Create Optimized Hybrid Mode Cubes

The Baseline, Calc Cache, Solve Order, and Data Distribution Optimize Cube utilities help you fine tune your cubes for better performance.

Create optimized hybrid mode cubes using these four Optimize Cube utilities:

Utility	Data Returned
Baseline	Cube performance metrics
Solve Order	Solve order of the members in the cube
Calc Cache	Data to help you choose the best calculator cache value for the cube
Data Distribution	Data to help you choose which dimensions to make sparse and which to make dense

- Optimize Baseline Metrics on a Hybrid Mode Cube
- Optimize the Solve Order on a Hybrid Mode Cube
- Optimize the Calculator Cache on a Hybrid Mode Cube
- Optimize Data Distribution on a Hybrid Mode Cube

Optimize Baseline Metrics on a Hybrid Mode Cube

The metrics tracked by the Baseline utility show how the system is performing. Use these metrics to determine the baseline performance, and then to measure the benefits of the subsequent optimizations that you make.

Before using this utility, you first create an application workbook, including the outline, configuration settings, calculation scripts and queries you want to include in the cube.

When you run the utility, it builds the cube, loads the selected data files, executes the selected calculation scripts, and runs the queries contained in the application workbook. It is important to have a representative sample of queries from your users.

The baseline utility creates a dashboard of the application and operational processes, which can help you to design and optimize the cube. As you implement changes and

rebuild the cube, the baseline helps you compare iterations of cube modifications. On the **Essbase.Stats.Baseline** tab of the application workbook, the baseline utility appends new tables with the latest data for each iteration.

Prepare to run the Optimize Cube Baseline Utility on a Hybrid Mode Cube

Complete these tasks before running the baseline utility:

1. Design and create your application workbook.

To create an application workbook, you can download a sample application workbook and then modify it to suit your needs. See Explore the Gallery Templates.

2. Clear the query sheets in the application workbook of Smart View metadata:
    a. Go to the Smart View ribbon.
    b. Choose **Sheet Info** and click **Delete**.

If the query sheets have metadata from a different server, cube designer displays a warning and pauses processing until you respond.

3. Modify the **Cube.Settings** worksheet with the following **Application Configuration** settings:

Setting	Value
ASODYNAMICAGGINBSO	FULL
HYBRIDBSOINCALCSCRIPT	NONE
INDEXCACHESIZE	100M
DATACACHESIZE	100M
ASODEFAULTCACHESIZE	100
MAXFORMULACACHESIZE	102400

Setting	Value
INPLACEDATAWRITEMARGINPERCENT	20
CALCCACHEDEFAULT	200000
LONGQUERYTIMETHRESHOLD	-1

Run the Optimize Cube Baseline Utility on a Hybrid Mode Cube

The Baseline utility identifies dense and sparse dimensions, data size (PAG and IND file sizes), block size, and the data, index, and calculator cache sizes. Additionally, it provides metrics for the data load, calculation, and query.

To run the Baseline utility:

1. From the Cube Designer ribbon, select **Admin Tasks** > **Optimize Cube**.
2. (Optional) Click **Customize** to choose which baseline operations to run.
    - **Build cube** - Build the cube defined in the application workbook and load the data in the data sheets.
    - **Run calc scripts** - Run the calculation scripts defined in each of the calculation sheets in the application workbook.

Calculation worksheets run in the order they appear in the application workbook. Optimize Cube ignores the **Execute Calc** property on the calculation sheets.

Only calculation scripts that can be run from Jobs are supported with Optimize Cube. You cannot run calculation scripts that depend on the current Smart View grid context (for example, calculations defined using the @GRIDTUPLES function, or those that use runtime substitution variables defined with <svLaunch> tags).

    - **Run queries** - Run the queries on the Query sheets.
    - **Export all** - Export all the data in the cube to the cube directory. After the export time and file size are recorded, the export file is deleted automatically.
3. Click **Create Baseline**.

If you don’t have a data sheet in the application workbook, you’ll be prompted to select data and rule files from the catalog. It is a good practice to store the data and rule files

in a shared directory in the catalog so that the files won't be lost when you rebuild the cube.

It will take some time to build the cube.

Essbase generates the **Essbase.Stats.Baseline** sheet and adds it to the workbook.

4. View the **Essbase.Stats.Baseline** sheet in the application workbook.
    - The first table on the sheet displays the size of the data load files, the number of data load cells, block size, and cache sizes.

Dataload File/s(GB)	140.5 MB
Dataload Cells	15,678,463
Block Size(Bytes)	157,920
Data Cache(MB)	100
Index Cache(MB)	100
Calc Cache(Bytes)	2,500

    - The colors in the baseline table identify the storage type for each dimension:
        - Green - dense dimension
        - Red - sparse dimension with at least one dynamic formula
        - Blue - sparse dimension with aggregations and without all dynamic parents and formulas
        - Gold - other sparse dimension

Baseline			
Dimension	Type	Stored Members	Total Members
Account	DENSE	987	1,515
Period	DENSE	20	142
Entity	SPARSE	12,791	16,133
Currency	SPARSE	2	3
Version	SPARSE	9	9
Initiatives	SPARSE	1	2
Year	SPARSE	13	13
Scenario	SPARSE	11	12
Function	SPARSE	0	35
PG_ATTR	SPARSE	0	163
PL_ATTR	SPARSE	0	134
MG_ATTR	SPARSE	0	10

- Under **Load and Calc**, the individual “Script:” rows identify which calculation script takes the longest to complete and thus might need optimizing.

Load and Calc				
Operation	Time (sec)	Blocks	Data (PAG)	Index (IND)
Initial Data Load	87.00	125,063	234,799,155	8,216,576
Script: All	29.00	199,749	641,187,891	16,408,576

- Under **Query**, **Blocks Read**, it shows the amount of data requested by the query.

Changing a dynamic dimension to stored reduces that amount.

- Under **Query**, **Formulas**, it shows the number of formulas executed in the query.

Review the solve orders of calculated members and make changes to reduce the number of formula executions and improve performance, or consider storing a calculated member containing formulas to reduce the number of formula executions and improve performance.

Query			
Operation	Time (sec)	Blocks Read	Formulas
Query: Test	0.33	275	84

- The last table on the sheet displays the export time and file size.

Export All	
Time (sec)	File Size(MB)
43.00	393.02

Optimize the Solve Order on a Hybrid Mode Cube

The Solve Order utility gives you a visual representation of the solve order flow used in the application. This can help diagnose query performance problems relating to formulas.

To run the Optimize Cube Solve Order utility:

1. From the Cube Designer ribbon, select **Admin Tasks** > **Optimize Cube**.
2. Click **Solve Order**.
3. View the **Essbase.Stats.SolveOrder** sheet of the application workbook.

Use the information in the **Essbase.Stats.SolveOrder** sheet to adjust the solve order to optimize query performance. See Optimize the Cube for Hybrid Mode and Solve Order in Hybrid Mode.

Optimize the Calculator Cache on a Hybrid Mode Cube

The Calc Cache utility recommends the optimal calculator cache setting for the cube.

Using the correct calculator cache setting can be an important performance enhancement when calculating entire sparse dimensions in a calculation script. Calculating an entire sparse dimension is a technique for reducing the number of blocks required by a query.

The default value of the calculator cache is 200,000 bytes. The maximum value is 20,000,000 bytes.

The calculator cache should be set to just large enough to contain the sparse dimensions that are calculated in the calculation script. Setting the calculator cache to larger than it needs to be has a negative impact on performance.

To optimize the calculator cache using the Calc Cache utility:

1. In order to reduce the amount of data requested by the query, calculate and store one or more dimensions using a calculation script.

The best choice is usually the largest dimension.

2. Move that dimension to be the first sparse dimension in the outline.

The calculator cache algorithm selects the sparse dimensions to place in the cache, beginning with the first sparse dimension.

3. Build the cube without loading data.

The cube must be built for the Calc Cache utility to work.

4. Run the Calc Cache utility.

The utility displays the correct cache setting next to each dimension up to 20 MB. Beyond 20 MB, it shows N/A. Generally, settings above a couple of MB are not needed.

   a. From the Cube Designer ribbon, select **Admin Tasks** > **Optimize Cube**.

   b. Click **Calc Cache**.

   c. View the **Essbase.Stats.CalcCache** sheet of the application workbook. You can view the recommended calculator cache settings in the **Essbase.Stats.CalcCache** worksheet, in the **Calc Cache** column.

Dimension	Storage	Total Members	Dependent Parents	Calc Cache (Bytes)
Account	DENSE	1,515		
Period	DENSE	142		
Entity	SPARSE	16,133		2,017
Currency	SPARSE	3		6,050
Version	SPARSE	9		54,449
Initiatives	SPARSE	2		108,898
Year	SPARSE	13		1,415,671
Scenario	SPARSE	12		2,831,342

5. Find the **Calc Cache** setting in the **Essbase.Stats.CalcCache** sheet, next to the sparse dimension(s) you calculated and stored in step 1.

6. If you calculated one dimension in step 1, set the calculator cache default to that value. If you calculated more than one dimension in step 1, choose the highest **Calc Cache** value from among the values you calculated.

Add this value to the Application Settings section of the **Cube.Settings** worksheet. Alternatively, you can set the value in the application configuration settings in the Essbase web interface. It is a good practice to round up, in order to allow a little more room.

Optimize Data Distribution on a Hybrid Mode Cube

The data distribution utility helps you better understand the data in an application, enabling you to make important decisions about how to optimize your cube.

Understanding the data helps you determine the following:

- Which dimensions to make dense and which to make sparse.

Dense dimensions define the blocks in a block storage application. Ideally, a block should contain dimensions with the most data and represent the predominant query layout for that application. For financial reporting applications, this usually means the Time and Account dimensions should be dense.

- Which dimensions to calculate and store using a calculation script.

One of the factors that affects query performance is the number of blocks requested by the query. If the number of blocks requested is too high, the query performance suffers. To reduce the number of blocks requested, pre-calculate upper level members of one or more sparse dimensions. First, set the dimension storage attribute of the upper members to a stored attribute (Store or Never Share), and then run a calculation script that aggregates that dimension using either CALC DIM or AGG.

- Which dimensions to use as the task dimension in the FIXPARALLEL command.

To optimize the calculation script used to aggregate the stored sparse dimensions, use the FIXPARALLEL command. It is important to select the correct task dimensions. A task dimension is the one that determines how the calculation is split into threads and executed in parallel. One or more sparse dimensions should contain the most data to reduce empty tasks, and ideally, that data should be evenly distributed.

To run the Data Distribution utility:

1. From the Cube Designer ribbon, select **Admin Tasks** > **Optimize Cube**.
2. Select **Data Distribution**.

This process can take a long time to run, especially on larger models.

3. View the **Essbase.Stats.DataDist** worksheet.

Dimension	Non-Aggregating	Contains Formulas	Base for attribute	Stored Members	Total Members
Account		X		987	1,515
Period				20	142
Entity			X	12,791	16,133
Currency	X			2	3
Version	X			9	9
Initiatives				1	2
Year	X			13	13
Scenario	X	X		11	12

DataFile	anondata.txt
Dataload Files Size	140.5 MB
Dataload Cells	15,678,463

Blocks	Cells per block
1,103,501	14.21
2,309,337	6.79
265,026	59.16
8,671,759	1.81
10,380,425	1.51
15,678,463	1.00
9,310,087	1.68
13,346,605	1.17