



Configuration Guide

SAP Business Planning and Consolidation 7.0 SP03

version for the Microsoft Platform

Target Audience

- Technical Consultants
- System Administrators

PUBLIC

Document version: 2.0 – 02/13/2009

Copyright

© Copyright 2009 SAP AG. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.

Microsoft, Windows, Excel, Outlook, and PowerPoint are registered trademarks of Microsoft Corporation.

IBM, DB2, DB2 Universal Database, System i, System i5, System p, System p5, System x, System z, System z10, System z9, z10, z9, iSeries, pSeries, xSeries, zSeries, eServer, z/VM, z/OS, i5/OS, S/390, OS/390, OS/400, AS/400, S/390 Parallel Enterprise Server, PowerVM, Power Architecture, POWER6+, POWER6, POWER5+, POWER5, POWER, OpenPower, PowerPC, BatchPipes, BladeCenter, System Storage, GPFS, HACMP, RETAIN, DB2 Connect, RACF, Redbooks, OS/2, Parallel Sysplex, MVS/ESA, AIX, Intelligent Miner, WebSphere, Netfinity, Tivoli and Informix are trademarks or registered trademarks of IBM Corporation.

Linux is the registered trademark of Linus Torvalds in the U.S. and other countries.

Adobe, the Adobe logo, Acrobat, PostScript, and Reader are either trademarks or registered trademarks of Adobe Systems Incorporated in the United States and/or other countries.

Oracle is a registered trademark of Oracle Corporation.

UNIX, X/Open, OSF/1, and Motif are registered trademarks of the Open Group. Citrix, ICA, Program Neighborhood, MetaFrame, WinFrame, VideoFrame, and MultiWin are trademarks or registered trademarks of Citrix Systems, Inc.

HTML, XML, XHTML and W3C are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Massachusetts Institute of Technology.






Java is a registered trademark of Sun Microsystems, Inc.

JavaScript is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.

SAP, R/3, xApps, xApp, SAP NetWeaver, Duet, PartnerEdge, ByDesign, SAP Business ByDesign, and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other product and service names mentioned are the trademarks of their respective companies. Data contained in this document serves informational purposes only. National product specifications may vary.

These materials are subject to change without notice. These materials are provided by SAP AG and its affiliated companies ("SAP Group") for informational purposes only, without representation or warranty of any kind, and SAP Group shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP Group products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

Icons in Body Text

Icon	Meaning
	Caution
	Example
	Note
	Recommendation
	Syntax

Additional icons are used in SAP Library documentation to help you identify different types of information at a glance. For more information, see *Help on Help* → *General Information Classes and Information Classes for Business Information Warehouse* on the first page of any version of *SAP Library*.

Typographic Conventions

Type Style	Description
<i>Example text</i>	Words or characters quoted from the screen. These include field names, screen titles, pushbuttons labels, menu names, menu paths, and menu options. Cross-references to other documentation.
Example text	Emphasized words or phrases in body text, graphic titles, and table titles.
EXAMPLE TEXT	Technical names of system objects. These include report names, program names, transaction codes, table names, and key concepts of a programming language when they are surrounded by body text, for example, SELECT and INCLUDE.
Example text	Output on the screen. This includes file and directory names and their paths, messages, names of variables and parameters, source text, and names of installation, upgrade and database tools.
Example text	Exact user entry. These are words or characters that you enter in the system exactly as they appear in the documentation.
<Example text>	Variable user entry. Angle brackets indicate that you replace these words and characters with appropriate entries to make entries in the system.
EXAMPLE TEXT	Keys on the keyboard, for example, F2 or ENTER.

SAP BPC 7.0 SP03 M Configuration Guide	5
Server Configuration	5
Modifying Machine.config Files.....	5
Setting COM+ Object Application Pooling Pool Size	7
Setting Memory Recycling in the IIS Application Pool	7
Stored Configuration Values	8
Partitioning	9
Considerations when Determining a Partitioning Scheme	12
Data Slice Definition	13
Partition Planning.....	15
Fact Table Partitioning.....	16
Issues with Versions prior to 4.2 SP2 QFE2.....	18
Tuning Business Planning and Consolidation	18
Hardware Configuration	20
Operating System Configuration.....	21
System Software Configuration	22
BPC Parameters.....	25
Application Customizations.....	26
Application Design	27
Application Maintenance.....	31
Avoid Using Remote Desktop Connections to Application Server.....	33
Appendix A: Identifying Performance Bottlenecks.....	33
Appendix B: FACT, FAC2, and WB Tables	36

SAP BPC 7.0 SP03 M Configuration Guide

The SAP BPC 7.0 SP03 M Configuration Guide contains the following topics for configuring Business Planning and Consolidation for proper and reliable functioning:

- [Server Configuration](#)
- [Partitioning](#)
- [Tuning Business Planning and Consolidation](#)
- [Avoid Using Remote Desktop Connections to Application Server](#)

Server Configuration

You configure the Business Planning and Consolidation .NET Web and Application server to ensure that it and its related services function properly and reliably.

Procedure

- You modify the machine.config file.
This ensures that there is a balance between the number of threads “working” requests and deadlock occurrences that would prevent the application server from processing. See [Modifying Machine.config Files](#).
- You set the application pooling pool size for Everest Update, OSoftSystemConfig, K2Processing, and OsoftDataService. See [Setting COM+ Object Application Pooling Pool Size](#).
- You configure Internet Information Services (IIS) to set memory recycling in an application pool. See [Setting Memory Recycling in the IIS Application Pool](#).

Modifying Machine.config Files

To ensure that the right balance is struck between the number of threads “working” requests and deadlock occurrences that would cause the application server to “sleep” and not process anymore, you can modify the machine.config file, which resides in the \WINDOWS\Microsoft.NET\Framework\v1.1.4322\Config\ directory.

Procedure

1. Set both the *<maxWorkerThreads>* and *<maxIOTheads>* to 100 as follows:

```
<ProcessModel  
    , , , , (Other settings)  
    maxWorkerThreads="100"  
    maxIoThreads="100"  
>
```

2. Adjust *<minFreeThreads>* such that its value is equal to that of multiplying the factor 88 to that of the number of CPUs on the server (i.e., *minFreeThreads=88 x # of CPU*).

Similarly adjust *<minLocalRequestFreeThreads>* such that its value is equal to that of multiplying the factor 76 to that of the number of CPUs on the server (that is, *minLocalRequestFreeThreads=76 x # of CPUs*).

Finally, make sure that the value of *<appRequestQueueLimit>* is increased to 500.

Assuming a server with two processors, here is an example:

Example

```
<httpRuntime  
    minFreeThreads="176"  
    minLocalRequestFreeThreads="152"  
    appRequestQueueLimit="500"  
>
```

End of the example.

3. Set *<machineKey>* options properly.

The *<machineKey>* tag in *web.config* or *machine.config* tells the .NET framework how to create the hash on forms tickets (which is the proof that you have been authenticated). The default is for each web application to create its own hash key. So, if you took your authentication cookie created on one machine and try to access another machine, it would fail when checking the hash code to make sure it hadn't been tampered.

Therefore, if you are using a web farm (multiple web servers where you can automatically be routed to any one of them within a particular session), the hash key needs to be the same on all the machines. This way, when the authentication cookie is checked, the authentication ticket used on one web server can still be valid on another server. Note that this is a requirement for the web servers, not the application server. You can specify it for just the BPC IIS application in its *web.config* file; you do not have to specify it for all IIS applications in the general

machine.config file. If you are not using a web farm, you do not need to change the *<machineKey>* element.

4. If using a deployment where the web servers do not reside on the same servers as the Application servers, adjust HTTP TCP/IP connections settings in the machine.config file of the web server(s) as follows:

```
<connectionManagement>

<add address="app server ip" maxconnection="48"/>

<add address="*" maxconnection="2"/>

</connectionManagement>
```

Setting COM+ Object Application Pooling Pool Size

For load balancing purposes, set the application pooling pool size for Everest Update, OSoftSystemConfig, K2Processing, and OsoftDataService applications.

Procedure

1. Choose *Start Administrative Tools Component Services COM+ applications* .
2. Right-click the application.
3. Choose *Properties*, then on the *Pooling & Recycling* tab, change the application pool size to 4.

Setting Memory Recycling in the IIS Application Pool

To run large requests, you configure Internet Information Services (IIS) to restart a worker process in an application pool so that it is recycled after using a set amount of memory.

Activities

To configure a worker process to be recycled, take the following steps:

1. Start Internet Information Services (IIS) Manager on the Application server.
2. Expand the local computer, and expand *Application pools*.
3. Right-click the application pool that contains the Osoft virtual directory, then select *Properties*.

4. From the *Recycling* tab, select the *Maxium used memory (in megabytes)* check box, and enter 1800 (the maximum amount), then select *OK*.

More Information

See

<http://www.microsoft.com/technet/prodtechnol/WindowsServer2003/Library/IIS/1eee28e2-b319-4b4e-8267-a8c0aa0dcf36.mspx?mfr=true>

Stored Configuration Values

The following values are modified as part of the installation process:

Application Server Registry Entries

```
- <ConfigStores Path="..\WINDOWS/system32/inetsrv/">
    <ConfigStore Name="MetaBase.xml" Type="xml" Alias="IIS MetaBase" />
</ConfigStores>

- <ConfigStores
Path="..\WINDOWS/Microsoft.NET/Framework/v1.1.4322/CONFIG/">
    <ConfigStore Name="machine.config" Type="xml" Alias="Machine Config"
/>
</ConfigStores>

- - <ConfigStores Path="Server Management/">
    <ConfigStore Name="OSoftInstall.xml" Type="xml" Alias="OSoft Install"
/>
</ConfigStores>

- <ConfigStores Path="Websrvr/bin/">
    <ConfigStore Name="OutlookSoft.config" Type="xml" Alias="OSoft Config"
/>
</ConfigStores>

- <ConfigStores Path="Websrvr/Web/">
    <ConfigStore Name="Web.config" Type="xml" Alias="Web Config" />
</ConfigStores>
```

Web Server Registry Entries

```
<Path Name="C:\WINDOWS\system32/inetsrv/" />

<ConfigStore Name="MetaBase.xml" Type="xml" Alias="IIS Metabase" />
```



```
<Path Name="C:/WINDOWS/Microsoft.NET/Framework/v1.1.4322/CONFIG/" />
<ConfigStore Name="machine.config" Type="xml" Alias="Machine Config" />
<Path Name="Server Management/" />
<ConfigStore Name="OsoftInstall.xml" Type="xml" Alias="BPC Install" />
<Path Name="Websrvr/bin/" />
<ConfigStore Name="*.dll" Type="jardll" Alias="BPC DLLs" />
<Path Name="Websrvr/Web/" />
<ConfigStore Name="Web.config" Type="xml" Alias="Web Config" />
<Path Name="Websrvr/bin/" />
<ConfigStore Name="bin/*.dll" Type="jardll" Alias="BPC DLLs" />
<ConfigStore Name="OutlookSoft.config" Type="xml" Alias="BPC Config" />
```

Partitioning

You can use partitioning to significantly reduce the overall processing time for medium and large cubes.

Note

Significant portions of the following information has been copied in “fair use” from Microsoft web sites.

.

Features

Partitioning is perhaps the single most significant thing you can do to improve performance in a large application.

Dividing a cube into multiple partitions reduces total processing time for the following reasons:

- Multiple partitions can be processed in parallel.

You can decrease the total time required to process a cube by processing multiple partitions in parallel (provided you have sufficient processor and memory resources). By default, Analysis Services processes each partition in a cube serially. For large cubes, parallel processing results in dramatic performance benefits in the following cases:

- During the initial load of the data warehouse

- During full cube processing
- During cube refreshes

Note

If the Analysis Services server has insufficient memory to store the aggregations for each partition being processed, Analysis Services uses temporary files, which negates the performance benefit you are trying to achieve through the use of parallel processing.

End of the note.

To process partitions in parallel, you must use a tool that calls the Decision Support Objects (DSO) interface. For an example, see the Parallel Processing sample application in the SQL Server 2005 Resource Kit.

- Only some partitions will need to be processed.

You can process only the partitions that have been updated with new data. This will decrease the overall time required to update a cube with new data. Analysis Services can process one or several small partitions more quickly than it can process a single large partition containing all of the data for the cube.

Example

If you partition your cube by time, you merely need to process the partitions containing data from the most recent time period, rather than processing the entire cube.

End of the example.

- Different partitions can have different aggregation designs.

You can design additional aggregations on heavily queried partitions and design fewer aggregations on less heavily queried partitions. If you keep the number of the partitions with a large number of aggregations small, and design fewer aggregations for those partitions that are queried less frequently, you can reduce overall processing time. However, if you plan to merge partitions at a later date, you must ensure that the partitions being merged have the same aggregation design at the time of the merger.

- Different partitions can have different storage modes.

You can use different storage modes for particular purposes.

Example

You can create a small partition using ROLAP to enable you to implement real-time OLAP, and then use MOLAP for all other partitions to maximize query responsiveness.

End of the example.

- Partitions can be refreshed individually.

You can refresh a partition more quickly than an entire cube, thereby consuming fewer resources and affecting fewer users. When a partition is incrementally updated, a temporary partition is created and then merged into the existing partition. This can result in data fragmentation, which is similar to disk or index fragmentation. As a result, you can occasionally do a refresh of a partition to enable Analysis Services to re-sort the data for faster multidimensional access, create better multidimensional mapping files, and make smaller aggregations.

The division of a cube into multiple partitions is transparent to the user. When a query requests data that spans multiple partitions, Analysis Services uses aggregations or scans data in each partition and then combines this information to resolve the query and return the results to the user. You can significantly increase query responsiveness and processing performance by horizontally segmenting the data by one or more keys, such as date or department, and dividing the cube into multiple partitions.

From an operational perspective, you can improve Analysis Services performance by keeping partition sizes reasonable and setting partition data slices. For more information on each of these configuration issues, see the *Microsoft Analysis Services Performance Guide* on www.microsoft.com/technet.

Each partition file has an extension of *fact.data*. When a partition file exceeds 5 GB or more than 20 million records, you should consider dividing the partition into multiple partitions. While the above parameters are useful as guidelines, these are not the only considerations. As a rule, however:

- Smaller partitions can be processed faster than larger partitions.
- When data changes, you do not need to process all of the partitions in the cube.
- When data changes, you do not need to process all of the partitions in the cube.
- Queries can be executed faster on smaller partitions if the data slice is set properly. In this scenario, Analysis Services would not need to scan as much data to resolve multiple queries.

Performance degradation typically occurs when the cache is repeatedly refreshed from the disk. This becomes especially problematic when there are large numbers of alternate hierarchies, since the time needed to reprocess each partition varies directly with the number of hierarchies.

More Information

[Considerations when Determining a Partitioning Scheme](#)

[Data Slice Definition](#)

[Partition Planning](#)

[Fact Table Partitioning](#)[Issues with Versions prior to 4.2 SP2 QFE2](#)

Considerations when Determining a Partitioning Scheme

Consider the aspects described in this topic when determining a partitioning scheme.

Features

Data Usage

The primary factor that you should use to determine your partitioning scheme is your data usage. Partitions can be sliced by any number of dimensions.

Example

- Time
- Entity
- Category
- Some combination of dimensions

End of the example.

Performance

In addition to the data size considerations, partition so that the data slices created allow a single partition to be accessed for logic, reports, etc. In many cases this makes the Time dimension a natural choice for partitioning and slices. But this is not necessarily always the case.

You can create partitions across multiple dimensions; TIME and ACCOUNT, for example. Because it may be necessary to access multiple partitions for certain tasks, this approach may improve performance in some cases and degrade it in others.

In many cases, partitioning just by TIME (without partitioning by any other dimensions) is the most effective strategy. Be sure to measure the results of your strategy. In one real-world case, a partitioning scheme that used CATEGORY and CURRENCY did not improve performance over the default.

There is some evidence to indicate that multi-dimension partitioning does not seem to improve performance by as great a margin as the implementation of an initial, single-dimension scheme (in other words, it is common to experience diminishing returns)

Administrative Implications

In addition to performance, you need to consider the administrative implications of your partitioning scheme:

- Balance ease of administration with scheme complexity: If the BPC administrator needs to spend hours every week repartitioning, it may not be the optimal solution.
- Time period modifications necessitate partition re-creation: When you create a partition according to time slices, if you want to modify, shift or roll-up time periods, you must re-create the partitions accordingly.

Caution

Allow for the complete processing of dimensions before processing any partitions based on that dimension.

End of the caution.

- Ensure completion of dependent partition processing: When processing partitions, you must ensure that a dimension is completely processed prior to processing any partitions based on it. You may experience negative consequences if you process a partition while one of its dimensions is also being processed.

Data Slice Definition

Before you can partition, you must define in detail the data slice for each partition.

Features

If the data slice value for a partition is set properly, Analysis Services can quickly eliminate irrelevant partitions from the query processing and significantly reduce the amount of physical I/O and processor time needed.

To enable Analysis Services to take full advantage of partitions, you must define the data slice for each partition in the Partition Wizard of Analysis Manager. The data slice identifies the actual subset of data contained in each partition. The Partition Wizard does not require you to set this data slice when you create a partition. As a result, it is possible to create a partition without setting the data slice.

Caution

Always set the data slice when creating the partition. Failure to do so can result in the addition of considerable system overhead (artificially increasing response times).

End of the caution.

Without the data slice, Analysis Services cannot limit a query to the appropriate partitions and must scan each partition even if zero cells will be returned. To draw an analogy with

SQL Server, creating a partition without a data slice is like creating a partitioned view without the CHECK clause. While you can do it, you force the query optimizer to scan all of the partitions in the view because you haven't given it enough metadata to figure out what partition to access when a query is issued. While the Analysis Services runtime engine does not use a relational query optimizer (it has its own component that accomplishes a similar operation), it uses the data slice in roughly the same way: as metadata to tell it which partitions to scan if an aggregate cannot be used or is not available.

If you partition a cube by month, and have 36 months worth of data (in 36 partitions), and if you don't specify the data slice, then the runtime engine must scan all 36 partitions to answer a query. If you specify the data slice, it could potentially only have to scan 1/36th the data, with an obvious improvement in performance.

To maximize querying performance with partitions, construct partitions with data slices that mirror the data slices required by your users. For example, suppose you are deploying a cube that typically tracks data as a time series (such as a financial cube), and most queries will retrieve data based on time period. You should partition the cube by time period to provide the greatest performance benefit. With very large cubes, partitioning along multiple dimensions (such as time and department) can yield substantial query responsiveness benefits. Remember that each partition can have a different aggregation design.

Recommendation

If you are creating rolling monthly partitions as each month closes, you should ensure that the data slice is set for each new partition after it is created.

End of the recommendation.

Data Slices Result in the Creation of JOIN and WHERE Clauses

Setting a data slice also causes Analysis Services to add a JOIN and a WHERE clause to the SQL statement used for retrieving data from the source database during processing. The WHERE clause limits the data retrieved by the SQL statement to the data that belongs in the data slice.

Example

If you say that a partition's data slice is June 2008, then Analysis Services adds a join to the time dimension and adds the WHERE clause:

```
WHERE <month field> = 'June' AND <year field> = '2008'
```

End of the example.

or whatever the appropriate member/level names are. If you do not define a data slice and you have multiple partitions, Analysis Services does not restrict the data that is retrieved from the source database. Without the data slice, if you just happen to have July 2008 data in the June partition, Analysis Services does not complain, it just double-counts the

July 2008 data. For more information, see *Maintaining Partitions* in SQL Server Books Online. By specifying the data slice, the system can add these JOIN and WHERE clauses that assist in maintaining the integrity of the data.

Note

We do not recommend changing the DataCompressionSettings registry settings.

Partition Planning

Partition planning ensures that you can reproduce and verify the partitioning scheme.

Prerequisites

You have chosen your basic scheme and determined the manner in which you'd like to slice your data.

Features

You define the partition specifications in a table like the one below.

Note

Ensure that the Fact prefix is part of every partition name. It is important to prepend Fact to each partition name, as *Save Application* follows the naming convention:

```
Fact<ApplicationName>]####
```

The following tables contain examples of the planning that you should complete prior to creating your partitions for various fact tables. Preparing this information in this format greatly decreases the amount of time that you need to create the partitions.

The following table contains planning details for the dbo.tblFactFINANCE table.

<i>Partition Name</i>	<i>Slice</i>	<i>Filter (The "Where" Clause)</i>	<i>Storage Mode</i>
FACTFINANCE2002	2002.TOTAL	(dbo.tblFactFINANCE.TIMEID >= '20020100') AND (dbo.tblFactFINANCE.TIMEID<= '20021200')	MOLAP
FACTFINANCE2003	2003.TOTAL	(dbo.tblFactFINANCE.TIMEID >= '20030100') AND (dbo.tblFactFINANCE.TIMEID<=	MOLAP

		'20031200')	
FACTFINANCE2004	2004.TOTAL	(dbo.tblFactFINANCE.TIMEID >= '20040100') AND (dbo.tblFactFINANCE.TIMEID<= '20041200')	MOLAP
FACTFINANCE2005JAN	2005.TOTAL.2005.Q1. 2005.JAN	dbo.tblFactFINANCE.TIMEID = '20050100'	MOLAP
FACTFINANCE2005FEB	2005.TOTAL.2005.Q1.2005.FEB	dbo.tblFactFINANCE.TIMEID = '20050200'	MOLAP
FACTFINANCE2005MAR	2005.TOTAL.2005.Q1. 2005.MAR	dbo.tblFactFINANCE.TIMEID = '20050300'	MOLAP
FACTFINANCE2005Q2	2005.TOTAL.2005.Q2	(dbo.tblFactFINANCE.TIMEID >= '20050400') AND (dbo.tblFactFINANCE.TIMEID<= '20050600')	MOLAP
FACTFINANCE2005Q3	2005.TOTAL.2005.Q3	(dbo.tblFactFINANCE.TIMEID >= '20050700') AND (dbo.tblFactFINANCE.TIMEID<= '20050900')	MOLAP
FACTFINANCE2005Q4	2005.TOTAL.2005.Q4	(dbo.tblFactFINANCE.TIMEID >= '20051000') AND (dbo.tblFactFINANCE.TIMEID<= '20051200')	MOLAP

The following table contains planning details for the dbo.tblFAC2FINANCE table.

<i>Partition Name</i>	<i>Slice</i>	<i>Filter (The “Where” Clause)</i>	<i>Storage Mode</i>
FAC2FINANCE	All	[no filter]	MOLAP

The following table contains planning details for the dbo.tblFACTWBFINANCE table.

<i>Partition Name</i>	<i>Slice</i>	<i>Filter (The “Where” Clause)</i>	<i>Storage Mode</i>
FACTWBFINANCE	All	[no filter]	ROLAP

Fact Table Partitioning

In Business Planning and Consolidation (BPC), the FINANCE cube has a number of partitions as part of the default installation. Of those, the FINANCE partition is the only one you need to address, by deleting that partition and replacing it with partitions that divide the data into smaller segments.

Procedure

1. Start SQL Server Business Intelligence Development Studio, which is part of the Microsoft SQL Server 2005 suite.
2. Choose *File Open Analysis Services Database* .
3. In the *Connect to Database* dialog box:
 - Choose *Connect to existing database*.
 - Enter the server name.
 - Choose the database.

4. Choose *OK*.

The Microsoft Visual Studio 2005 screen appears.

5. In the context menu of the FINANCE cube, which is located in the CUBES section of the Solution Explorer pane, choose *Open* to open a tab displaying the default FINANCE cube partitions.
6. Delete the FINANCE partition.
7. Choose *New Partition*.

The Partition Wizard dialog box appears.

8. In the *Specify Source Information* dialog box:
 - Choose *Finance* as the Measure group.
 - Choose *Data Source View.AppDef* as the Look in.
 - Select *dbo.tblFactFinance* as the table to use.
9. Choose *Next*. In the *Restrict Rows* dialog box:
 - Choose *Specify a query to restrict rows*.
 - Enter a query. For more information, see *Partition Planning*.
 - Choose *Check* to verify the validity of the query.
 - Choose *Next*. In the *Completing the Wizard* dialog box, choose *Design aggregations later*.

- Choose *Finish*.

10. Repeat steps 7 through 9 for each partition you need to create.

Issues with Versions prior to 4.2 SP2 QFE2

The following are issues with versions prior to 4.2 SP2 QFE2.

Integration

OLAP Partition Changes – Administrative Impact

There are significant implications for the creation of these partitions on administrative processes. SAP has not had an opportunity to test all possible combinations of events. As such, we recommend limiting administrative activity as much as possible during the month end.

- Administrative tasks *known* to be problematic with the partitioning scheme:
 - Optimizing (full process and incremental optimization) will be a problem, as the administrative utilities assume the default partitioning scheme. This will have to be customized.
 - Saving applications. This depends on the default partitioning scheme, and this will be problem. Do not save the application.
- Administrative tasks *suspected* to be problematic with the partitioning scheme:
 - Adding a member to a dimension. This is not likely to be problematic, but this process has not yet been fully certified.
 - Drillthrough is implemented by dimension, not partition, and should therefore be unaffected. You should test this in your implementation.
- Administrative tasks *not affected* by the partitioning:
 - Security: V&P security will be unaffected.
 - Lite optimization will be unaffected.

Tuning Business Planning and Consolidation

This section provides guidance and techniques for developing high-performance applications by explaining what can be done to tune each layer, from the hardware up through the system software to the application and the reports.

Features

The practices in this section will help to provide you with a solid understanding of your application and a foundation for a high performance solution. Its overall organization is as a checklist. By working through this section from beginning to end, you can determine which best practices have been applied and which have not.

Note

Not all techniques are necessary, or even appropriate, for all situations. Also, the concepts in this section have not been tested in all scenarios and your results may vary.

The Tuning Layers

The layers addressed are:

- Hardware
- Operating System
- SQL Server
- Analysis Services
- AppSet database
- Data load processes
- Reports and Input Schedules
- Application Maintenance

Tuning - Minimum Consideration

If nothing else, you should focus on the following:

- Get big servers
- Eliminate HTTP connections
- Use EVDRE for Excel templates

More Information

[Hardware Configuration](#)

[Operating System Configuration](#)

[System Software Configuration](#)

[BPC Parameters](#)

[Application Customizations](#)

[Application Design](#)

[Application Maintenance](#)

[Appendix A: Identifying Performance Bottlenecks](#)

[Appendix B: FACT, FAC2, and WB Tables](#)

Hardware Configuration

Use the following guidelines to configure your BPC hardware.

Prerequisites

Hardware configuration can be very complex, and its relationship to performance may not be readily apparent. You should engage technical consulting to ensure all details are in order.

Features

Your review should touch on all these points

How many servers?

While your performance profile may vary, for large applications you should have at least three servers: A web, application and database server (SQL box and an OLAP). You may consider load balancing of the web and application tier. If possible a forth server is recommended, by splitting the OLAP and SQL components on different servers will provide the most scalable solution.

In release 4.2, the web and application tier could not be separated. When testing 4.2, we had found that having 1 web/app server with about 300 users accessing it at one time, we would max out the web server and started getting connectivity error messages. While we found the breaking point at 600 users for a single server, we found 300 users took a while to get to the breaking point.

With release 5.x, the web and application tier have been split allowing better scalability. By splitting the web from the application tier there are additional options to scale out both or either tier via load balancing. The appropriate configuration can only be determined when the application design is known. Working with the SAP resources will provide the experience to assist.

Another test on release 4.2, with 4 hyperthreaded (8 virtual) 3.5 GHz CPUs, multiple high speed I/O channels, and 6 GB of RAM, SQL and AS are both on the same server. We have good performance on this server, as many of the best practices in this document have been implemented. Similar results are expected with release 5.x.

What's in the server?

It is no secret that screaming fast hardware will typically provide the biggest performance boost per dollar spent. In particular:

- Lots of really fast CPUs, dual core or hyperthreaded
- Enough RAM to hold the entire database and all processes
- High speed, properly configured, I/O devices

For any customer purchasing hardware today and concerned about performance, they should consider no less than:

- Four dual core 3GHz or faster CPUs
- 8 GB RAM Minimum (16 or 32 is better)
- Three or four separate RAID drives and controllers for operating system and files, data, temp and log files

Don't forget the clients

Because BPC is a client application, client workstations should have as much RAM as possible. Informal testing showed that upgrading a client machine RAM could reduce complex expansion and report times by over 50%.

Recommendation

In case the clients are using Citrix on shared hardware, please contact technical consulting.

End of the recommendation.

Operating System Configuration

Use the following guidelines to configure your operating system.

Prerequisites

You should be running Windows 2003 Enterprise Edition to take advantage of large memory and multiple CPUs.

Memory Configuration

PAE or Physical Address Extension is an operating system memory switch that allows Windows to effectively address memory in excess of 4 GB. Details are beyond the scope of this document; you should make sure all memory is addressable.

Disk Configuration

In general, you should have data, logs, tempdb and the OS on separate physical drives.

Virus Scanning Software

Make sure Virus scanning is limited to the files absolutely needed (so don't virus scan the transaction log or the reads/writes done by SQL and Analysis Services).

Setting Connection Timeouts (IIS 6.0)

Connection time-outs help reduce the loss of processing resources consumed by idle connections. When you enable connection time-outs, IIS enforces the time-outs at the connection level. See the procedure below.

Procedure

To set a global WWW or FTP service connection time-out value:

1. In IIS Manager, expand the local computer, right-click the Web Sites or FTP Sites folder, and click *Properties*.
2. On the Web Site or FTP Site tab, in the *Connection timeout* box, type the maximum number of seconds that IIS should maintain an idle connection before resetting the connection.
3. For the WWW service, verify that the *Enable HTTP Keep-Alives* box is selected.
4. Click *Apply*, and then click *OK*.

System Software Configuration

Use the following guidelines to configure your system software. System software refers to the SQL, Analysis Services, and other layers above the Operating System but not including BPC and the application itself.

Features

SQL Server Configuration

Make sure you are running the SAP supported SQL Server version for both SQL and OLAP (see the *SAP BPC 7.0 M Master Guide*).

Make sure AWE is configured for the large amount of memory you've installed in the hardware.

If disk I/O is an issue, look into using file separation techniques.

Analysis Services Configuration for SQL 2000 only

The following procedure sets up Analysis Services so that BPC can give you greatly improved performance on data sends and retrieves:

1. On the server running Analysis Services, choose *Start Programs Microsoft SQL Server Analysis Services Analysis Manager* .
2. Expand the Analysis Servers folder, then right-click on Analysis Server, and choose *Properties*.
3. Display the performance and memory settings.
4. Change these values based on your server setup:

Setting	Recommend value
Maximum number of threads	Two times the number of physical processors in the system (not including hyperthreading).
Large level defined as	Maximum number of levels in the largest dimension, or 10,000 if the largest dimension has more than 10,000 members.
Minimum allocated memory	One-half of the server's RAM not to exceed 3GB (3072)
Memory conservation threshold	Maximum of 2047MB for 2GB of RAM or Maximum of 2.7GB if set for 3GB with the 3GB switch with 4GB of RAM (Enterprise edition OS)

5. These parameters assume Analysis Services is installed alone. If Analysis Services is installed with other base components please contact an SAP technical consultant for proper consideration.
6. The max number of members Large Level... is by default set to 1,000. To be precise, this relates to the number of members in a single level. It will not hurt to set this to at least the number of members in your largest dimension, or 10,000, whichever is less.
7. Choose OK, then close Analysis Manager.

Analysis Services Configuration for SQL 2005 only

We have found the majority of the setting for SQL 2005 should follow the Microsoft documented guidance. We have found recently that Query/MaxThread value should be 10 (default value) plus the total number of BPC databases (AppSets) plus 2 times the number of CPUs. Without making the appropriate setting, processing time will be significantly slower. For more details contact the SAP support team.

Operating System Services

Turning off services that are not required will save memory and CPU. The following is a list of services that you do not need on the BPC server.

- Alert
- Application Management Transfer service
- ClipBook
- Computer Browser
- Distributed Link Tracking client
- Distributed Transaction Coordinator
- Fax Service
- Indexing Service
- Internet Connection Sharing
- Logical Disk Manager Administrative Service
- Messenger
- Microsoft NetMeeting Remote Desktop Sharing
- Network DDE
- Performance logs and alerts
- Protected Storage
- QoS RSVP
- Remote Access Auto Connection Manager
- Remote Access Connection Manager
- Remote Procedure Call (RPC) locator
- Routing and Remote Access
- RunAs service
- Security Account Manager
- Server
- SmartCard
- SmartCard Helper
- System Event Notification

- TCP/IP NetBIOS Helper Service
- Telephony
- Telnet
- Uninterruptible Power Supply
- Utility manager
- Windows Installer
- Windows Time

You should work closely with technical consulting and your IT team to ensure that these services are not actually in use by other processes on the server, and that their disabling will not adversely affect functionality or performance.

BPC Parameters

Use the following guidelines to set BPC parameters.

Features

SAP BPC Send Governor Configuration Modifications

Some background on what the Send Governor (SG) accomplishes, so that you can understand better how this works. The SG is designed to manage the Microsoft Analysis Services locks. This ensures consistent performance for the user and avoids deadlocks.

Send Governor values to set in tblDefaults are:

Send Governor Settings (KeyID)	Default	Batch mode	Real time mode
THREAD_MAXNUM_SG:	3	0	Increase
INTERVAL_CHECK_SEND:	3000 msec	Increase	Decrease
MAXCELLS_THREAD_SG:	1000000	Increase	Decrease
UNITPER_SP:	1000000	Increase	Decrease

For an AppSet which will have a lot of concurrent send activity, you should consider adding an additional 1 to 2 seconds (on average) during sends in order to help improve data read and lite optimize (specific function of BPC administration) performance.

Note

The time to do a Send from BPC for Excel includes both the time to “send” plus (on average) half the Interval_Check_Send time.

We saw a significant drop in locks by decreasing the THREAD_MAXNUM_SG to 1:

A customer quote:

“As soon as I saw more than 35 users concurrent I could see the locks higher than 10 for more than 20 seconds. I tried switching the thread_maxnum_sg to 1 and immediately (after about 10 sec) the locks dropped....At that moment the wbtrigger was at 10k rows. Noticing this I tried putting the wbtrigger to 20K and did not notice any huge locks anymore...”

“During concurrency of about 70 users all remained well...”

“Seeing this I would need more time for testing with even higher wbtriggers, at this stage it looks like the thread_maxnum_sg parameter is making the real difference.”

Application Customizations

Use the following guidelines to customize your BPC applications.

Features

A core area of planning for large applications consists of knowing what data will flow into which of the FACT, FAC2 and WB tables when, and how. By proactively managing this flow, you are taking the most significant step to ensuring a smooth performance profile for your applications. [Appendix B: FACT, FAC2, and WB Tables](#) provides a little more background on the Fact tables and how they are used.

Consolidations Package Changes

Intercompany Package

You should configure the package to write to FAC2 not to WB.

Currency Translation Package

Currency translation will generate a lot of records in the database. These can be created in the WB table by default, but this can result in (worst case) millions of records in the WB table, which would create a serious performance issue. At one customer in Europe, the currency translation process generated 2.5 million WB records.

The package should be configured to insert into the FAC2 table. The default FXTRANS logic can also be modified to performance this function.

Note

Both packages can support the insertion of records into FACT after which we can process just the FACT partition. This would be seriously considered for a very large production environment where the FAC2 table will be heavily used by other processes.

Data Manager Package Changes

As mentioned above, smart use of the WB, FAC2 and FACT tables is essential.

By default BPC import package writes into FAC2 and triggers a processing of the associated partition. When many users concurrently import their data, all the processing will easily lock the system. This situation is exacerbated if other processes are being run, such as FX conversion or Intercompany eliminations.

This condition has been particularly acute when users are allowed to run packages whenever they choose, often resulting in a large number of submissions in a very short period of time (at budget or closing deadline, naturally).

The following techniques can be used to mitigate the performance impacts:

- You can import into WB table if you are importing so frequently that locking has become an issue, especially if the imported files are relatively small. A scheduled light optimization will take care of periodically emptying the WB.
- “Throttling” the throughput. In a customer case, the submitted files are not immediately imported, but are put in a queue. A separate process merges the files and imports the merged file. This process is both more efficient and reduces locks.
- You should also look to use bulk data loads into the FACT table if there is a maintenance window which allows for such loading.

Application Design

Use the following guidelines to design your applications for maximum performance.

Dimensions

The UNARYOPERATOR property should be avoided if at all possible.

The choice of concurrent dimension is key. BPC locking is by three to five dimensions: for example a customer uses the Entity, Category and Time type for the other two locking dimensions. The combination of Entity, Category and Time is locked by BPC when data is sent. (This is one of several kinds of locks in the system: SQL has locks, Analysis Services has locks, and BPC has locks. Here we are talking about BPC Locks).

Example

Budgeting and forecasting by function – Sales, Marketing, IT, Legal: A Function dimension is secured and used for this purpose. When a user chooses *Send* he sends to all entities, so other users are locked out until that send ends. The sends queued up quickly and response time might increase from a minute to 10 or 20 minutes. The solution is to change the concurrent dimensions to the function dimension instead of the entity type dimension. Therefore the legal entity dimension is not considered in the lock (albeit still a secured dimension).

End of the example.

Alternative hierarchies in BPC show dimension processing to be faster than 4.x since it uses a different schema. However query performance will still be impacted. Consideration of the need for alternative hierarchies is necessary to ensure optimal performance for reporting.

More dimensions create more complex joins and retrievals. If you can break a large cube up into “business process specific” cubes with simpler dimensionality, you can gain performance (and usability) improvements. This is anecdotal.

Reports and Input Schedules

General Guidelines

When building or reviewing sheets:

- See how much data is to be entered: many small sheets vs. one big workbook have advantages and disadvantages and should be discussed.
- Avoid refreshing after sends if possible. Use Excel formulas to calculate totals instead of retrieving them. This makes the process even more real-time.
- Use park-n-go if applicable. This discourages multiple send and refresh.
- Try to avoid using multiple EVGETs in a single cell. (i.e. EVGETs + EVGETs)
- Try to avoid having the dimension parameter of an EVGETs formula depend on the result of another EVGETs formula. See reporting best practices on the Corporate Performance Management Community on SDN (<https://www.sdn.sap.com/irj/sdn/bpx-cpm>)

Asymmetric Queries

Be aware of:

- Sheet calculation order
- How Row and Column ID headings are created
- Nested Expansions

BPC for Excel queries are designed to optimize the refresh from the Column/Row grid; meaning it expects some commonality in the columns (i.e. Time dimension members

across all columns and most, if not all, columns point to the same CV). When reports vary from this format the performance degrades. An example of a report that doesn't follow a "clean" grid format is one where each column has a different set of dimensions mapped to it. In other words, it is not just one dim across the columns but many dimensions referenced with consistency.

Example

Assuming a basic 4 dimension application (to really see the performance degradation you need an 8 to 12 dimension application):

If you have time in the columns and accounts in the row, all other dimensions are in the page key (a single reference per query). This is ideal.

Now if you have category and time in the columns and accounts in the row... you're still good, because entity is in the page key.

If you have a different entity, category, and time period in each column and accounts in the row, performance is terrible.

So the bottom line is the nested expansion cannot assist here. It does not depend on expanding up to 3 dimensions in a column or row. It is related to varied dimension members for each cell. There is no way to optimize the query for the data retrieval.

This applies to the older expansion functions EVEXP, EVENE, etc., and, as stated below, it is not quite true with EVDRE. Yet, certainly any function will have a harder time doing its things as the number of dimensions to nest increases.

End of the example.

EVDRE

EvDRE can have a huge impact on performance in BPC.

Example

A triple-nested expansion could take more than 60 minutes to expand and zero suppress to a result set of ~2000 rows. The same triple-nested expansion using EvDRE takes less than 30 seconds to fully expand and refresh.

End of the example.

EVDRE can also provide benefits from its built-in query engine. For example EVDRE goes directly to SQL when retrieving values that exist in the FACT tables. This may have a dramatic impact on scalability, because OLAP does not scale as well, especially during periods of heavy data input activity.

A customer system was tested using EVDRE. This was a complex schedule which took 3 minutes to refresh when you had the system to yourself (a high-end multiserver environment). Running sixty simultaneous users, the query time increased to only 3 minutes 45 seconds. This is very good scalability.

Logic type (MDX, SQL, or SQL Server Stored Procedures)

Generally, most applications strictly use two versions of “logic” to solve calculations in BPC, either MDX or SQL logic (a proprietary syntax of BPC). The big difference between MDX and SQL is that MDX logic, generally speaking, is not scalable. It means that when the number of concurrent users increases the performance degradation is huge. For this basic reason the SQL logic is preferred to MDX.

The approach to using MDX is:

1. Dimension logic is useful in the account dimension for the ratio (KPI) account only. All other formulas should be better defining on SQL.
2. Dimension logic which spans time (such as dynamic open balances, etc) should be used with great care. These are often better performed in SQL logic.
3. Due to the big impact on the performance, do not define a large quantity of hierarchies in the dimension.
4. The source/destination regions defined in the logic must be small as much as possible.
5. Reduce as much as possible the number of COMMITS
6. Be very cautious about using MDX-based logic calculations dependent upon large dimensions. At base-level members, these MDX functions will perform well and often are desirable; however at parent levels in large hierarchies, these calculations often result in massive Analysis Server CPU utilization.

The approach to using SQL is:

1. Select the right trigger in order to work only with the existing records and not with all possible cases. For example; in the budget application you have to calculate all revenues by region and product. In this case you have $REVENUES = UNITS * PRICE$. The right trigger is units because normally price is define for all products vice versa units are defining per regions/product, this because not all products are sold in all regions. In this case the system will apply the formula only for the existing cases, vice versa if you define PRICE as trigger the system will execute the calculation also in case the product is not filled, of course the result is the same but in the second case (Price as trigger) the system has to execute more calculations.
2. Load in memory only the information that you need. For example if you have to calculate REVENUES you define as a region only PRICE and UNITS, you do not need to load in memory the account REVENUES, the system will be faster because will scan less records.
3. Use properties when possible to write more efficient & compact logic. For example if you have to apply the same calculation on a specific set of accounts instead of specifying all single accounts, you can define a property and in the

SQL logic you can test the property (using a SELECT statement) in order to define the account set. This approach is much faster than specifying all accounts and then using a WHEN criteria to perform the filtering.

4. Reduce as much as possible the number of COMMITs – the GO command can be a useful alternative but care should be given here as well.

Important: even if at a first glance it may seem that the GO instruction can be used in place of a COMMIT instruction, this is not quite true. All instructions that are COMMIT-specific (like for example XDIM_MEMBERSET) are still COMMIT-specific and not GO-specific. In other words you cannot redefine the data region to process for each GO instruction, but only for each COMMIT instruction. The GO instruction only sets a stop-and-go point between WHEN / ENDWHEN structures of the same COMMIT section, i.e. of the same data region.

Any GO instruction defines the end of a logic section, more or less like a COMMIT instruction, in the sense that the logic is executed normally up to that point.

Do not leave too many calculations in default logic. Typically currency conversion or Intercompany Eliminations can be left to a later phase.

Finally, as a third alternative to standard BPC-based SQL logic is pure SQL Server stored procedures. BPC SQL Logic allows for the calling of SQL stored procedures directly from the BPC logic, thus it makes database-only processing a real possibility. This last alternative might be useful under the following circumstances:

1. There is a large amount of data to be allocated.
2. Standard SQL Logic requires heavy use of temporary “memory variable” for its processing.

Of course, making the decision to use SQL Logic versus SQL stored procedures is a careful balancing act. SQL stored procedures do not allow for simple administration like the BPC SQL Logic does, so this can often require BPC Administrators to rely on IT-oriented database resources. However, from a scalability and performance standpoint, the SQL stored procedures method will often times be preferable.

Application Maintenance

Use the following guidelines to maintain applications for efficiency.

Features

Compression

Data compression (through eAdmin, not other mechanisms) can have a very significant impact on cube and partition processing times. It is nearly linear: If compression can reduce the number of records in the Fact tables by 50%, you can expect a nearly 50% reduction in the time to process the cube. This will, of course, vary with alternate hierarchies, etc.

Here are three customer case studies:

Test	Percent reduction in records	Percent reduction in process time
1	75% (from 40 to 10 million)	70% (from 2.5 hrs to 45 min)
2	50% (from 32 to 16 million)	67% (from 2 hrs to 40 minutes)
3	75% (From 4 to 1 million)	(from 30 min to min)

Lite optimization

As data grows in the Writeback table, it has a noticeable impact on report performance. By regularly performing Lite Optimizes, report performance can be kept reasonable.

Storing historical data

A very large amount of data can be a problem for processing times, especially if the maintenance window is small.

Many times you will want to store more information than needed on a regular basis. This could be in the form of versions of a Budget or Forecast, number of years worth of data, storing LC and USD for USD entities (or what ever the primary reporting currency is), or before and after allocation results. It is important to reconsider this need so that the application set in use shows what you will need on a regular basis.

1. Clear data that is no longer in use.
 1. After the Budget is finalized, you can typically clear the data from the version categories.
 2. Additionally, after the Budget for the current year is complete, prior year's Budgets can be removed.
 3. Store only Forecast versions that you will be analyzing.
2. If you want to have those versions for reference or audit, move them to a separate application set, which will almost never be used, and clear them from the active one. Remember multiple application sets on the same server will take resources, so evaluate this accordingly.
3. Archive historical data.

1. Many companies only want to analyze a certain number of year's worth of data. If you have data outside that threshold, extract the data or backup the database so that you can always get back to it and then delete it from the database.
2. You can also use Book Publication to display historical reports on an as needed basis without storing the data in the fact table.
3. Create a historical appSet to hold historical information that few people need to see. This is often the best choice, as you can also maintain the historical entity and other structures for historical reference. At a customer, they create a historical application set at the end of each fiscal year, so they have each year's "snapshot" available when needed.

Keep the Wizards directory small

This reduces the initial load time. Although not a hardship in many cases, a large Wizards directory can negatively affect user experience.

Avoid Using Remote Desktop Connections to Application Server

Input and output errors may occur when sending data from a client machine to the BPC Application server when connected using Remote Desktop and Microsoft Terminal Services. These error appear in the system event log of the Application server. To avoid this, do not connect from the client to the Application server using Remote Desktop.

Appendix A: Identifying Performance Bottlenecks

Use this information to identify bottlenecks that decrease performance.

Features

Monitoring Tools

There are two basic performance monitoring tools provided with the Windows server operating system: the Task Manager and the Performance Monitor.

The Task Manager should be tracking on the Processes tab, with the "show processes from all users" box checked. The CPU header should be selected, so that the processes using the most CPU time will be identified at the top of the list. Critical to the success of this test.

The performance monitor provides much more detail than the task manager, but it must be configured to capture the relevant metrics. Performance Monitor can also capture the performance profile for analysis later. This document doesn't document how to use performance monitor, but will identify what to monitor.

Other Monitoring tools include:

- Client and server logs
- Black box – SAP product support
- PSS Diag – Microsoft Professional Services

Memory

Although all of the counters in the Memory performance object are useful, two stand out when measuring Analysis Services overall performance:

Memory \ Pages/sec

This performance counter indicates the number of I/O operations needed to support virtual memory. Ideally, this number should be as low as possible; a high number demonstrates too little available physical memory. Increasing available physical memory should reduce the number of page faults, and therefore reduce the amount of virtual memory used to support active processes such as Analysis Services.

Memory \ Available bytes

This performance counter indicates the amount, in bytes, of available physical memory. Combined with the Pages/sec system counter, this counter can be used to further quantify the amount of available physical memory.

Disk I/O

Disk storage performance is central to Analysis Services performance. The following counters can be compared directly with various performance counters maintained by the Analysis Services performance objects to determine if disk storage represents querying or processing performance impact.

A common cause of poor disk storage performance is the performance of other applications running on the system. Disk storage supports all applications running on a given system; active applications draw resources away from Analysis Services. These performance counters can provide a better picture of absolute querying and processing performance by comparing all of the disk storage activity on the system with the disk storage activity tracked by Analysis Services.

Analysis Services: Agg Cache \ Evictions / Sec

Evictions measure how frequently Analysis Services is flushing its cache and refreshing from the database / disk. This can make performance very poor.

Physical Disk \ Avg. Disk Bytes/Read

This represents the average number of bytes transferred from disk storage during a single read operation.

Physical Disk \ Current Disk Queue Length and Physical Disk: Average Disk Queue Length

This counter represents the current number of queued disk operations. If this number spikes during poor processing performance, especially during the base or aggregating phases, then the current disk storage solution may not be adequate to support the needs of Analysis Services. Ideally, the value of this performance counter should be as low as possible at any given time.

Processor

Scaling up to multiprocessor servers allows for much greater Analysis server performance. Scaling up, however, does not necessarily provide a linear increase in performance. Other factors, such as physical memory or disk storage, can affect the increase provided by scaling up an Analysis server.

Processor \ % Processor Time

If you are consistently above 80-90% find out what process is taking all of the CPU time. The Task Manager will provide the information regarding which process is actually using the CPU. Please contact SAP Services and Technical Consulting if the total CPU is regularly running over 90%, or is steadily using up 100% of one or more of the CPUs in the system.

Analysis Services — Analysis Server: Connections \ Current Connections in Progress

This counter indicates all the connections waiting on OLAP at any given point.

Analysis Server: Locks \ Current Locks

This counter is particularly useful. It measures the current number of locked objects, and can indicate when locks are preventing users from accessing data in a timely fashion.

Analysis Server: Locks \ Current Lock Waits

This counter measures the number of clients waiting for a lock.

SQL Server: Buffer Manager \ Buffer Cache Hit Ratio

The buffer cache hit ratio measures the percentage of pages that were in memory and did not require a disk access to get at the data. A buffer cache hit ratio should be nearly 99 percent. If the ratio is lower, there may be memory constraints that affect performance.

Appendix B: FACT, FAC2, and WB Tables

Use this information to maintain system performance in relation to system storage.

Features

At a summary level it could be said that there are 3 tiers of storage:

- WRITEBACK table
- FAC2 table
- FACT table

This scheme was adopted as prior experience in developing and working with other Corporate Performance Management (BPC) products showed that large applications experience degrading performance as data volumes and concurrency of users increase. The intention, successfully achieved, was to allow a scalable solution no matter how far a client needed to take the application.

In contrast other vendors adopted a “black box” approach which, when a certain scale is reached, necessitates moving to different software from the same vendor when but with a changed architecture, data model or size of database albeit with the same functionalities. Having to rescale as the vendor does is not uncommon. As an example, Hyperion FM reached the limit of 100.000 records in their custom dimension sub-cubes in several large applications. This limit has been removed in version 4.1 but with an associated impact on performance.

The distinction comes down to whether or not a product has been designed to scale or has inherent design limitations necessitating redevelopment of the software.

Each application will have three tables are associated with it. For the application ‘Finance’, the three tables are named ‘tblFACTFinance’, ‘tblFAC2Finance’ and ‘tblFACTWBFinance’.

WB – real time data input (ROLAP partition)

This is data that is the most current data sent to the system. Data sent by BPC for Excel data sends and Investigator browser data sends is placed in real-time storage.

FAC2 – short term and Data Manager imports (MOLAP partition)

This is data that is not real-time data, but is also not in long-term storage yet. When you load data via Data Manager (automatic data load from external data sources), it loads the data to short-term storage so that the loaded data does not affect system performance. Only the cube partition associated with this table is processed, so the system is not taken offline.

Fact – long term history (MOLAP partition)

This is the main data storage. All data eventually resides in long-term storage. Data that is not accessed very often remains in long-term storage so that the system maintains performance.

This structure allows SAP BPC to maintain the same performance over time even when there is a large increase in data volumes.

Periodically clearing real-time data greatly optimizes the performance of the system and an “Optimization” process is required (this could be scheduled automatically based on given parameters like a numbers of records threshold).

Lite Optimization

- Clears Real-time data storage (WRITEBACK) and moves it to short-term data storage (FAC2). This option doesn't take the system offline, and can be scheduled during normal business activity.

Incremental Optimization

- Clears both real-time and Short-term data storage (WB and FAC2) and moves both to Long-term data storage (FACT).
- This option should be run when the system is offline, but it will not take the system offline so it should be run during off-peak periods of activity.

Full Process Optimization

- Clears both real-time and short-term data storage and processes the dimensions.
- This option takes the system offline and takes longer to run than the incremental optimization.
- It is best run scheduled at down-time periods – for example after a month-end close.

The *Compress Database* option is available to rationalize the Fact Tables. This sums multiple entries for the same CurrentView into one entry so that data storage space is minimized. Compressed databases also process more quickly.