Essbase Optimization

Flash Back Friday!!

Kevin Cox

Who Am I?

- Kevin Cox
- Essbase (Arbor/Hyperion/Oracle) since 1996 (4.1.2)
- Work
  - BDalton Software Etc 1986-1993 ("Gamestop 1.0")
  - Target Corp 1993-2000
  - EPM consultant 2000-2011
  - Lennox Intl 2011-curr
- NTxHUG

Kscope Plug

- If you are employed doing Hyperion/Essbase/Oracle, you NEED to go
- Presentations about solutions, by the solver
- Real industry experts (consulting and client)
- I "sell" it to management as my CPE need
- Casual conversations priceless
- Never had issue with getting near instant payback on Kscope investment

ORACLE

ENTERPRISE PERFORMANCE MANAGEMENT

ARBOR SOFTWARE

What is Essbase

Hyperion

- **E**xtended **S**pread**S**heet data**BASE**
- Developed in early 1990s
- "Dimensions"
- Following new database concept of OLAP
- Data arranged in an array of "Blocks"
- Later "short-hand": Block Storage Option- BSO
- For this presentation: "Essbase" = BSO

Essbase: Truths

- Unique database storage
- Works in specific order
- Excels at math, “can” do logic
- Needs data in memory to work on it
- Highest processing “cost” is I/O time
  - *Finding*
  - Loading
  - Writing

An Extended Spread Sheet

- Think of an Excel workbook
- You can't just pull cell A1 into memory
- When opened, it requires computer memory
- Calculations referencing other cells in same workbook are fast
- References to external books take more time
- You have many similar, but slightly different workbooks (months, products, years, etc)

Storage: Dense

- "high probability that one or more cells is occupied"
- Dense considerations
  - highly populated
  - calculations (Ratios, variances, etc)
  - Heavily relevant (one value implies another)
  - Populated at once (if possible)
  - Relatively static (not constantly adding new)
- Once Dense dims defined, ALL blocks are defined by that exact definition (none bigger nor smaller)
- Guideline: 8k-64k block size, 2-3 dims (based on 1990s computers)

Storage: Dense

- Blocks are RARELY (never) 100% populated
  - >50% good, ~20% common
- Essbase uses "#Missing" (relational "NULL")
- Essbase compresses empty space out of block before writing to disk
  - Helpful, but troublesome (a trademark)

Storage: Dense

- In memory, blocks are fully expanded ("block size")
- Blocks need to be loaded to memory ANY time referenced
  - Retrieved
  - Calculated (also referenced in calculation)
  - Updated
- Block size impacts remote access

Storage: Sparse

- All potential blocks are determined by members in sparse dimensions (including ancestors) and all are given a unique serial number ("shared" members don't count)
- Essbase does all processing in block number order

Storage: Sparse

- Sparse dimensions define individual blocks
- Potential blocks frequently in the 10^{10}+ range
- "Existing blocks" usually in single digit %
- Index files (*.ind) are a guide to where block are located

An Extended Spread Sheet

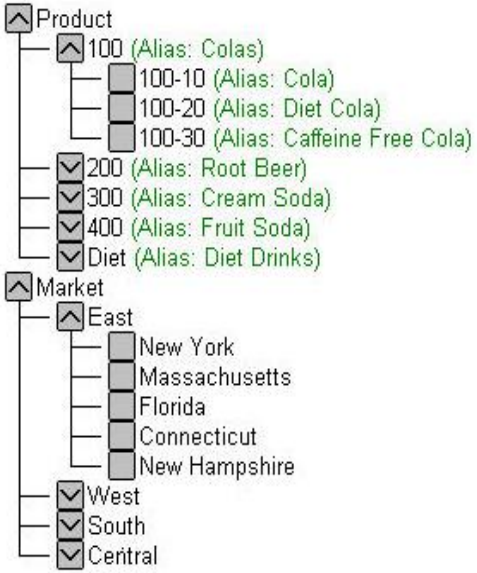
- Dense dimensions define size of the "workbook" definition (rows/cells/sheets)
- Sparse dimensions ultimately create different, multiple workbooks
  - Sparse combinations are "file names"

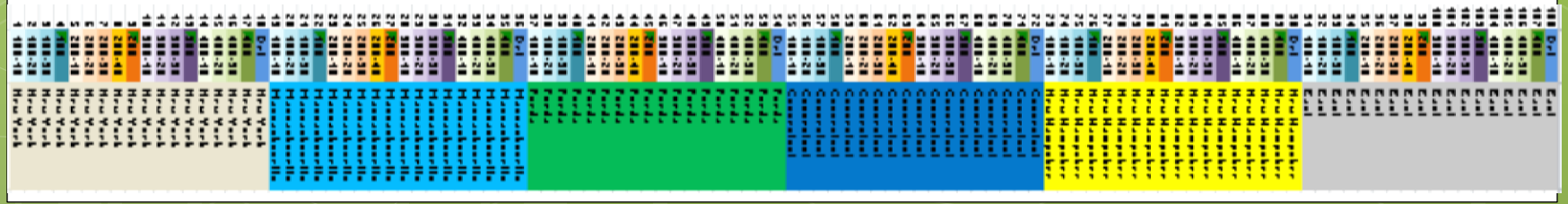
Storage: Block Numbering

- Outline order
- First block is lev0 of each sparse
- 2nd block is second member of first listed dimension
- Reason for “hourglass”

Storage: Visual

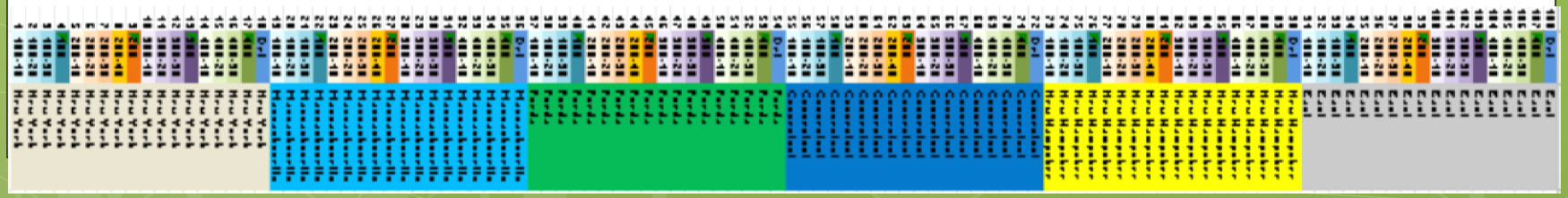
Sample: Basic





Fragmentation

- "Fragmentation" is any blocks out of sequence on disk
- Inevitable Causes
  - Aggregation
  - Loading from multiple sources
  - Periodic loads*
- Preventable Causes
  - Bad load sequencing
  - Derived values stored in dense
  - Periodic loads*



Storage Compression

Calculation

- Without other instructions, calculation runs on block calculations in block order for all existing “dirty” blocks
- Top Down / Bottom Up
- Housekeeping
- “Smart Calc”/”Update Calc”/”UPDATESTATUS”
- Dense Order
- FIX (emptyset)
- Member calc
- Limit limit limit
- Block creation issue
- Datacopy

Calculation

- Assuming "Net Sales" has a member formula in outline
  - "Gross Sales" – "Discounts"
- Calc script could be as simple as

  "Net Sales;"
- Alternately, formula in calc script
  - "Net Sales" = "Gross Sales" – "Discounts"
- Best practice is to get logic in outline (not "buried" in a calc script)
- …and dynamic

Calc Script member calculation

- any logic not simply “a = b + c” has to be in a “member calculation block”

```
A ( If B > C
      A = B – C;
    Else
      A = C – B;
    EndIf)
```

- All functions, logic, etc, USUALLY need to be in a logic block

Calc Script calculation

- Almost anything possible in block
  - Functions
  - “Cross dims” on both sides
- Not possible…
  - Calc Script Procedural functions (FIX, Datacopy)
- Make sure Block member exists

CELL/BLOCK (d) TOPDOWN/BOTTOMUP (s)

- BLOCK: Faster, default order, at once, but requires no dependence
- CELL: Slower, cell-by-cell

- BOTTOMUP: Faster, follows outline order
- TOPDOWN: Slower, more deliberate

PARALLELCALC/TASKDIMS

- Default mode is single lane, one processor
- Splits work into multiple lanes, for multiple processors
- Uses bottom dimension to split
- TASKDIMS allows more dimensions in split calculation

Calc All

- Simplest calculation is using Essbase's "Calc All" function
- Calculates all dimensions
  - Member formulas
  - Aggregation properties
- Frequently more than needed
- Could be not enough if "Intelligent calc"
  - "Should" get everything

Calc vs Agg

- "Calc All" / "Calc Dim" runs calculations on dimensions listed, including aggregation
- "Agg" only looks at Agg properties
  - MUCH faster
  - does NOT do member calcs

AGG("Market", "Product");

Intelligent Calculation

- “Dirty” blocks
  - Blocks that have been updated
- Intentions are pure
- Difficult to manage

Housekeeping

- SET AGGMISSG
- SET UPDATECALC
  - CLEARUDPATESTATUS
- SET MSG (ONLY)
- SET EMPTYMEMBERSETS

Housekeeping: AGGMISSG

- SET AGGMISSG ON
- Assumes all data is loaded at lowest level
- Benefit: Database doesn't have to "look" before aggregating
- Optimization: HIGH

Housekeeping: UPDATECALC

- Dirty Blocks
- Great concept, hard to control
- Tends to result in not enough calculation
- Therefore, usually “OFF”
- Also turn off “CLEARUPDATESTATUS”

Housekeeping: MSG

- During development, use “SET MSG ONLY” to get loose guideline for speed
- After implemented, SET MSG SUMMARY is common

Control Flow: FIX/EXCLUDE

- Calculation runs through all blocks, unless limited
- FIX/EXCLUDE limits dimensions to fewer members
- If member not found, ALL members calculated
  - Use EMPTYMEMBERSETS
- Encouraged to have a separate FIX for each dimension

FIX (cont)

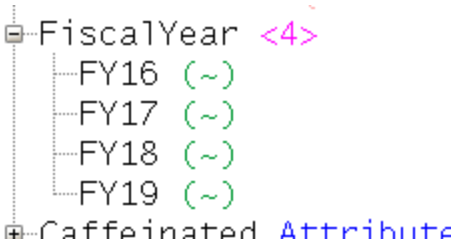
- Will limit dimensions where member present, otherwise “ALL”
- Can be Nested, with caution
  - Further Fix on same dimension sometimes causes issues

FIX(@IDESCENDANTS(“Period”))

FIX(“Jan”)

FIX (cont)

```
FIX(    @CHILDREN("EAST"),
        @RSIBLINGS("FY19"))
    (calculations)
ENDFIX
```




```
FIX(@CHILDREN("EAST"))
 FIX(@RSIBLINGS("FY19"))
 (calculations)
 ENDFIX
ENDFIX
```

Math vs Logic

Essbase CAN do logic, but it is MUCH faster to do math

1. If(Act==0) Act=#Missing; Endif
2. Act = Act * Act/Act;

   (credit: Roske--leverages 0/#Missing math)

- Ex 2 is multiples faster than Ex 1 (10x?)

- Boolean values…True = 1, False = 0

Clean up Fragmentation

- “Block adjacency ratio”
- Dense Restructure
- Remove zeros in database
- Export all/clear/reimport

CLEARDATA/CLEARBLOCK

- CLEARDATA *mbrname* sets values to #mi
  - Can't be used in IF statement
  - Mbrname = #Missing
  - Block not deleted
- CLEARBLOCK *type* sets values to #mi AND evaluates full block for deletion

FIX SPARSE/CALC DENSE

- Old Essbase Saying
- FIX primarily benefits limiting Sparse dimensions
  - However, shows SOME improvement on FIX on Dense
- CALC can be used on Sparse, but logically fits better with Dense/Accounts/Period

Block Creation Issue

- Calc only runs on existing blocks, therefore 99x faster than running on ALL potential (thanks Essbase!!)
- …but calc, by definition, creates new data
- If calc needed to create a new block, this in NOT default functionality
- Example: “Currency” is sparse
  - “USD” = “EUR” * ExcRate;

CREATE

- CREATEBLOCKONEQ
  - Can create blocks if needed
  - Slow, because it isn't just calculating
  - Procedural
- CREATENONMISSINGBLK
  - Slower functionality due to logic required
  - Be careful
- @ALLOCATE

DATACOPY

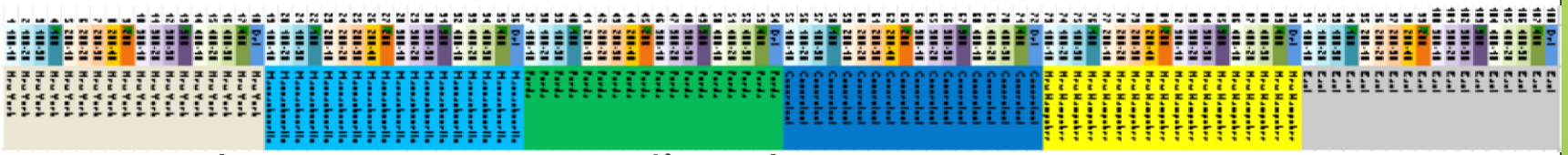
- My preference: Datacopy
- Copies all data from source to target
  - Including #Mi values (and #Mi blocks)
- Helps with Block Creation issue
- Assuming properly limited via FIX, provides better control
- Example: Datacopy EUR to USD, and then fix on USD and calculate

Dense Design

- Best Practice: ONLY store loaded data
- All derived data should be dynamic
- Ratios/Variance/Two Pass dynamic
- Test block sizes/cache size
  - As low as 2kb, as high as 800+kb
- Test calculations with SET MSG ONLY;

Sparse Design

- Think about what dimensions you may limit via FIX, place FIXed dims low



- Place non-agg dims low
- 80/20 on rollups
  - Make the most used rollup the primary rollup
  - Alternate rollups…consider dynamic

General Outline

- Reduce Interdimensional Irrelevance
  - Make sure most of your member from each dimension relate to most in other dimensions
  - Example: Entity Income Statement app, lots of entities, lots of accounts
    - Add a “truck” dimension with 1000+ trucks that keeps track of mileage and gas

Essbase Optimization

- Questions / Comments

Implied Share

- Created when parent has one child
- Results in ONE “bucket” being created (instead of two)
- “NEVER SHARE” member setting
- “IMPLIED_SHARE” config setting