Evolution or Revolution – the new Hybrid Essbase

Cameron Lackpour
Tim German
Dan Pressman
Developing Essbase Applications

- Like the best, most advanced Essbase conference there ever could be
- Advanced content
- Good practices
- Written by some of the most well known Essbase developers
- Source code at www.developingessbasebook.com
- You should buy it
You’re Invited!

Join us November 4 & 5, 2014
Sheraton Imperial Hotel & Convention Center
Raleigh/Durham, North Carolina for the

East Coast Oracle Users Conference

CALL FOR PRESENTATIONS (Deadline July 14, 2014):
Share your knowledge by presenting at the 2014 conference. Presentations may be targeted to novice, intermediate, advanced, or all attendees and are scheduled for 60 minutes.

VENDOR OPPORTUNITIES (contact Melissa@eastcoastoracle.org)

Plan now to attend ECO 2014 – one of the largest and most economical Oracle training opportunities on the East Coast.

For details, to submit an abstract, or register visit www.eastcoastoracle.org.
Who are you?

- Sad BSO bitter enders who just won’t let go.
- Geeks who love BSO and know that its calc language cannot be beat.
- ASO bigots who can’t believe Oracle’s really merged the two.
- Interested neutral parties who use the best tool for the job, but wish they could just pick one.
What this presentation will not cover

- All of the other cool things that 11.1.2.3.500 supports
  - Faster MDX Aggregate and Sum functions
  - FIXPARALLEL
  - CALCPARALLEL with @XREF and @XWRITE
  - In-place data writing with Exalytics
Session purpose

- Technology exploration
- Hybrid as a statement of product direction
  - Hybrid *does* work
    - It *is* awesome
    - It *is* revolutionary
  - At the same time…
  - It is a 1.0 release (just about)
    - Limitations necessarily apply
- Hybrid good practices from testing only
- Where Hybrid is going in the near future
What this presentation will cover

- The history behind ASO, BSO, and now Hybrid
- What is Hybrid Essbase?
- Architecture as we understand it
- What’s supported, what’s not
- Using Hybrid
- Size and load testing
- Designing for Hybrid
- Potential use cases
- Hybrid futures
Just what is Hybrid Essbase?

- One giant step towards a single engine
- It is both BSO and ASO Essbase
  - “60% ASO, 40% something else”
  - BSO when stored
  - ASO-ish everywhere else
  - Hybrid functionality optional
Licensing

- Requires full use Essbase
  - BI price list identifies this as Essbase Plus
  - Cost? Speak with your friendly neighborhood Oracle salesman
Hybrid operations

- ASO accessed through a BSO lens
  - Blocks still exist, but only to load into a tablesapce
  - BSO outline
  - ASO engine behind the scenes
  - BSO calc script language translated into internal MDX

- Uses two query processors
  - One for Hybrid, the other for BSO

- Hybrid first, failover to BSO

- To effectively use Hybrid, one must understand BSO and ASO
A note about the architecture review

- BSO and ASO are (mostly) well understood
- We can deduce certain things about Hybrid
  - Blocks = elements of BSO
  - Ephemeral tablespaces = elements of ASO
  - Aggregate storage cache = more elements of ASO
- But
  - EAS only returns BSO statistics
  - No published documentation
  - Confusing elements
    - What has been translated internally?
    - Where is the dividing line between BSO and ASO and whatever else is in Hybrid?
    - What is that other element? HyperRoll?
BSO History, Architecture, Storage, and Engine

- Original storage engine
- Data architecture
  - Blocks
    - Organized into PAG files
    - Pointer lists in IND files
- Storage
  - Stored and dynamic level 0
    - Dynamic typically in dense dimensions
  - Stored and dynamic upper level
    - Stored common in sparse dimensions
    - Dynamic common in dense dimensions
    - Poor performance on large dynamic sparse aggregations
    - Attribute calculations are dynamic sparse aggregations
  - Values can be written to upper level stored member combinations
  - Not an in-memory database
- Calculation engine
  - Block Storage Option
ASO History, Architecture, Storage, and Engine

● New storage engine circa Essbase 7.1

● Data architecture
  ● Stores cells in stored tablespaces
    ● Manifested in folder structure
      ● default
      ● metadata
      ● log
      ● temp
    ● Bitmap index drives aggregations
    ● Attribute calculations often have no additional cost

● Storage
  ● Stored at level zero only
  ● Dynamic upper level unless materialized
  ● Data cannot be written to upper level member combinations
  ● In memory database

● Calculation engine
  ● Aggregate Storage Option
Hybrid History, Architecture, Storage, and engine

- Enjoy the speculation reflecting the current released version
  - Subject to change
- Data architecture
  - BSO and ASO and “something else”
  - PAG/IND files and temporary tablespaces
  - Bitmap index drives aggregations if dimension is dynamic
  - Attribute calcs are Classic BSO sparse aggregations
- Storage
  - Blocks at level zero and any stored sparse hierarchy
  - ASO-like bitmap above level zero
  - Data cannot be written to upper level members if dimension is fully dynamic
  - No materialization of aggregated views
  - In-memory database
- Calculation engine
  - They call it Hybrid, but it is ASO-like, whenever possible
    - Hybrid always, even at level zero, even in dense intrablock calculations, if it can
  - But if it can’t, it’s BSO
  - Essbase will always return the right number, but it may be slow
BSO Outline, Scaling, Dimension, and Hierarchy types

- **Outline**
  - Unpaged
  - Dimension types, density/sparsity, and outline order determine calculation order

- **Scaling**
  - Stored upper level data quickly grows with dimensions and hierarchy
  - Calculations can be slow, but the stored retrievals are fast
  - Database is not stored in RAM
  - Theoretical limit is $2^{128}$ potential sparse member combinations; $2^{52}$ cells/block
  - Real world limit is approximately 1,000,000 members; block size variable
  - Interdimensional irrelevancy must be avoided

- **Dimension Types**
  - None, Accounts, Time, Country/Currency (only valid for currency partition)
  - Dense or sparse
  - Attributes

- **Hierarchy types**
  - N/A
ASO Outline, Scaling, Dimension, and Hierarchy types

- **Outline**
  - Paged in memory
  - Outline maps bitmap key to member names on retrieval

- **Scaling**
  - Database size does not grow with hierarchy complexity
  - Leaf data and materialized aggregations grow proportionally to input data
  - Keeping the database completely in RAM is significant for performance
  - Theoretical limit of $2^{52}$ dimension level permutations
  - Real world limit is approximately 10 to 50 million members
    - Time to build and load a limiting factor
    - Interdimensional irrelevancy irrelevant

- **Dimension Type**
  - None, Accounts, Time
  - Compression
  - Attributes

- **Hierarchy types**
  - Dynamic
  - Stored
  - Multiple Hierarchies enabled
Hybrid Outline, Scaling, Dimension, and Hierarchy types

- **Outline**
  - Unpaged
  - If BSO
    - Dimension types, density/sparsity, and outline order determine stored and dynamic calculation order
  - If Hybrid
    - Outline maps bitmap key to member names on retrieval
    - Dimension types, density/sparsity, and outline order determine *dynamic* calculation order

- **Scaling**
  - Database size does not grow with hierarchy complexity
  - Leaf data grows proportionally to input data
  - Keeping the database completely in RAM is vital for performance
  - Same theoretical limits as Classic BSO
    - $2^{^128}$ sparse members; $2^{^52}$ cells/block
  - Real world limits as yet unknown
  - Interdimensional irrelevancy somewhat less relevant

- **Dimension types**
  - Same as BSO

- **Hierarchy types**
  - We believe them to be Multiple Hierarchies Enabled, Stored when possible, non-selectable
BSO Caches, Compression, Languages, and Calculations

- Caches
  - Index
  - Data/Data File
  - Calculator
  - Dynamic Calculator

- Compression
  - Bitmap, RLE, ZLIB

- Languages
  - BSO calc script language formulas and procedural calcs
  - Report writer
  - MDX queries
  - APIs

- Calculations
  - If dynamic, in memory
  - If stored, results in either cells in a block or new blocks
  - Follows dimension type, outline order
  - Different order rules for stored or dynamic calculations
ASO Caches, Compression, Languages, and Calculations

- **Caches**
  - Aggregate storage cache aka Pending cache

- **Compression**
  - Compression dimension

- **Languages and tools**
  - MDX formulas and procedural calcs
  - Report writer
  - MDX queries
  - APIs

- **Calculations**
  - Dynamic on retrieve
  - Stored procedural calculations optional
  - Aggregations only can be materialized
  - Stored hierarchy order irrelevant; solve order for all others
Hybrid Caches, Compression, Languages, and Calculations

- **Caches**
  - Data, index, calculator, dynamic calculator caches if BSO
  - Dynamic calculator cache less important
  - **And** aggregate storage caches
- **Compression**
  - No compression dimension viewable
  - Bitmap, RLE, ZLIB, None
- **Languages and tools**
  - **No** MDX in formulas or calc scripts
  - BSO Calc Script language
  - Report Writer
  - MDX queries
  - APIs
- **Calculations**
  - If Hybrid, **always** dynamic on retrieve
  - If Classic, dynamic or stored calculations
  - All dynamic calcs in memory
  - Hybrid engine follows BSO dynamic calculation order
What gets removed?

- BSO
  - N/A

- ASO
  - Paged outline structure
  - User definable compression dimension
  - Hierarchy types
  - Solve order
  - Query hints
  - Persistent tablespaces
  - Aggregation materialization
  - XOLAP
What doesn’t work in 11.1.2.3.500?

- Time Balance
- Attributes
- Formulas that use cross dims
- Queries that use both TWOPASS and single pass calculation members from the same dimension
- XOLAP
What doesn’t work in 11.1.2.3.500, part II

- Data extraction
  - Essbase report writer
  - DATAEXPORT
- Any member formula that fires in TOPDOWN
- ASODYNAMICAGG PARTIAL
  - All member formulas as documented
- Partitions of source Hybrid to target Classic
What does work?

- All queries work, *always*
  - Engine selection *never* results in incorrect answers
  - But if they revert to Classic, and the sparse dimension is dynamic, they can be S-L-O-W
- Sparse or dense queries that do not violate the Hybrid exceptions
- Source Classic to target Hybrid partitions
What does it look like in EAS?

- Just like a BSO database, but with dynamic upper sparse members
EAS hasn’t caught up to Hybrid just yet

- Dynamic calc warnings
# Member functions

<table>
<thead>
<tr>
<th>Supported</th>
<th>Unsupported</th>
</tr>
</thead>
<tbody>
<tr>
<td>@CHILDREN</td>
<td>Everything else</td>
</tr>
<tr>
<td>@EXP</td>
<td></td>
</tr>
<tr>
<td>@INT</td>
<td></td>
</tr>
<tr>
<td>@ISMBR</td>
<td></td>
</tr>
<tr>
<td>@MIN</td>
<td></td>
</tr>
<tr>
<td>@MINSRANGE</td>
<td></td>
</tr>
<tr>
<td>@MOD</td>
<td></td>
</tr>
<tr>
<td>@MODE</td>
<td></td>
</tr>
<tr>
<td>@NOTEQUAL</td>
<td></td>
</tr>
<tr>
<td>@POWER</td>
<td></td>
</tr>
<tr>
<td>@RANGE</td>
<td></td>
</tr>
<tr>
<td>@REMAINDER</td>
<td></td>
</tr>
<tr>
<td>@ROUND</td>
<td></td>
</tr>
<tr>
<td>@VAR</td>
<td></td>
</tr>
<tr>
<td>@VARIANCEP</td>
<td></td>
</tr>
<tr>
<td>@VARPER</td>
<td></td>
</tr>
</tbody>
</table>
# Valid formulas

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sparse to sparse</td>
<td>Sparse dimension formula that references other sparse members</td>
</tr>
<tr>
<td>Dense to dense</td>
<td>Dense dimension formula that references other dense members</td>
</tr>
<tr>
<td>Sparse to dense/sparse</td>
<td>Sparse dimension formula that references member combinations from dense and sparse members; dense referenced members must be stored</td>
</tr>
</tbody>
</table>
Enabling Hybrid

- Syntax in Essbase.cfg
  ASODYNAMICAGGINBSO [appname [dbname]] NONE | PARTIAL | FULL

- No parameters means all BSO databases are Hybrid
  ASODYNAMICAGGINBSO
  - This is probably not a good idea

- Application specific
  ASODYNAMICAGGINBSO [appname]NONE | PARTIAL | FULL

- Database specific
  ASODYNAMICAGGINBSO [appname] [dbname] NONE | PARTIAL | FULL

- Recommend settings
  ASODYNAMICAGGINBSO NONE
  ASODYNAMICAGGINBSO [appname1] PARTIAL | FULL
  ASODYNAMICAGGINBSO [appname2] PARTIAL | FULL
NONE, PARTIAL, and FULL

- NONE
  - All Hybrid functionality is turned off

- PARTIAL
  - Only +, -, and ~ are supported in Hybrid
  - Member formulas are calculated in block storage

- FULL
  - All operators
  - Hybrid engine for formulas that meet criteria
My Essbase server

- Turn it all off with ASODYNAMICCBSO NONE
- Explicit enabling by application/database
Location of tablespaces

- ASO MaxL statement is highly configurable
  - default, temp, log, and metadata can be unique
  - Separate default and temp for best performance

```
alter tablespace TABLSP-NAME add/alter/delete set
max_file_size/set max_disk_size;
```

- Hybrid Essbase.cfg has one setting
  - All tablespaces can be moved from default Essbase bin drive
  - Only Temp seems to be used
  - Separate from .PAG files for best performance

ASODYNAMICAGGINB SoFOLDERPATH [appname] path-to-directory
Ephemeral tablespaces in directories

- Started

- Stopped
Aggregate storage cache and its role in Hybrid

- Same MaxL for ASO and Hybrid

```sql
alter application APP-NAME set cache_size ;
alter application APP-NAME get cache_size ;
```

- 32 megabytes is the default for both engines

- Less important than in ASO
  - In some cases Hybrid beats ASO for performance
    - Less is more
    - Hybrid only reads required blocks on retrieval
    - Level 0 block count relatively not that big
  - Could be impacted by high number of simultaneous queries; more testing required
How do we know that Hybrid worked, or didn’t?

- Query times
  - If it did fire, it’ll be fast
  - If it didn’t, potentially quite slow
- Essbase application log file review
These are the magic words

[Sun Apr 13 13:36:14 2014]Local/SampleH/Basic/hypadmin@Native Directory/12752/Info(1204002)

Hybrid Aggregation Mode enabled.

- There is no other log indication that a query used the Hybrid engine
- You will of course notice it on retrieve if it takes 20 minutes
- Remember that Essbase will always return the correct data value
You will get used to this

- **XREF**
  
  Hybrid Aggregation Mode disabled for [XREF test] due to [xref is not supported yet].

- **Time Balance**
  
  Hybrid Aggregation Mode disabled for [Opening Inventory] due to [time balance is not supported yet].

- **Top-down**
  
  Hybrid Aggregation Mode disabled for [YTD] due to [top-down formula is not supported yet].

- **Non-supported**
  
  Hybrid Aggregation Mode disabled for [YTD] due to [expression in formula contains non-supported function].

- **Partition mappings**
  
  Hybrid Aggregation Mode disabled for [Loaded_Net Revenues_ALL_Prod] due to [a formula that appears in partition is not supported yet].
Design considerations

● Potentially no more AGGs/CALC DIMs
● Database size can be vastly smaller
● Valid sparse dynamic calc member formulas are now cost free

● Caches
  ● Data cache need only contain level zero blocks
  ● Index cache can be much smaller because the block count is so much smaller
  ● Only exist to get blocks into tablescape
  ● Aggregate storage cache sizing now possibly relevant
Design considerations, part II

- **Blocksize**
  - Required blocks are pulled into the tablespace on retrieve
  - Must balance size versus count as always
  - Question now is: How fast does Hybrid read the blocks into the tablespace?

- **Dimension count**
  - Dimensional irrelevancy a little less important
  - Attributes can become stored sparse dimensions
    - Subject to overall block count limit
Mix and match as required

- Dynamically sparse dimensions are an option
- Use stored and dynamic dimensions as required
  - Use BSO to support advanced calculations, Hybrid for simple rollups
  - Akin to dropping a sparse dimension
- Mix stored and dynamic subhierarchies
  - Aggregate primary hierarchies in BSO, alternates in Hybrid
  - Aggregate only some subtotals, Hybrid the rest
- BSO goal is to reduce scope
  - Creativity (and testing) is your friend
How can BSO databases take advantage of Hybrid beyond hierarchies?

- Coloring outside the lines
- BSO limits still apply
  - $2^{128}$ sparse member combinations
  - Theoretical limit of $2^{52}$ cells per block
- Example: Can we get around slow attribute dimensions by making them stored?
Converting attributes to base

- BSO attributes can be painful
  - Sparse dynamic calculations are often bad news
- Convert three attribute dimensions to stored
- A bad idea in BSO
  - > 60,000 second calculation times
  - > 1 TB storage
- But what about Hybrid?
  - No increase in level zero blocks or size
  - Small increase in load time and retrieve
- “Attributes” are now useful for reporting
Test databases

- Derived from ASO Planning presentation
- BSO, ASO, and Hybrid (attribute and stored attribute) versions of the same database
- With attributes and attributes as stored members
- Goals
  - Disk requirements
  - Memory requirements
  - Batch load, aggregate (BSO only), retrieve top levels
## Dimension counts

<table>
<thead>
<tr>
<th>Dimension</th>
<th>BSO</th>
<th>ASO</th>
<th>Hybrid</th>
</tr>
</thead>
<tbody>
<tr>
<td>HSP_Rates</td>
<td>15</td>
<td>N/A</td>
<td>15/15</td>
</tr>
<tr>
<td>Account</td>
<td>3,225/2,800</td>
<td>3,224/3,192</td>
<td>3,228/2,800</td>
</tr>
<tr>
<td>Period</td>
<td>19/13</td>
<td>19/18</td>
<td>19/13</td>
</tr>
<tr>
<td>Year</td>
<td>7/7</td>
<td>7/6</td>
<td>7/7</td>
</tr>
<tr>
<td>Scenario</td>
<td>4/4</td>
<td>4/3</td>
<td>4/4</td>
</tr>
<tr>
<td>Version</td>
<td>3/3</td>
<td>3/2</td>
<td>3/3</td>
</tr>
<tr>
<td>Currency</td>
<td>3/3</td>
<td>3/2</td>
<td>3/3</td>
</tr>
<tr>
<td>Product</td>
<td><strong>72,177/72,177</strong></td>
<td>72,177/72,176</td>
<td><strong>72,177/63,557</strong></td>
</tr>
<tr>
<td>PostCode</td>
<td><strong>45,468/45,468</strong></td>
<td>45,468/45,467</td>
<td><strong>45,468/43,589</strong></td>
</tr>
<tr>
<td>Analytic</td>
<td>N/A</td>
<td>6/4</td>
<td>N/A</td>
</tr>
<tr>
<td>Fx Rates</td>
<td>N/A</td>
<td>14/13</td>
<td>N/A</td>
</tr>
<tr>
<td>Product attribute #1</td>
<td>28</td>
<td>28</td>
<td>28</td>
</tr>
<tr>
<td>PostCode attribute #1</td>
<td>1,095</td>
<td>1,095</td>
<td>1,095</td>
</tr>
<tr>
<td>PostCode attribute #2</td>
<td>31</td>
<td>31</td>
<td>31</td>
</tr>
</tbody>
</table>
Data description

● Data set
  ● Dan’s 5% solution
    ● Period and Account are 5% dense
  ● 76 million rows
    ● 907 million non-empty cells
  ● 16 gigabytes in size

● Is this big?
  ● BSO – yes, huge
  ● ASO – no, small
  ● Hybrid – no, moderate
The essence of BSO data explosion

- Aggregation causes upper level members combinations (blocks) to be evaluated and created
  - This is why BSO aggregations get slower as dimensions are processed in an AGG
- Lower level member combinations (blocks) remain the same on aggregation
- Remove the likely upper level combinations and database size drops dramatically
How can so little mean so much?

<table>
<thead>
<tr>
<th>Dimension</th>
<th>Type</th>
<th>Declared</th>
<th>Stored</th>
<th>Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td>HSP_Rates</td>
<td>Sparse</td>
<td>15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Account</td>
<td>Dense</td>
<td>3,225</td>
<td>2,800</td>
<td>425</td>
</tr>
<tr>
<td>Period</td>
<td>Dense</td>
<td>19</td>
<td>13</td>
<td>6</td>
</tr>
<tr>
<td>Year</td>
<td>Sparse</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>Scenario</td>
<td>Sparse</td>
<td>4</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>Version</td>
<td>Sparse</td>
<td>3</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>Currency</td>
<td>Sparse</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Product</td>
<td>Sparse</td>
<td>72,177</td>
<td>63,557</td>
<td>8,620</td>
</tr>
<tr>
<td>Postcode</td>
<td>Sparse</td>
<td>45,468</td>
<td>43,589</td>
<td>1,879</td>
</tr>
<tr>
<td>Potential</td>
<td></td>
<td>12,404,991,700,080</td>
<td>10,472,059,355,940</td>
<td>1,932,932,344,140</td>
</tr>
</tbody>
</table>

Potential but *unlikely* level zero blocks

Potential but *likely* upper level blocks

Dense, irrelevant
How big is it in Essbase?

**Load time (secs)**

- BSO: 1,222
- BSO with Stored Att.: 1,201
- BSO Hybrid: 1,414
- BSO Hybrid with Stored Att.: 1,598
- ASO: 8,061
- ASO with Stored Att.: 5,817

**Agg time (secs)**

- BSO: 17,962
- BSO with Stored Att.: >60,000

**Level 0 storage (GB)**

- BSO: 8.5
- BSO with Stored Att.: 8.5
- BSO Hybrid: 8.5
- BSO Hybrid with Stored Att.: 8.5
- ASO: 14
- ASO with Stored Att.: 14

**Upper level data size (GB)**

- BSO: 200
- BSO with Stored Att.: >1,000
Query performance, or why those BSO aggs might be worthwhile

**Top of house, no attributes**

- ASO: 12.683, 2nd Pass: 14.992
- BSO: 0, 1st Pass: 0.187
- BSO Hybrid: 2.262, 1st Pass: 18.002
- BSO Hybrid with Stored Att.: 2.309, 1st Pass: 43.789

**Both sparse in POV**

- ASO: 33.025, 2nd Pass: 35.274
- BSO: 0.016, 1st Pass: 0.281
- BSO Hybrid: 5.867, 1st Pass: 21.607
- BSO Hybrid with Stored Att.: 2.309, 1st Pass: 43.789

**Product in grid**

- ASO: 0.421, 2nd Pass: 1.327
- BSO: 0.031, 1st Pass: 0.032
- BSO Hybrid: 1.855, 1st Pass: 1.888
- BSO Hybrid with Stored Att.: 2.887, 1st Pass: 2.947

**Postcode in grid**

- ASO: 0.874, 2nd Pass: 0.889
- BSO: 0.021, 1st Pass: 0.048
- BSO Hybrid: 3.495, 1st Pass: 3.399
- BSO Hybrid with Stored Att.: 4.385, 1st Pass: 4.444
The unexpected impact of attributes

- Hybrid fails on attributes, so all calcs are then dynamic BSO
- How can **two** BSO attributes (plus BSO dynamic base) be faster than one?
  - The numbers don’t lie, although are somewhat fantastic
  - This is BSO behavior, but manifests in Hybrid because it forces BSO
  - One attribute dimension is slow, two attribute dimensions from different base dimensions are fast
  - Two attribute dimensions from the same base dimension are slow
Coming soon to an Essbase server near you

- We were privileged to test an alpha version of Hybrid on Oracle’s servers in a supervised environment
- Oracle are adding
  - Dynamic Time Series
  - Time Balance
  - Hybrid can now evaluate valid Hybrid calcs vs. BSO-only calcs simultaneously
- Still not there
  - Attributes
  - Cross dims
  - Top-down
  - Report scripts
  - DATAEXPORT
When does Hybrid make sense?

- **Use cases**
  - Simple BSO rack and stack databases
  - Dynamic calcs of more than 100 children
- **Partitions**
  - Transparent partitions between two BSO databases
  - Hybrid is a good candidate for replacing BSO to ASO partitions if Hybrid rules can be followed
- **Databases where mixed stored and dynamic sparse dimensions mitigate Hybrid’s limitations**
- **Technology exploration**
Where does Hybrid not make sense

- Violation of any of the unsupported functions
- Conversion of ASO to BSO
- Hyperion Planning
  - Cross dims, top-down, powerful functions are all typical Planning use cases
    - If upper level allocations are not used, Hybrid can eliminate top of the house blocks
    - This is a semi-dangerous design decision
  - Expect this to change in future
Q&A

Cameron Lackpour
cameron@clsolve.com
http://camerons-blog-for-essbase-hackers.blogspot.com/
@CameronLackpour

Tim German
tim.german@qubix.com
http://www.cubecoder.com/