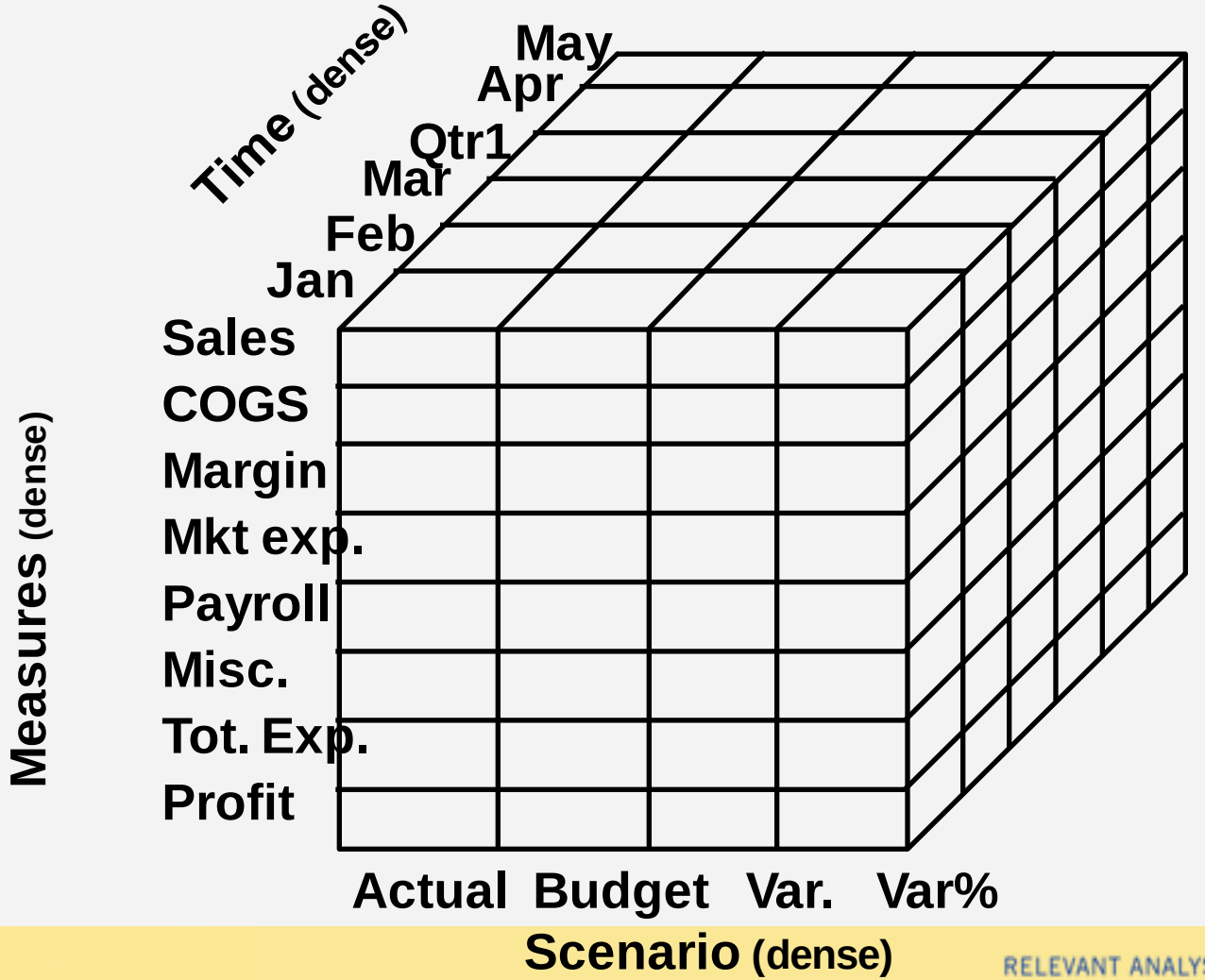


- Tagging dimensions as dense and sparse determines the overall size and structure of the database

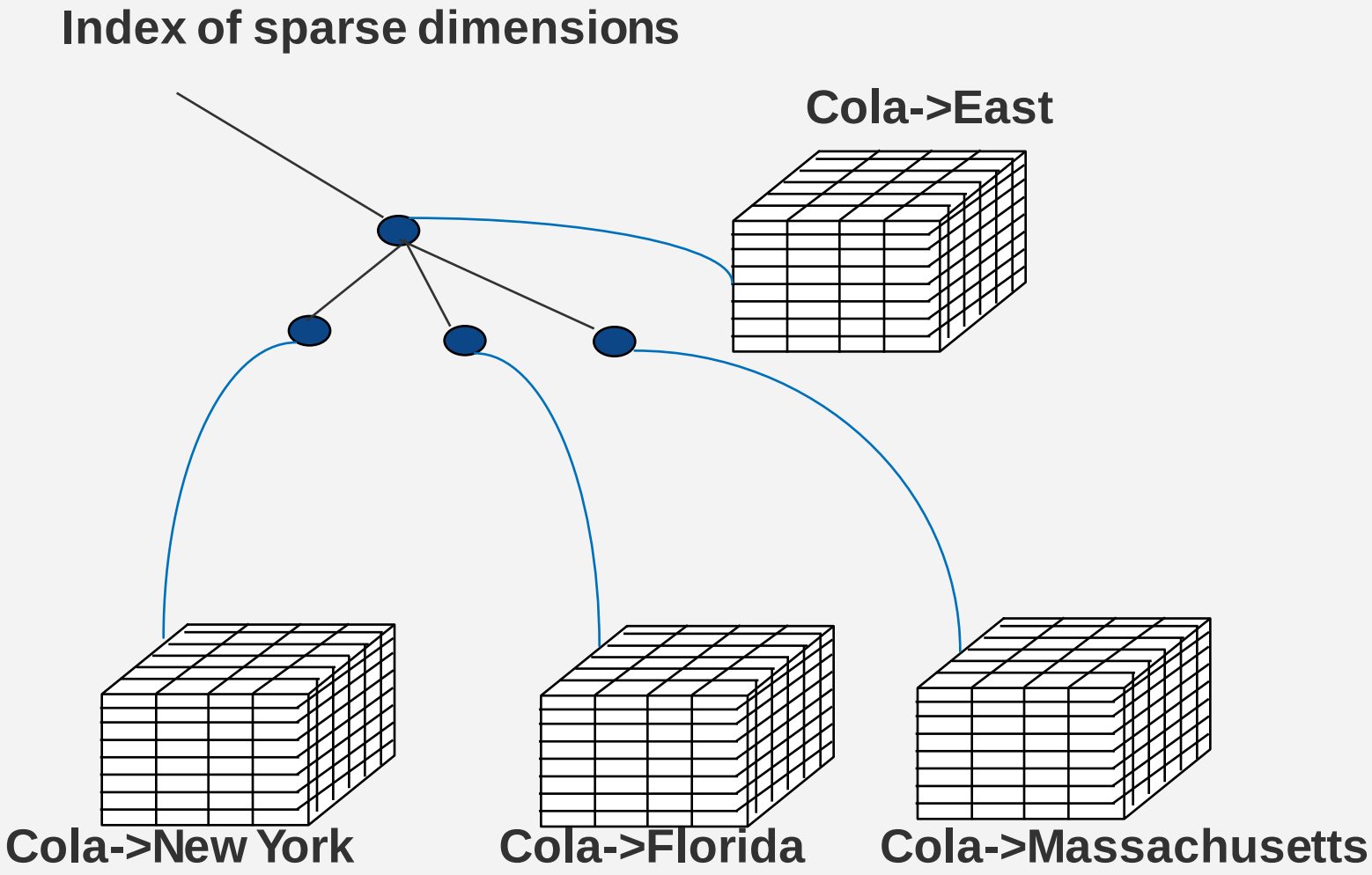
Blocks in Essbase

- Dimensions tagged dense will make up the cells within a block
- Dimensions tagged sparse will create the blocks themselves
  - Essbase creates a data block for each unique combination of sparse dimension members that contain data
- Tagging dimensions as dense and sparse determines the overall size and structure of the database

Dense Dimensions Make Up Cells within a Block



Sparse Dimensions Create Blocks



Dense and Sparse

- Look for small block sizes
  - If you ask the DBA Guide, 8 to 100 KB is the recommended block size*
    - If data blocks are smaller than 8K, the index is very large, forcing Essbase to write to and retrieve index from disk
    - If data blocks are larger than 100KB, intelligent calc does not work efficiently
  - If you ask Edward, keep the block size between 1 and 8Kb (for 32-bit) with closer to 8K the better (but smaller than 8K is better than bigger than 8K hence the 1-8Kb recommendation)

- **Choosing the dense and sparse dimensions and then ordering them in the outline is probably the biggest way you can effectively tune your Essbase BSO databases**

Top 10 Essbase Optimization Tips

1. Effective settings of Dense and sparse for BSO (BSO)
2. Order the dimensions in the outline (BSO)
3. Effective settings for Stored and Dynamic hierarchies (ASO)
4. Set data and index caches (BSO)
5. Set aggregate storage cache - 32MB for every 2 GB of data size (ASO)
6. Calc scripts - only calculate the dimensions and members required (BSO)
7. Defrag your BSO Databases
8. Effective design of member stores (stored, dynamic calc, label only) (BSO)
9. Use aggregations - with query hints, hard and soft restrictions (ASO)
10. Faster loads with load buffers, concurrent loads, incremental loads and partial clears (ASO)

interRel CONSULTING

RELEVANT ANALYSIS. RELEVANT SOLUTIONS.

Thank You!

eroske@interrel.com
WEBSITE: www.interrel.com