Optimizing performance for SAP BusinessObjects Profitability and Cost Management

John Carter
SAP Solution Management
Session 1906: Optimizing performance for SAP BusinessObjects Profitability and Cost Management

In this session we review the techniques for achieving the optimum performance for PCM, covering design options, data loading, assignments, rules and end user interaction.
Learning Points

- Understand the factors that influence the performance of SAP BusinessObjects Profitability and Cost Management
- Appreciate how design decisions impact performance, and how to make optimum design choices
- Review a standard model optimization checklist
VIEWS OF OPTIMISATION
Views of Optimisation

Not everybody has the same optimization priorities

- End to end processing time
- Time to access reports
- Rapid data entry and interactive calculations
- Minimum time and effort to make model changes
- Minimum TCO and license costs

Ensure you have a clear understanding of what performance criteria are important. Design your solution around those criteria. Optimising performance for one goal can have detrimental impacts on others.
We can trade end to end calculation time with reporting time by using results published to the database rather than pulling results from the application tier in real time.
FACTORS THAT INFLUENCE PERFORMANCE
Factors than influence performance

- Application Design
- Model Size and Data Density
- Model Complexity and Rules
- Hardware
- Concurrency & Usage profile

There are many different items that influence performance – these are some of the most important.
Approaches to Optimization

- Business Design
- System Design
- Model Design
- Technical
- Hardware

Application performance optimized may be approached by considering different layers of the implementation.
DESIGN DECISIONS
Design Decisions

For example

- Consider PCM within the broader landscape
- Question the level of detail
- Use multiple models
- Periodic model with published results
- Split Data entry from calculation
- ...

For large applications where performance is critical, verify the design with experienced experts.
Design Decisions – consider the broader landscape

Transactional sources: SAP & non SAP

This design moves over 2 billion records between environments
Design Decisions – consider the broader landscape

Consider shifting some of the workload into the reporting tier when PCM is not the primary reporting mechanism.

This design moves about 100th times less data between environments.

Transaction sources: SAP & non SAP
Design Decisions – level of detail

Quantity of values in each range

Values

Real Experience. Real Advantage.
Design Decisions - level of detail

Sum of values in each range

Values

-100,000,000 -1,000,000 -100,000 -10,000 -1,000 -10 0 1 10 100 1,000 10,000 100,000 1,000,000

Sum

-40,000,000 -20,000,000 0 20,000,000 40,000,000 60,000,000 80,000,000 100,000,000 120,000,000

Quantity of values in each range

Values

-100,000,000 -1,000,000 -100,000 -10,000 -1,000 -10 0 1 10 100 1,000 10,000 100,000 1,000,000

Quantity

0 2,000,000 4,000,000 6,000,000 8,000,000 10,000,000 12,000,000 14,000,000 16,000,000
### Design Decisions - Level of Detail

#### Efficiency Ranking for Each Value Range

<table>
<thead>
<tr>
<th>Value Range From</th>
<th>Value Range To</th>
<th>Quantity of Values in Range</th>
<th>Quantity of Values in Range %</th>
<th>Values in Range Summed</th>
<th>+/- Values Sub-Totled</th>
<th>Significance of Values in Range</th>
<th>Efficiency Ranking</th>
</tr>
</thead>
<tbody>
<tr>
<td>-$1</td>
<td>$0</td>
<td>1,530,040</td>
<td>8.744321%</td>
<td>-$111,810</td>
<td>-$89,309,806</td>
<td>0.02%</td>
<td>-8.72%</td>
</tr>
<tr>
<td>$0</td>
<td>$1</td>
<td>13,239,289</td>
<td>75.66378%</td>
<td>$1,002,541</td>
<td></td>
<td>0.20%</td>
<td>-75.47%</td>
</tr>
<tr>
<td>$1</td>
<td>$10</td>
<td>1,448,828</td>
<td>8.280187%</td>
<td>$5,028,363</td>
<td></td>
<td>1.00%</td>
<td>-7.28%</td>
</tr>
</tbody>
</table>

- 75% of time spent allocating 0.2% of cost

---

Real Experience. Real Advantage.
Design Decisions – level of detail

- It is tempting to always design applications at the lowest level of detail
- Volumes of data obviously impact performance
- Avoid being “precisely wrong rather than approximately right”

Compare the number of results records to the allocated cost base. Spectrum analysis.
LOADING DATA
Loading Data

- DataBridge
- DataLoader (includes Financial Information Management)

DataBridge loads data through the application tier.
DataLoader loads data directly through the database tier.
Loading Data

DataLoader throughput is about 5 times faster than DataBridge
- More involved to set up (use FIM to simplify)
- Recommended to take the model off line
- Will not create master data as needed
- Not all possible loads are covered (addressed in PCM 10)

Consider DataLoader for loading data when:
- Load time is important
- Volumes are large
- Regular, repetitive process
REPORTING
Reporting – Direct through Books

- Use calculated results rather than triggering calculation through the report
- Reporting from “realized” grid values if faster than reporting on virtual grid values
- CalculateSlice is used to pre-calculate consolidated results

Pre-calculate results

- Use model options to include the reported grid values (or associated base values) in the scope of calculation
- CalculateSlice for consolidated results
Reporting – Direct through MDX O DBO

- Can be no better than through PCM books, so follow hints for direct reporting through books
- MDX “contract” requires all result combinations to be returned all at once, so filter through
  - The enquiry scope
  - Security

- Test out the enquiry through ViewBuilder
- Educate users to limit the scope of their enquiry and apply security to enforce scope limitation.
Reporting - Indirect through an Export

- Exporting is a single threaded process - reference pre-calculated results rather than have the export force a calculation
- The export tables are not optimised for reporting
  - Transfer and restructure into a reporting dedicated DataMart

- Reporting through exports is a cost effective and efficient way of reporting to large user communities
- Expect to invest effort in data restructuring to implement this solution
ASSIGNMENTS
Assignments - Cost Object

Select the relevant Responsibility Centre
- All
- Current
- Parent
- RC Unassigned

- Avoid selecting All for Responsibility Centre for CO assignments
- If this is needed, consider loading the driver by RC and again into RC Unassigned
Assignments – Cost Object

Use focused assignments rather than calculating additional drivers

Avoid calculating additional driver values by using the ability to focus the target of assignments to subsets of the cost objects.
Assignments – Cost Object

- Driver ratios are calculated once and applied to all process/activity costs that use those ratios throughout the model
- Overrides are more “expensive” to process than default assignments

- Reusing an existing driver is more efficient
- Avoid using overrides unless really necessary
Assignments – Resource Driver Assignments

- Overrides are more “expensive” to process than default assignments
- Activity reassignments can be complex to calculate
  - Re-iterative assignments will repeat until convergence
  - Reassigning to All Activities tends to result in a large number of small allocated values

- Avoid using overrides – increased flexibility for assignment in PCM 10 helps with this
- Define reassignments with care
RULES
Rules - When not to use

Internal PCM functionality is more efficient than writing rules

- Focused assignments rather than deriving additional drivers to achieve the same focus
- Consolidations
- Functions (ResourceDriverNonSpare)

- Avoid using rules when an internal PCM facility exists
- Use native PCM functions if available
Rules - Compile

- **Compiled**: use the internal PCM engine and are optimised during compilation
- **Compiled with VBScript**: use the PCM engine and fail over to VBScript for unknown functions
- **VBScript**: use VBScript for calculation

- Switch to compiled rules for optimum performance
- Compiled with VBScript is to help with conversion / migration
- VBScript for backwards compatibility
Rules – Compile

Examples of what the compiler will do during optimisation
- Optimises the use of temporary variables
- Replaces member names with EPO Codes
- Re-orders script to execute most efficiently (ensures dependent items are calculated first)

- Complied rules go through an optimisation stage to make the rule set as efficient as possible.
- Compilation can highlight errors in the rules.
[Rules – Combinations]

Minimise the number of combinations processed
- Restrict: To a fixed subset of the model
- RestrictCombinations: To a subset based on other populated data
- IF ... THEN ... ELSE: Nest to reduce the number of combinations processed as soon as possible

- Models contain a vast number of multi-dimensional combinations. Help the rules engine to consider a minimum of these by employing the provided rule script features.
Rules - RestrictDimension & RestrictCombinations

Function CellValue
    RestrictDimensionParent("Products","PR_Seg1A")
    RestrictDimensionParent("Products","PR_Seg2")
    RestrictCombinationActivityDriverDataValue("ADV002")
    RestrictReturnZero
    If IsChild("Products","PR_Seg1A") Then
        CellValue = ActivityDriverValue(...,"ADV002") * 1.1
    ElseIf IsChild("Products","PR_Seg2") Then
        CellValue = ActivityDriverValue(...,"ADV002") * 1.5
    End If
End Function

- RestrictDimension & RestrictCombinations functions provide optimum rule calculation performance
Rules – Member Comparison

When a rule needs to act differently based on the member

- A Restrict limits the calculation most efficiently
- Compare codes rather than strings
- If comparison includes calculation (for example string manipulation) consider creating a property

Properties are calculated once for each dimension member, far less than the number of calculated combinations.
LARGE NUMBERS OF END USERS
Large Number of Data Entry Users (1000’s)

We occasionally have seen a need to collect small volumes of data from very large numbers of users:

- Data entry model is separated out from calculation model
- Security needs to limit what is available to the users
- Keep the data entry screen as small and simple as possible
- Minimise the handshaking needed – remove results from the view if possible

Data entry for very large numbers of users is possible – with care and attention.
Best Practices

- Have a clear understanding of what performance criteria are most important, and prioritize optimizations to best meet those criteria
- Many factors impact performance, and a holistic approach is essential to balance potentially conflicting objectives
- Design and Deploy with performance objectives in mind
- Consider taking expert advice
Key Learning's

- Consider PCM within the broader system landscape
  - ETL
  - Reporting
- Configure Model Calculation Options appropriately and be prepared to change them as the model usage profile changes over time
- Use DataLoader for loading large volumes of data on a regular basis
- Use internal PCM functionality is preference to rules
- Optimize rules by minimizing the number of combinations considered
Further information

**SAP Public Web:**

- SAP Developer Network (SDN): [www.sdn.sap.com](http://www.sdn.sap.com)

**John Carter**
Director PCM Solution Management
**SAP BusinessObjects Division**
25 King Street, Knutsford, Cheshire, WA16 6DW, UK
T +44 1565 687014
M +44 7787 154923
mailto:john.carter@sap.com
[www.sap.com/sapbusinessobjects](http://www.sap.com/sapbusinessobjects)
Thank you for participating.

Please remember to complete and return your evaluation form following this session.

For ongoing education in this area of focus, visit www.asug.com.

SESSION CODE: 1906
Return on Investment

- If applicable, highlight any aspect of your presentation that can contribute to a significant ROI. Omit slide from presentation if necessary.