

<b>Solution Management Phase</b>	<b>SAP Solution</b>	<b>Topic Area</b>	<b>Solution Manager Area</b>
<i>Operations &amp; Optimization</i>	<i>SAP Business Intelligence (BI)</i>	<i>End-user support concept</i>	<i>Configuration Management</i>



THE BEST-RUN E-BUSINESSES RUN SAP 



## Tuning WebIntelligence Reports

Best Practice for Solution Management

Version Date: 1.2 February 2010

Applicability, Goals, and Requirements.....	3
Goal of Using this Service .....	3
Staff and Skills Requirements .....	3
System Requirements .....	3
WebIntelligence and SAP BW – Architecture .....	4
Performance measurement.....	6
Check which user is connecting to SAP BW .....	6
Getting the end to end statistics .....	7
Gathering the database runtime .....	10
Gathering the OLAP runtime .....	10
Gathering MDX runtime.....	11
Gathering the Single Statistical Records STAD .....	11
Bringing together the results.....	14
Turning on logging in BOE (for OLAP access) .....	14
Tuning WebIntelligence reports .....	16
Tuning database runtime.....	16
Tuning OLAP runtime.....	16
Tuning MDX runtime .....	18
General tuning hints .....	18
Performance tracing and note search .....	19
Tuning WebIntelligence runtime .....	21
General Considerations.....	21
Customizing BI Universe definition .....	22

- Removing unnecessary L00 objects .....22
- Removing unused or redundant detail objects .....22
- Optimizing detail object syntax .....22
- Adding keys to objects used in an LOV for filtering .....23
- Scheduling vs. on-demand reporting .....23
- Query Drill for hierarchies.....24
  - Hierarchies with linked nodes .....24
- Filtering.....24
  - Static filtering with Webl Query filters.....24
  - Static filtering with BEx Query restrictions .....25
- Dynamic filtering of characteristic values .....25
  - Dynamic filtering with BEx query variables.....25
  - Dynamic filtering with Webl filters .....26
- Large LOVs for prompting .....26
- Reports with high data volume.....26
  - Reducing the size by optimizing Webl queries .....26
    - Remove unused fields from the query.....26
    - Refactor queries to extract more constant master data .....27
  - Reducing the number of rows per request by using guided navigation .....30
    - Using Drill .....30
    - Using Report Linking .....30

---

# Applicability, Goals, and Requirements

## **Goal of Using this Service**

*In the Business Intelligence area, a significant change in SAP's strategy for reporting solutions has happened since the acquisition of Business Objects. Until then, SAP promoted the use of the Business Explorer Suite including, among others, the Business Explorer Query Designer, Analyzer, Web Analyzer, Web Application Designer, and Report Designer.*

*Since Business Objects is part of SAP, the usage of the reporting toolset formerly known as Business Objects Information Discovery and Delivery (IDD) is getting more and more widespread within the SAP user community. With products both being prominent and setting de-facto standards such as Crystal Reports, WebIntelligence, Voyager and XCelsius, the user base is moving towards more ease of use, intuitiveness, and flexibility in Business Intelligence reporting.*

*All the SAP Business Objects reporting tools share the same technological basis - the Business Objects Enterprise Platform. This document describes Best Practices for solving common known problems related to this platform.*

## **Staff and Skills Requirements**

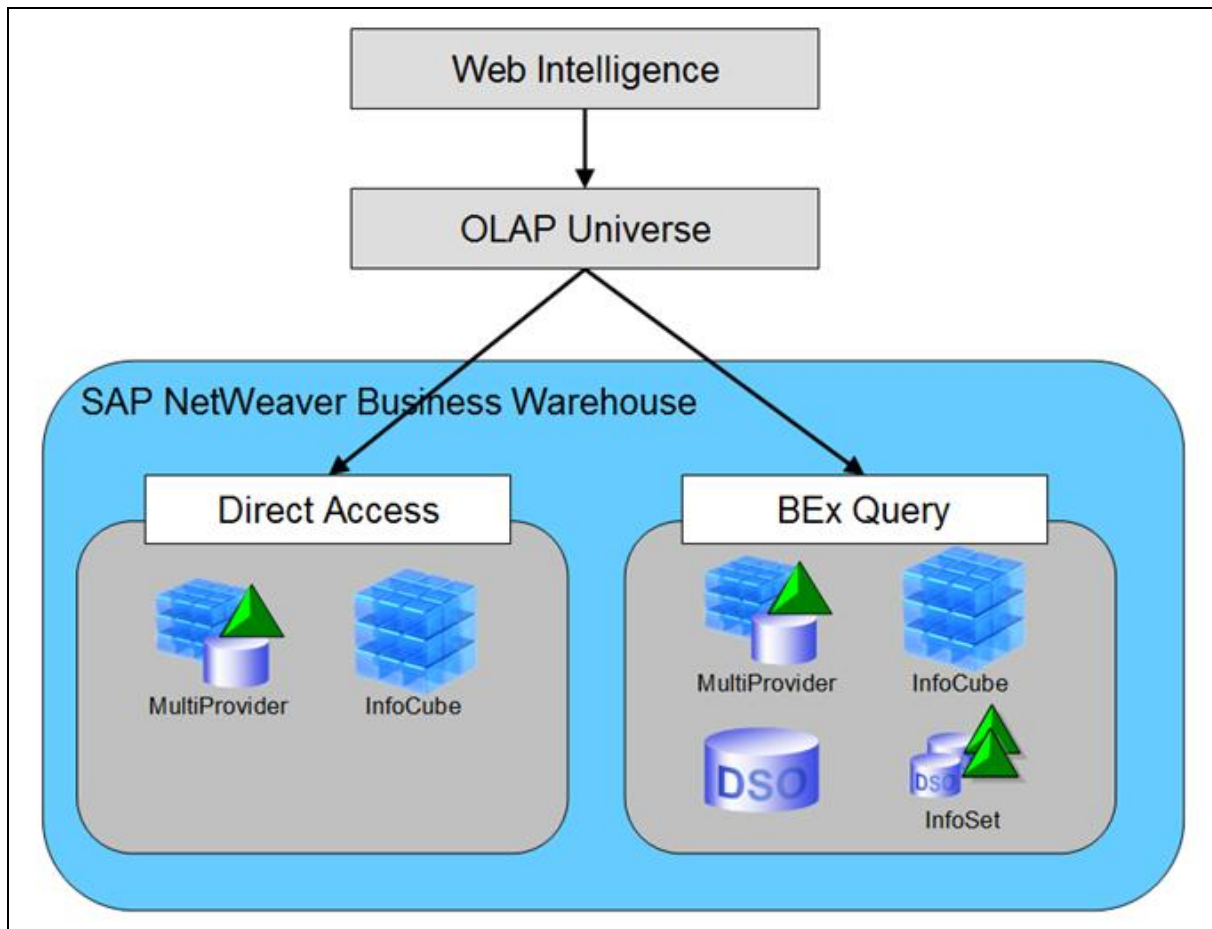
*To implement this Best Practice, you require a good general technical understanding of the SAP Business Objects Enterprise Platform and SAP BW. For example, the role and tasks of the different servers like the WebI Report Servers in a reporting scenario should be known.*

## **System Requirements**

- *Unless stated otherwise, the information given in this document is applicable to any Business Objects Enterprise XI 3.1 installation connecting to an SAP BW 7.01 (EHP1) system. In some cases we also include troubleshooting information relevant for the previous release XI R2.*
-

## WebIntelligence and SAP BW – Architecture

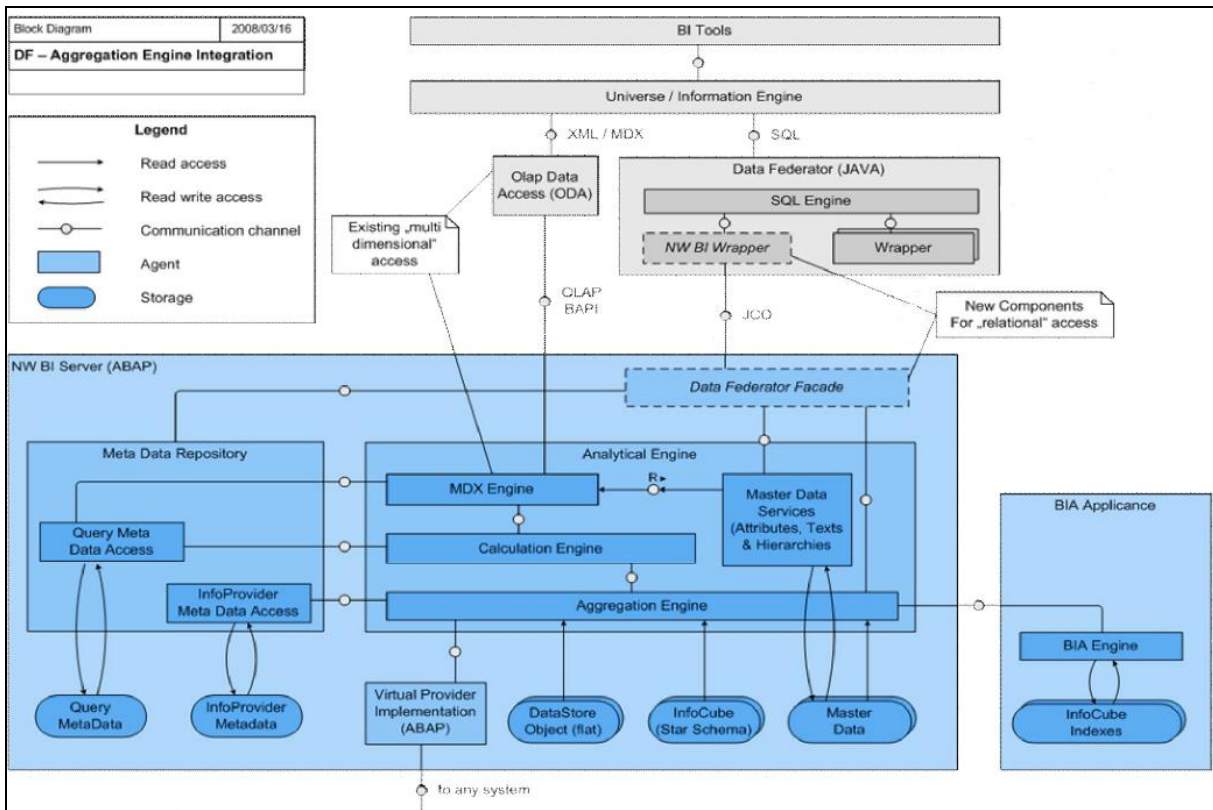
This section describes general ways how to connect WebIntelligence and SAP BW systems.



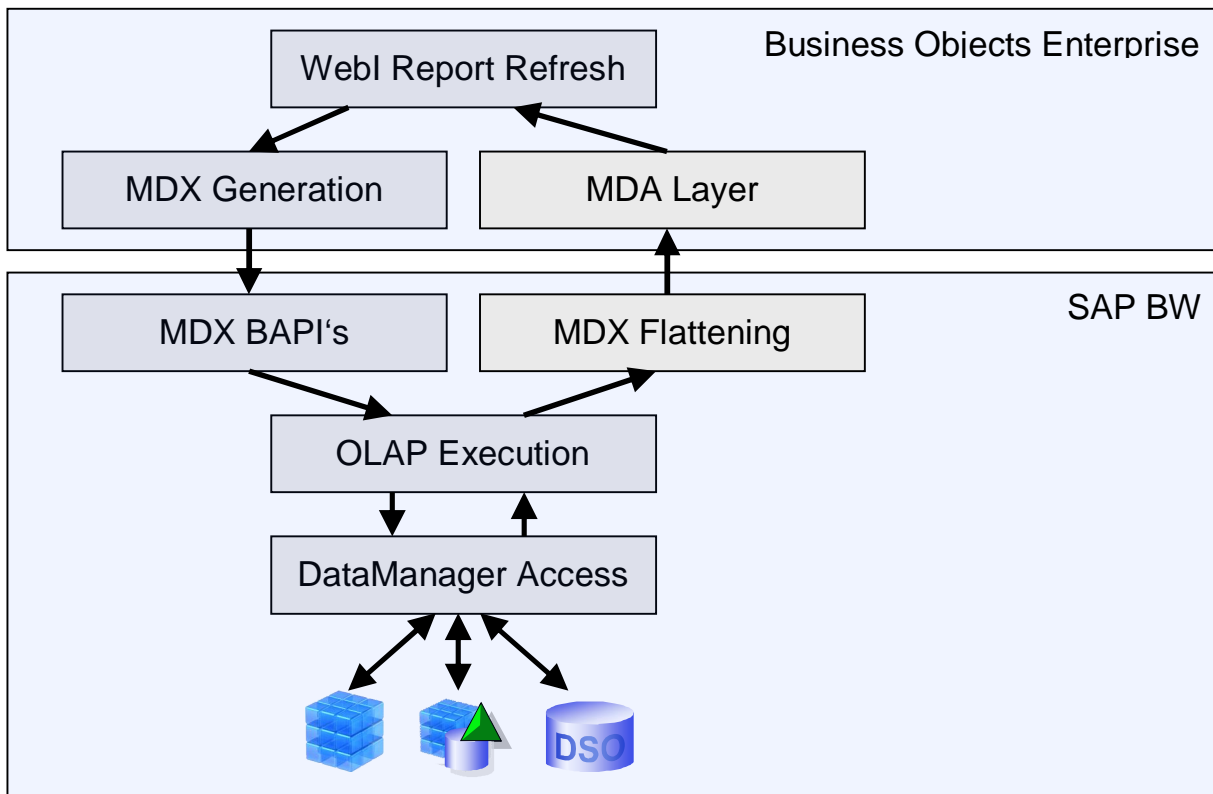
WebIntelligence connects to SAP BW via an OLAP Universe. In most use cases, the Universe will connect via a BEx Query to either a MultiProvider or an InfoCube. In rare cases there can be queries on DataStoreObjects (DSO) or InfoSets as well.

The other possible way of connecting a universe to SAP BW is by using direct access to MultiProviders or InfoCubes. In this case there are no OLAP functionalities like calculated key figures, hierarchies, authorization checks, and so on. Furthermore, a connection can be established with DataFederator. DataFederator connects directly to the database tables of SAP BW. Calculated key figures and other OLAP features are not available from SAP BW in this case.

The following illustration shows the detailed connection mechanisms including DataFederator for the WebIntelligence SAP BW integration.



The detailed execution for a MDX query is illustrated in the following flow chart:



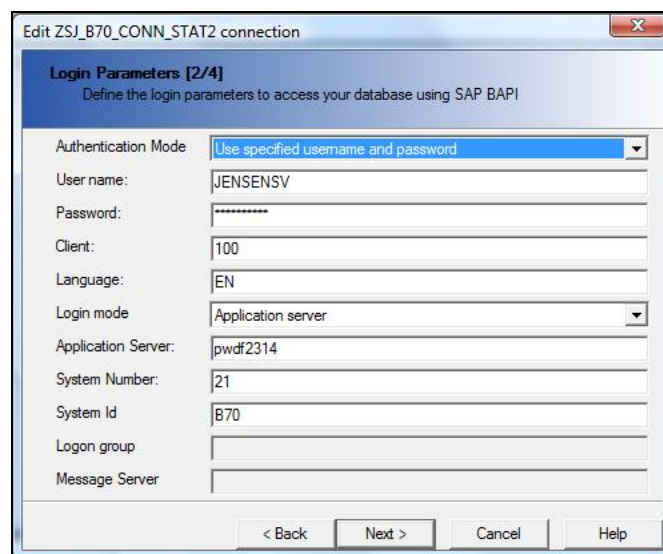
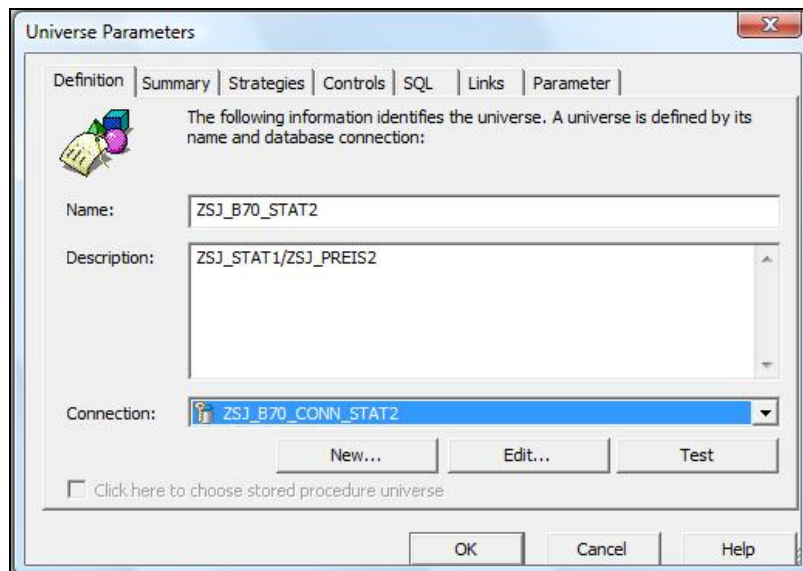
## Performance measurement

This section describes how to find out how much time is spent in which of the above mentioned interfaces / modules. It is very important for performance tuning to know where the significant runtime is spent. Before starting any tuning efforts, it is essential to separate the total runtime into the different processing parts.

## Check which user is connecting to SAP BW

First of all, we have to find out which user is connecting to SAP BW. Each WebI report has at least one underlying query which is on top of a Universe. There can be more than one query in one WebI report and therefore even more than one Universe be involved. In this case it should be checked if all of the involved Universes are using the same SAP User or Single-Sign-On (SSO).

The configured user in the Universe can be checked, for example, in the Universe Designer. Import the specific Universe and go to *File* → *Parameters*. In the Universe Parameters, there is a connection. Pressing "Edit" on the connection will reveal the configured user.



Take down the user name, since we need it subsequently to do traces, gather statistics, and so on in the SAP BW backend. If the Authentication Mode is set to “Use Single Sign On when refreshing reports at view time”, the user connecting to SAP BW will be the same as the user you log onto InfoView with SAP authentication.

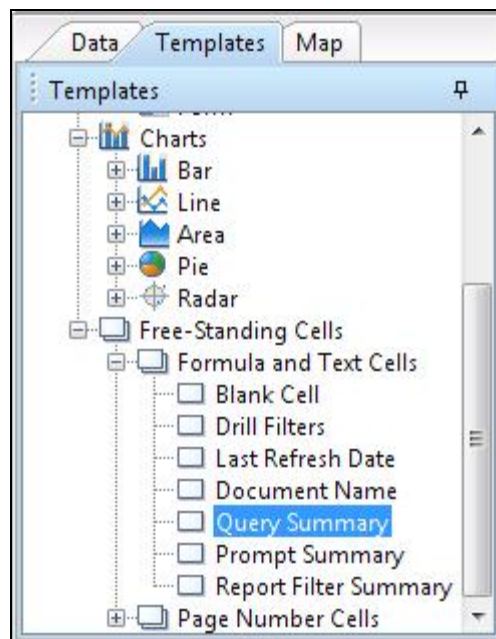
## Getting the End to End Statistics

Make sure the statistics data collection for the specific query/queries used by the Universe(s) is turned on in SAP BW. Use transaction RSDSTAT and make sure the following settings are applied for the BEx query(s) (X On – 2 All):

Maintenance of Statistics Properties						
Information <span>Delete Statistical Data</span>						
Query <span>InfoProvider</span> <span>Web Template</span> <span>Workbook</span>						
<span>Replace Values</span>				Settings: <span>X On</span> <span>2 All</span>		
Query Name	InfoProvider	Author	Last Changed	Stati...	OLA...	Cha...
ZSJ_PREIS2	ZSJ_STAT1	JENSENSV	10.09.2009 02:46	X	2	<input checked="" type="checkbox"/>

We are now ready to run the WebIntelligence report (either in InfoView or in RichClient). Press the “Refresh Data” button in the WebIntelligence report or press the button “Run Query” if you were editing the query. After the refresh has completed, we need to get the complete runtime of the report and the total records returned. If there are multiple queries involved in the WebIntelligence report, please add them up.

To see the mentioned runtime and reports, the “Query Summary” template from within WebIntelligence can be used. Drag the “Query Summary” into your report panel:



It will look somehow like the following illustration:

```

*** Query Name:Query 1 ***

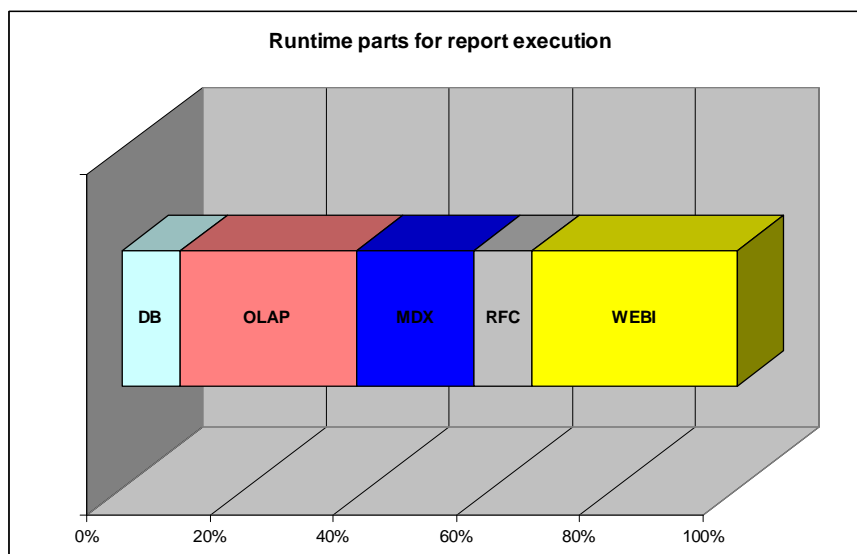
** Query Properties:
  Universe:ZSJ_B70_STAT3
  Last Refresh Date:20.11.09 11:47
  Last Execution Duration: 71
  Number of rows: 23.940
  Retrieve Duplicate Row: ON

** Query Definition:
  Result Objects: L01 Product, L02 Product, L03 Product,
L04 Product, L01 Calendar Year/Month, Index

```

The total runtime of the report execution in this case was 71 seconds. The number of records retrieved was 23.940<sup>1</sup>.

The overall runtime can be split into 5 different parts (Database, OLAP, MDX, RFC and WebIntelligence). Each of these parts will be explained and described in the following sections. The following illustration visualizes the different components of the total report execution time:




Now you have to gather the statistics in the SAP BW system. Use transaction SE16 (Data Browser) and go to table RSDDSTAT\_OLAP. In the selection screen enter the user identified further up and the respective date and time. Keep in mind the time is in UTC.

Keep in mind the maximum number of hits in the selection screen is 200 by default. The statistics records can easily exceed that value when running WebIntelligence reports so please first evaluate the number of records (with the respective button) and change the maximum number of hits accordingly. Otherwise, important runtime steps could be cut off.

<sup>1</sup> Throughout this document all numbers are displayed in German format (dot as thousands separator and comma as decimal separator) in order to match the format in the screenshots.



SESSIONUID		to		▶
STEPUID		to		▶
HANDLEID		to		▶
HANDLETP		to		▶
EVENTID		to		▶
UNAME	JENSENSV	to		▶
STEPPT		to		▶
STEPCNT		to		▶
UTIME	11:47:00	to	13:00:00	▶
CALDAY	20.11.2009	to	20.11.2009	▶
RUNTIME		to		▶
INFOPROV		to		▶
OBJNAME		to		▶
OBJPROP		to		▶
STATLEVEL		to		▶
EVTIME		to		▶
EVCOUNT		to		▶
EVENTIDCNT		to		▶
STARTTIME		to		▶
Width of Output List	250			
Maximum No. of Hits	200			

Press the execute button  and you will get the table with all the statistics in SAP BW. Now, you have to separate the backend runtimes into database, OLAP and MDX runtime.


The statistics result table will look like this:

SESSIONUID	STEPUID	HANDLEID	HANDLETP	EVENTID	UNAME	STEPPT	STPCNT	UTIME	CALDAY	RUNTIME	INFOPROV	OBJNAME	OBJPROP	STATLEVEL
D6XBMS30528YGR1T19VLRRMG	D6XBMS30528Y9G2M4HJ84DC0		DFLT		JENSENSV	MDX		1 11:47:45	20.11.2009	0.100000				2
D6XBMS30528YGR1T19VLRRMG	D6XBMS30528Y9G2M4HJ84DC0	9999	MDX	40000	JENSENSV	MDX		1 11:47:45	20.11.2009	0.100000				2
D6XBMS30528YGR1T19VLRRMG	D6XBMS30528Y9H4APCL453QRK99		DFLT		JENSENSV	MDX		2 11:47:45	20.11.2009	0.000000				2
D6XBMS30528YGR1T19VLRRMG	D6XBMS30528Y9H4APCL453QRK99	9999	MDX	40000	JENSENSV	MDX		2 11:47:45	20.11.2009	0.000000				2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3		DFLT		JENSENSV	MDX		3 11:47:45	20.11.2009	0.718000				2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3		DFLT	1	JENSENSV	MDX		3 11:47:45	20.11.2009	0.718000				2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3	3	OLAP	3000	JENSENSV	MDX		3 11:47:45	20.11.2009	0.718000	ZSL_STAT1	ZSL_PRE03	H0_00	2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3	3	OLAP	3010	JENSENSV	MDX		3 11:47:45	20.11.2009	0.718000	ZSL_STAT1	ZSL_PRE03	H0_00	2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3	3	OLAP	3000	JENSENSV	MDX		3 11:47:45	20.11.2009	0.718000	ZSL_STAT1	ZSL_PRE03	H0_00	2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3	3	OLAP	3010	JENSENSV	MDX		3 11:47:45	20.11.2009	0.718000	ZSL_STAT1	ZSL_PRE03	H0_00	2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3	3	OLAP	3999	JENSENSV	MDX		3 11:47:45	20.11.2009	0.718000	ZSL_STAT1	ZSL_PRE03	H0_00	2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3	3	OLAP	4400	JENSENSV	MDX		3 11:47:45	20.11.2009	0.718000	ZSL_STAT1	ZSL_PRE03	H0_00	2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3	3	OLAP	4900	JENSENSV	MDX		3 11:47:45	20.11.2009	0.718000	ZSL_STAT1	ZSL_PRE03	H0_00	2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3	3	OLAP	18000	JENSENSV	MDX		3 11:47:45	20.11.2009	0.718000	ZSL_STAT1	ZSL_PRE03	H0_00	2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3	3	OLAP	40020	JENSENSV	MDX		3 11:47:45	20.11.2009	0.718000	ZSL_STAT1	ZSL_PRE03	H0_00	2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3	9999	MDX	40000	JENSENSV	MDX		3 11:47:45	20.11.2009	0.718000	ZSL_STAT1	ZSL_PRE03	H0_00	2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3	9999	MDX	40020	JENSENSV	MDX		3 11:47:45	20.11.2009	0.718000	ZSL_STAT1	ZSL_PRE03	H0_00	2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3		DFLT		JENSENSV	MDX		4 11:47:45	20.11.2009	0.000000				2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3		DFLT	1	JENSENSV	MDX		4 11:47:45	20.11.2009	0.000000				2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3	3	OLAP	3999	JENSENSV	MDX		4 11:47:45	20.11.2009	0.000000	ZSL_STAT1	ZSL_PRE03	H0_00	2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3	9999	MDX	40000	JENSENSV	MDX		4 11:47:45	20.11.2009	0.000000	ZSL_STAT1	ZSL_PRE03	H0_00	2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3	9999	MDX	40020	JENSENSV	MDX		4 11:47:45	20.11.2009	0.000000	ZSL_STAT1	ZSL_PRE03	H0_00	2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3		DFLT		JENSENSV	MDX		5 11:47:45	20.11.2009	0.015000				2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3		DFLT	1	JENSENSV	MDX		5 11:47:45	20.11.2009	0.015000				2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3	3	OLAP	3999	JENSENSV	MDX		5 11:47:45	20.11.2009	0.015000	ZSL_STAT1	ZSL_PRE03	H0_00	2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3	9999	MDX	40000	JENSENSV	MDX		5 11:47:45	20.11.2009	0.015000	ZSL_STAT1	ZSL_PRE03	H0_00	2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3	9999	MDX	40020	JENSENSV	MDX		5 11:47:45	20.11.2009	0.015000	ZSL_STAT1	ZSL_PRE03	H0_00	2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3		DFLT		JENSENSV	MDX		8 11:47:45	20.11.2009	0.018000				2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3		DFLT	1	JENSENSV	MDX		8 11:47:45	20.11.2009	0.018000				2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3	3	OLAP	3999	JENSENSV	MDX		8 11:47:45	20.11.2009	0.018000	ZSL_STAT1	ZSL_PRE03	H0_00	2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3	9999	MDX	40000	JENSENSV	MDX		8 11:47:45	20.11.2009	0.018000	ZSL_STAT1	ZSL_PRE03	H0_00	2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3	9999	MDX	40020	JENSENSV	MDX		8 11:47:45	20.11.2009	0.018000	ZSL_STAT1	ZSL_PRE03	H0_00	2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3		DFLT		JENSENSV	MDX		7 11:47:45	20.11.2009	0.000000				2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3		DFLT	1	JENSENSV	MDX		7 11:47:45	20.11.2009	0.000000				2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3	9999	MDX	40000	JENSENSV	MDX		7 11:47:45	20.11.2009	0.000000				2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3		DFLT		JENSENSV	MDX		8 11:47:45	20.11.2009	0.000000				2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3		DFLT	1	JENSENSV	MDX		8 11:47:45	20.11.2009	0.000000				2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3	3	OLAP	3999	JENSENSV	MDX		8 11:47:45	20.11.2009	0.000000	ZSL_STAT1	ZSL_PRE03	H0_00	2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3	9999	MDX	40000	JENSENSV	MDX		8 11:47:45	20.11.2009	0.000000	ZSL_STAT1	ZSL_PRE03	H0_00	2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3	9999	MDX	40020	JENSENSV	MDX		8 11:47:45	20.11.2009	0.000000	ZSL_STAT1	ZSL_PRE03	H0_00	2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3		DFLT		JENSENSV	MDX		9 11:47:45	20.11.2009	0.000000				2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3		DFLT	1	JENSENSV	MDX		9 11:47:45	20.11.2009	0.000000				2
D6XBMS30528YGR1T19VLRRMG	D6XBMS305289R133ANAP79YF3		DFLT	1	JENSENSV	MDX		9 11:47:45	20.11.2009	0.000000				2

For further information on SAP BW statistics and detailed descriptions of the specific EVENTID's please see the SAP Library at:

[http://help.sap.com/saphelp\\_nw70/helpdata/en/43/e39fd25ff502d2e1000000a1553f7/content.htm](http://help.sap.com/saphelp_nw70/helpdata/en/43/e39fd25ff502d2e1000000a1553f7/content.htm)

# Gathering the Database Runtime


Click on the header of the column "EVENTID" to mark the whole column. Set a filter by pressing  and select the between range from 9000 to 9011.

EVEN...	UNAME	STEPTP	STPEC...	UTIME	CALDAY	RUNTIME	INFOPROV	OBJNAME	OBJPROP	STATLEVEL	EVTIME	EVCOUNT
9000	JENSENSV	MDX	19	11:47:47	20.11.2009	28.03100	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	2,030000	0
9010	JENSENSV	MDX	19	11:47:47	20.11.2009	28.03100	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0,000000	36.732
9011	JENSENSV	MDX	19	11:47:47	20.11.2009	28.03100	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0,000000	37.537

In the column "EVTIME" we have the runtime on the database. In case there is more than one access to the database, you can sum up the EVTIME to get the total database runtime. In this example, we have a database time of 2,03 seconds.

If database runtime is high, it is also interesting to check EVENTID 9011 (Database selects) and EVENTID 9010 (Database transfers). If event 9011 is far higher than 9010, this means that much more records had to be read from the database than necessary. This is sometimes a hint for suboptimal database access and could, in some cases, be avoided by building aggregates for the specific cube (INFOPROV).

# Gathering the OLAP Runtime

Click on the header of the column "EVENTID" to mark the whole column. Set a filter by pressing  and select the between range from 2500 to 4999.

HANDLEID	HANDLETP	EVEN	UNAME	STEPTP	STPEC	UTIME	CALDAY	RUNTIME	INFOPROV	OBJNAME	OBJPROP	STATLEVEL	EVTIME	EVCOUNT	EVENTIDC	STARTTIME
3	OLAP	3000	JENSENSV	MDX	3	11:47:45	20.11.2009	0.718000	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0.016000	0	12	20.091.120.114.746.6290000
3	OLAP	3010	JENSENSV	MDX	3	11:47:45	20.11.2009	0.718000	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0.031000	0	1	20.091.120.114.746.6290000
3	OLAP	3500	JENSENSV	MDX	3	11:47:45	20.11.2009	0.718000	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0.327000	0	11	20.091.120.114.746.6290000
3	OLAP	3510	JENSENSV	MDX	3	11:47:45	20.11.2009	0.718000	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0.015000	0	3	20.091.120.114.746.6290000
3	OLAP	3999	JENSENSV	MDX	3	11:47:45	20.11.2009	0.718000	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0.094000	0	12	20.091.120.114.746.6290000
3	OLAP	4400	JENSENSV	MDX	3	11:47:45	20.11.2009	0.718000	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0.000000	0	1	20.091.120.114.746.6290000
3	OLAP	4800	JENSENSV	MDX	3	11:47:45	20.11.2009	0.718000	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0.032000	0	1	20.091.120.114.746.6290000
3	OLAP	3999	JENSENSV	MDX	4	11:47:46	20.11.2009	0.000000	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0.000000	0	2	20.091.120.114.746.3940000
3	OLAP	3999	JENSENSV	MDX	5	11:47:46	20.11.2009	0.015000	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0.000000	0	2	20.091.120.114.746.4570000
3	OLAP	3999	JENSENSV	MDX	6	11:47:46	20.11.2009	0.016000	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0.000000	0	2	20.091.120.114.746.5190000
3	OLAP	3999	JENSENSV	MDX	8	11:47:46	20.11.2009	0.000000	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0.000000	0	2	20.091.120.114.746.6130000
3	OLAP	3999	JENSENSV	MDX	9	11:47:46	20.11.2009	0.000000	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0.000000	0	2	20.091.120.114.746.6440000
3	OLAP	3999	JENSENSV	MDX	10	11:47:46	20.11.2009	0.016000	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0.000000	0	2	20.091.120.114.746.6600000
3	OLAP	3999	JENSENSV	MDX	11	11:47:46	20.11.2009	0.000000	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0.000000	0	2	20.091.120.114.746.7380000
3	OLAP	3999	JENSENSV	MDX	13	11:47:46	20.11.2009	0.000000	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0.000000	0	2	20.091.120.114.746.9100000
3	OLAP	3999	JENSENSV	MDX	14	11:47:46	20.11.2009	0.000000	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0.000000	0	2	20.091.120.114.746.9570000
3	OLAP	3999	JENSENSV	MDX	16	11:47:47	20.11.2009	0.000000	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0.000000	0	2	20.091.120.114.747.1130000
3	OLAP	3999	JENSENSV	MDX	17	11:47:47	20.11.2009	0.000000	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0.000000	0	2	20.091.120.114.747.1600000
1	MDX	4400	JENSENSV	MDX	18	11:47:47	20.11.2009	0.285000	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0.000000	0	1	20.091.120.114.747.2070000
3	OLAP	3999	JENSENSV	MDX	18	11:47:47	20.11.2009	0.285000	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0.000000	0	1	20.091.120.114.747.2070000
3	OLAP	2500	JENSENSV	MDX	19	11:47:47	20.11.2009	28.03100	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0.015000	0	4	20.091.120.114.747.5040000
3	OLAP	2520	JENSENSV	MDX	19	11:47:47	20.11.2009	28.03100	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0.000000	0	1	20.091.120.114.747.5040000
3	OLAP	3000	JENSENSV	MDX	19	11:47:47	20.11.2009	28.03100	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0.031000	0	2	20.091.120.114.747.5040000
3	OLAP	3100	JENSENSV	MDX	19	11:47:47	20.11.2009	28.03100	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	1.858000	48.728	1	20.091.120.114.747.5040000
3	OLAP	3110	JENSENSV	MDX	19	11:47:47	20.11.2009	28.03100	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0.799000	0	1	20.091.120.114.747.5040000
3	OLAP	3200	JENSENSV	MDX	19	11:47:47	20.11.2009	28.03100	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	16.92200	34.200	1	20.091.120.114.747.5040000
3	OLAP	3999	JENSENSV	MDX	19	11:47:47	20.11.2009	28.03100	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0.000000	0	2	20.091.120.114.747.5040000
7	OLAP	3000	JENSENSV	MDX	23	11:48:23	20.11.2009	0.250000	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0.000000	0	12	20.091.120.114.823.3790000
7	OLAP	3010	JENSENSV	MDX	23	11:48:23	20.11.2009	0.250000	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0.119000	0	1	20.091.120.114.823.3790000
7	OLAP	3500	JENSENSV	MDX	23	11:48:23	20.11.2009	0.250000	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0.048000	0	11	20.091.120.114.823.3790000
7	OLAP	3510	JENSENSV	MDX	23	11:48:23	20.11.2009	0.250000	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0.000000	0	3	20.091.120.114.823.3790000
7	OLAP	3900	JENSENSV	MDX	23	11:48:23	20.11.2009	0.250000	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0.000000	0	1	20.091.120.114.823.3790000
7	OLAP	3999	JENSENSV	MDX	23	11:48:23	20.11.2009	0.250000	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0.015000	0	10	20.091.120.114.823.3790000
7	OLAP	3999	JENSENSV	MDX	24	11:48:23	20.11.2009	0.016000	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0.000000	0	2	20.091.120.114.823.6760000
7	OLAP	3999	JENSENSV	MDX	25	11:48:23	20.11.2009	0.015000	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0.000000	0	2	20.091.120.114.823.7700000
7	OLAP	3999	JENSENSV	MDX	26	11:48:24	20.11.2009	0.000000	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0.000000	0	2	20.091.120.114.824.0200000
3	OLAP	3999	JENSENSV	MDX	27	11:48:56	20.11.2009	0.286000	ZSJ_STAT1	ZSJ_PREIS3	H0_00____	2	0.016000	0	1	20.091.120.114.856.1760000

The sum of the EVTIME in this example is 20,33 seconds. Another important measure for the OLAP runtime is the EVCOUNT of the 3200 event(s). This is the number of cells which OLAP had to calculate and can significantly influence the OLAP performance.


EVENTID	Short Text	Long description
3200	OLAP: Data Transfer at Front End	During a data transfer to the front end, exception aggregations and simple currency translations are carried out, formulas are calculated, and the correct number of decimal places for the data cells is

	determined. In addition, the result set is sorted according to the interface settings. The number of cells that have been read is in the counter.
--	---

For further investigation of long OLAP runtimes event 3100 can be taken into account as well.

EVENTID	Short Text	Long description
3100	OLAP: Read Data	This event measures the time that is needed to group together the data requests to the data manager or to read the OLAP cache. The number of characteristic combinations that have been read is in the counter.

## Gathering MDX runtime

Click on the header of the column "EVENTID" to mark the whole column. Set a filter by pressing  and select the between range from 40000 to 40036.

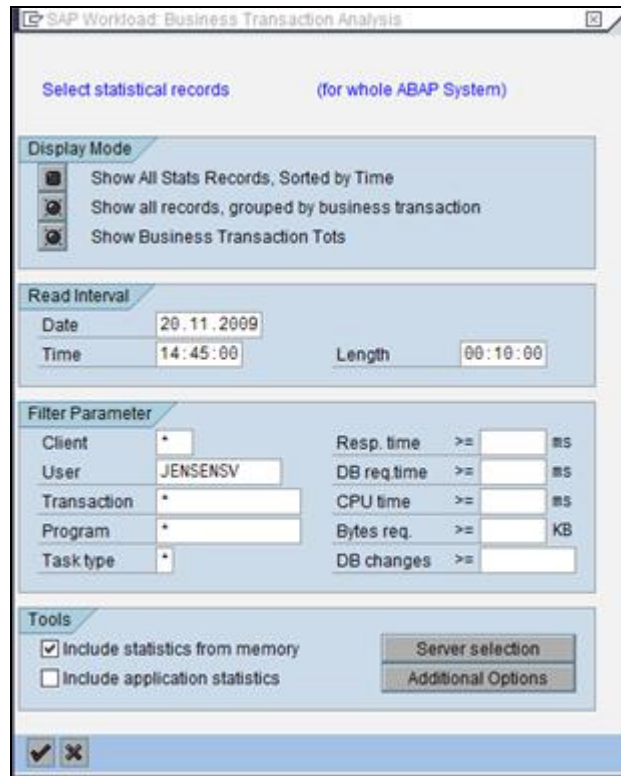
NO	HANDLED	HANDLETH	EVEN	UNAME	STEPTP	STPC	UTIME	CALCV	RUNTIME	INFOPROV	OBJNAME	OBJPROP	STATLEVEL	EVTIME	EVCOUNT	EVENTDOC	STARTTIME	
C	C	9999	MDX	40020	JENSENSV	MDX	8	11:47:45	20.11.2009	0.000000	ZSJ_STAT1	ZSJ_PREIS3	HO_00	2	0.000000	0	1	20.091.120.114.745.8130000
C	C	9999	MDX	40000	JENSENSV	MDX	9	11:47:45	20.11.2009	0.000000	ZSJ_STAT1	ZSJ_PREIS3	HO_00	2	0.000000	0	1	20.091.120.114.745.8440000
C	C	9999	MDX	40020	JENSENSV	MDX	9	11:47:45	20.11.2009	0.000000	ZSJ_STAT1	ZSJ_PREIS3	HO_00	2	0.000000	0	1	20.091.120.114.745.8440000
C	C	9999	MDX	40000	JENSENSV	MDX	10	11:47:45	20.11.2009	0.016000	ZSJ_STAT1	ZSJ_PREIS3	HO_00	2	0.015000	0	1	20.091.120.114.745.8600000
C	C	9999	MDX	40020	JENSENSV	MDX	10	11:47:45	20.11.2009	0.016000	ZSJ_STAT1	ZSJ_PREIS3	HO_00	2	0.000000	0	1	20.091.120.114.745.8600000
C	C	9999	MDX	40000	JENSENSV	MDX	11	11:47:45	20.11.2009	0.000000	ZSJ_STAT1	ZSJ_PREIS3	HO_00	2	0.000000	0	1	20.091.120.114.745.7380000
C	C	9999	MDX	40020	JENSENSV	MDX	11	11:47:45	20.11.2009	0.000000	ZSJ_STAT1	ZSJ_PREIS3	HO_00	2	0.000000	0	1	20.091.120.114.745.7380000
C	C	9999	MDX	40500	JENSENSV	MDX	12	11:47:45	20.11.2009	0.016000	ZSJ_STAT1	ZSJ_PREIS3	HO_00	2	0.015000	0	1	20.091.120.114.745.7690000
C	C	9999	MDX	40000	JENSENSV	MDX	13	11:47:45	20.11.2009	0.000000	ZSJ_STAT1	ZSJ_PREIS3	HO_00	2	0.000000	0	1	20.091.120.114.745.9100000
C	C	9999	MDX	40020	JENSENSV	MDX	13	11:47:45	20.11.2009	0.000000	ZSJ_STAT1	ZSJ_PREIS3	HO_00	2	0.000000	0	1	20.091.120.114.745.9100000
C	C	9999	MDX	40000	JENSENSV	MDX	14	11:47:45	20.11.2009	0.000000	ZSJ_STAT1	ZSJ_PREIS3	HO_00	2	0.000000	0	1	20.091.120.114.745.9570000
C	C	9999	MDX	40020	JENSENSV	MDX	14	11:47:45	20.11.2009	0.000000	ZSJ_STAT1	ZSJ_PREIS3	HO_00	2	0.000000	0	1	20.091.120.114.745.9570000
C	C	9999	MDX	40000	JENSENSV	MDX	15	11:47:47	20.11.2009	0.000000	ZSJ_STAT1	ZSJ_PREIS3	HO_00	2	0.000000	0	1	20.091.120.114.747.0190000
C	C	9999	MDX	40020	JENSENSV	MDX	15	11:47:47	20.11.2009	0.000000	ZSJ_STAT1	ZSJ_PREIS3	HO_00	2	0.000000	0	1	20.091.120.114.747.0190000
C	C	9999	MDX	40000	JENSENSV	MDX	16	11:47:47	20.11.2009	0.000000	ZSJ_STAT1	ZSJ_PREIS3	HO_00	2	0.000000	0	1	20.091.120.114.747.1130000
C	C	9999	MDX	40020	JENSENSV	MDX	16	11:47:47	20.11.2009	0.000000	ZSJ_STAT1	ZSJ_PREIS3	HO_00	2	0.000000	0	1	20.091.120.114.747.1130000
C	C	9999	MDX	40000	JENSENSV	MDX	17	11:47:47	20.11.2009	0.000000	ZSJ_STAT1	ZSJ_PREIS3	HO_00	2	0.000000	0	1	20.091.120.114.747.1600000
C	C	9999	MDX	40020	JENSENSV	MDX	17	11:47:47	20.11.2009	0.000000	ZSJ_STAT1	ZSJ_PREIS3	HO_00	2	0.000000	0	1	20.091.120.114.747.1600000
C	C	1	MDX	40000	JENSENSV	MDX	18	11:47:47	20.11.2009	0.265000	ZSJ_STAT1	ZSJ_PREIS3	HO_00	2	0.203000	0	2	20.091.120.114.747.2070000
C	C	1	MDX	40010	JENSENSV	MDX	18	11:47:47	20.11.2009	0.265000	ZSJ_STAT1	ZSJ_PREIS3	HO_00	2	0.000000	0	1	20.091.120.114.747.2070000
C	C	1	MDX	40910	JENSENSV	MDX	18	11:47:47	20.11.2009	0.265000	ZSJ_STAT1	ZSJ_PREIS3	HO_00	2	0.062000	0	1	20.091.120.114.747.2070000
C	C	1	MDX	40911	JENSENSV	MDX	18	11:47:47	20.11.2009	0.265000	ZSJ_STAT1	ZSJ_PREIS3	HO_00	2	0.000000	0	1	20.091.120.114.747.2070000
C	C	1	MDX	40020	JENSENSV	MDX	18	11:47:47	20.11.2009	0.265000	ZSJ_STAT1	ZSJ_PREIS3	HO_00	2	0.000000	0	1	20.091.120.114.747.2070000
C	C	1	MDX	40000	JENSENSV	MDX	19	11:47:47	20.11.2009	28.03100				2	4.157000	0	1	20.091.120.114.747.5040000
C	C	1	MDX	40030	JENSENSV	MDX	19	11:47:47	20.11.2009	28.03100				2	2.094000	0	1	20.091.120.114.747.5040000
C	C	1	MDX	40000	JENSENSV	MDX	20	11:48:19	20.11.2009	1.422000				2	0.000000	0	1	20.091.120.114.819.4260000
C	C	1	MDX	40912	JENSENSV	MDX	20	11:48:19	20.11.2009	1.422000				2	3.344000	33.972	1	20.091.120.114.819.4260000
C	C	1	MDX	40913	JENSENSV	MDX	20	11:48:19	20.11.2009	1.422000				2	0.234000	0	1	20.091.120.114.819.4260000
C	C	1	MDX	40032	JENSENSV	MDX	20	11:48:19	20.11.2009	1.422000				2	0.844000	0	1	20.091.120.114.819.4260000
C	C	1	MDX	40000	JENSENSV	MDX	21	11:48:21	20.11.2009	0.016000				2	0.000000	0	1	20.091.120.114.821.6130000
C	C	1	MDX	40030	JENSENSV	MDX	21	11:48:21	20.11.2009	0.016000				2	0.000000	0	1	20.091.120.114.821.6130000
C	C	1	MDX	40000	JENSENSV	MDX	22	11:48:21	20.11.2009	0.000000				2	0.000000	0	1	20.091.120.114.821.6600000
C	C	1	MDX	40031	JENSENSV	MDX	22	11:48:21	20.11.2009	0.000000				2	0.000000	0	1	20.091.120.114.821.6600000
C	C	9999	MDX	40000	JENSENSV	MDX	23	11:48:23	20.11.2009	0.250000	ZSJ_STAT1	ZSJ_PREIS3	HO_00	2	0.031000	0	1	20.091.120.114.823.3790000
C	C	9999	MDX	40020	JENSENSV	MDX	23	11:48:23	20.11.2009	0.250000	ZSJ_STAT1	ZSJ_PREIS3	HO_00	2	0.031000	0	1	20.091.120.114.823.3790000
C	C	9999	MDX	40000	JENSENSV	MDX	24	11:48:23	20.11.2009	0.016000	ZSJ_STAT1	ZSJ_PREIS3	HO_00	2	0.016000	0	1	20.091.120.114.823.6760000
C	C	9999	MDX	40020	JENSENSV	MDX	24	11:48:23	20.11.2009	0.016000	ZSJ_STAT1	ZSJ_PREIS3	HO_00	2	0.000000	0	1	20.091.120.114.823.6760000
C	C	9999	MDX	40000	JENSENSV	MDX	25	11:48:23	20.11.2009	0.015000	ZSJ_STAT1	ZSJ_PREIS3	HO_00	2	0.015000	0	1	20.091.120.114.823.7700000
C	C	9999	MDX	40020	JENSENSV	MDX	25	11:48:23	20.11.2009	0.015000	ZSJ_STAT1	ZSJ_PREIS3	HO_00	2	0.000000	0	1	20.091.120.114.823.7700000
C	C	9999	MDX	40000	JENSENSV	MDX	26	11:48:24	20.11.2009	0.000000	ZSJ_STAT1	ZSJ_PREIS3	HO_00	2	0.000000	0	1	20.091.120.114.824.0200000
C	C	9999	MDX	40020	JENSENSV	MDX	26	11:48:24	20.11.2009	0.000000	ZSJ_STAT1	ZSJ_PREIS3	HO_00	2	0.000000	0	1	20.091.120.114.824.0200000
C	C	1	MDX	40000	JENSENSV	MDX	27	11:48:56	20.11.2009	0.266000				2	0.250000	0	1	20.091.120.114.856.1760000
C	C	1	MDX	40002	JENSENSV	MDX	27	11:48:56	20.11.2009	0.266000				2	0.000000	0	1	20.091.120.114.856.1760000
															= 8.64000			

The sum of the EVTIME in this example is 8,64 seconds. This is the overall MDX runtime including the flattening. In SAP BW 7.0 systems prior EHP1 SP3 you won't find events greater than 40032 because the flattening of the data is done on the Business Objects XI server. If there are events between 40033 and 40036 then the flattening is done in the BW backend.

## Gathering the Single Statistical Records STAD

To get the memory consumption, Remote Function Calls (RFC) and the details about all the BAPIs called during report execution, we need to enter transaction STAD in SAP BW.

When starting STAD, set a filter on the connection user for the Universe and select the appropriate start time and the length of the interval to be investigated (according to the total runtime of the report).



This will return a set of RFC calls to the backend. Normally, the major runtime influencing part can be easily determined by the column "Response time (ms)".

Started	Server	Transaction	Program	T Scr. Wp	User	Response time (ms)	Time in WPs (ms)	Wait time (ms)	CPU time (ms)	DB req. time (ms)	VMC elapsed time (ms)	Memory used (kB)	Transfe kBytes
		*	*	*	JENSENSV	0			0	0			
12:46:31	pwdf2314_870_21	RFC		R 0	JENSENSV	593	593	0	16	6	0	4 262	2
12:47:45	pwdf2314_870_21	RFC		R 1	JENSENSV	36.838	36.822	16	31.891	1.251	0	131.137	6.164
12:48:23	pwdf2314_870_21	RFC		R 3004 1	JENSENSV	1.195	1.195	0	168	108	0	131.137	6
12:48:56	pwdf2314_870_21	RFC		R 3004 0	JENSENSV	908	908	0	78	140	0	131.137	303

Double-click that line to get to the details. In this example, we have to take the line with 36.838ms response time.

Analysis of time in work process			
CPU time	31.891 ms	Number	Roll ins 0
RFC+CPIC time	0 ms		Roll outs 1
			Enqueues 1
Total time in workprocs	36.822 ms	Load time	Program 419 ms
Response time	36.838 ms		Screen 0 ms
			CUA interf. 0 ms
Wait for work process	16 ms	Roll time	Out 273 ms
Processing time	35.150 ms		In 0 ms
Load time	419 ms		Wait 0 ms
Generating time	0 ms		
Roll (in+wait) time	0 ms	Frontend	No.roundtrips 0
Database request time	1.251 ms		GUI time 0 ms
Queue time	0 ms		Net time 0 ms

In the header menu, click on "Task / Memory" to get the total memory usage in SAP BW (in the example it was 131 MB).

Total memory used	131.137 kB
Max. memory used in roll area	169 kB
New allocated paging memory	0 kB
Max. extended memory used	
In transaction	151.444 kB
In dialog step	151.444 kB

Now, the last part missing for the total execution is the time spent sending the data over to the Business Objects Enterprise (BOE) server including the commit time from the server that data was received properly.

In the header menu, you can click on "RFC" to get to the RFC statistics. Something like the following overview will be displayed.

System: B70		Instance: pwndf2314_B70_21	
Analysed time: 20.11.2009 / 12:45:00 - 20.11.2009 / 12:49:00			
Record: 12:47:45 - 12:48:22		RFC	
		R JENSENSV	
Remote Function Calls			
as Client			
Number	Connections	1	
	Destinations	1	
	Users	1	
	Calls	1	
Time	Calling	81 ms	
	Remote execution	1 ms	
	Idle	0 ms	
Data	Sent	1.033 Bytes	
	Received	1.303 Bytes	
as Server			
Number	Connections	1	
	Destinations	1	
	Users	1	
	Calls	27	
Time	Calling	35.601 ms	
	Remote execution	31.051 ms	
	Idle	670 ms	
Data	Sent	920.072 Bytes	
	Received	10.925 Bytes	

We are only interested in the "as Server" part because of the BAPI function calls from WebIntelligence, which are handled as server calls in SAP BW. The number of calls (in this example 27) is the number of BAPI calls from the BOE server. Select the 27 or the green part of the "Calls" line to see the different BAPI calls (for example, BAPI\_MDDATASET\_GET\_AXIS\_DATA) with the statistical time data.

For the moment we are just interested in the overall time. The remote execution time (in this example 31.051 ms) should nearly match the sum of times we evaluated earlier for database, OLAP, and MDX runtime. There may be a little overhead in STAD for loading the program and so on, and therefore the time is probably a little bit higher than our totaled times.

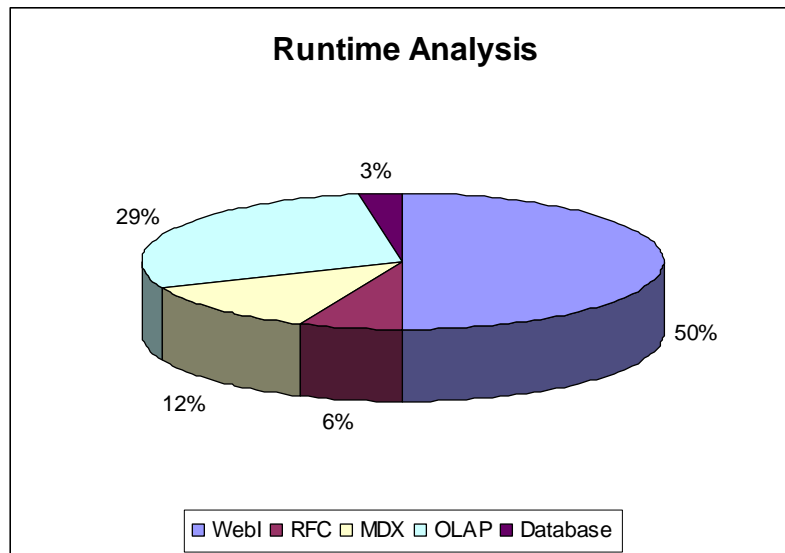
To get the data transfer time, we just have to calculate the difference between the "Calling" time 35.601ms and the "Remote execution" time 31.051ms which is the backend execution. The time for sending over the data in this example is 4,55 seconds.

Sometimes it is also important to check the size of the dataset. In this example the sent record set was 920.072 bytes.

## Bringing Together the Results

We have collected all the necessary parts to start an end to end statistics evaluation.

In this example, execution of a report the total runtime from the query summary in WebIntelligence was 71 seconds. Out of this 71 seconds, 2,03 seconds were spent on the database, 20,33 seconds were spent in OLAP processing, 8,64 seconds were spent in MDX execution, and 4,55 seconds for the RFC calls (data send).



Keep in mind that for this example, an SAP BW 7.0 (prior to EHP1) was used. The high runtime in WebIntelligence is because the flattening of the multi-dimensional dataset is done on the BOE server prior to EHP1 SP3.

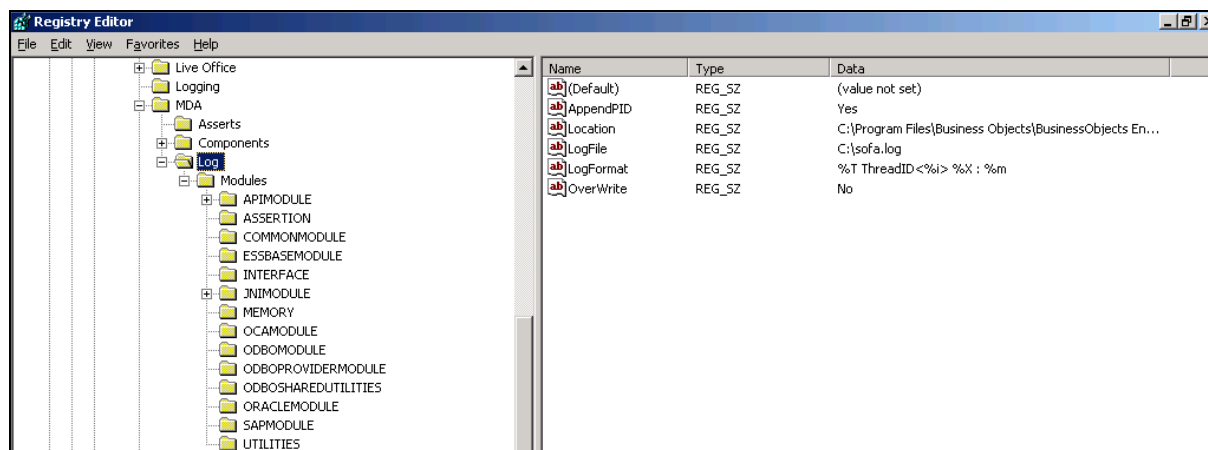
## Turning on Logging in BOE (for OLAP Access)

To get the generated MDX statement for a specific WebIntelligence query and a detailed log file, it is necessary to turn on logging on the Business Objects Enterprise server. If there is more than one server handling WebIntelligence report, this has to be done either on all possible servers, or by configuring a specific report to be handled by a single processing server.

To be able to trace the SAP connectivity for WebIntelligence or Voyager, the necessary registry entries need to be configured.

The entries can be found in the following part of the registry:

- HKEY\_LOCAL\_MACHINE\SOFTWARE\Business Objects\Suite 12.0\MDA\Log



The figure shows that underneath the Log entry, each module of the OLAP connectivity can be configured for tracing.

For the SAP connectivity, the relevant registry values are:

- HKEY\_LOCAL\_MACHINE\SOFTWARE\Business Objects\Suite 12.0\MDA\Log\Modules\SAPMODULE
  - Verbosity (highest value is 10 decimal).
  - MDX Query Log (full path to the logfile).
- HKEY\_LOCAL\_MACHINE\SOFTWARE\Business Objects\Suite 12.0\MDA\Log
  - LogFile (full path to the logfile).

These settings will generate two logfiles:

- A SOFA logfile that includes all steps that have been performed on the SAP server side
- A MDX logfile that includes all executed MDX statements
- After setting the registry value, the corresponding services from BusinessObjects Enterprise need to be restarted (WebIntelligence services, Connection Server, Multi Dimensional Analysis Server).

To activate the connectivity trace for WebIntelligence:

1. Click Start.
2. Select Run.
3. Click regedit.
4. Click OK.
5. Navigate to the following path: HKEY\_LOCAL\_MACHINE\SOFTWARE\Business Objects\Suite 12.0\MDA\Log\Modules\SAPMODULE.
6. Set the value for the Verbosity to 10 decimal.
7. Set the value MDX Query Log to, for example, "C:\Logfiles\mdx.log".
8. Start WebIntelligence.
9. Create a new report based on an OLAP Universe.

**Important:** Always keep in mind that logging can produce a very large file and should never be used in a production environment. It needs to be turned off after investigation by removing the registry entries again and restart the WebIntelligence server.

# Tuning WebIntelligence Reports

Now that we have evaluated the different parts of the overall runtime it is important to know how to handle and tune each of these components. Therefore, the tuning approaches are split into database, OLAP, MDX and WebIntelligence again.

## Tuning Database Runtime

In many scenarios of WebIntelligence reports, database runtime is a significant part of the overall report execution time. This is due to the fact that larger result sets are requested from the database compared to BEx reporting. In BEx, there is a selection for different read modes. Normally, BEx reports are set to "Query to read when you navigate or expand hierarchies". WebIntelligence has a different approach where data is loaded at refresh time to a so called "Microcube" on the BOE server. This is comparable to a BEx report in "Query to read all data at once".

The following tuning measures may help:

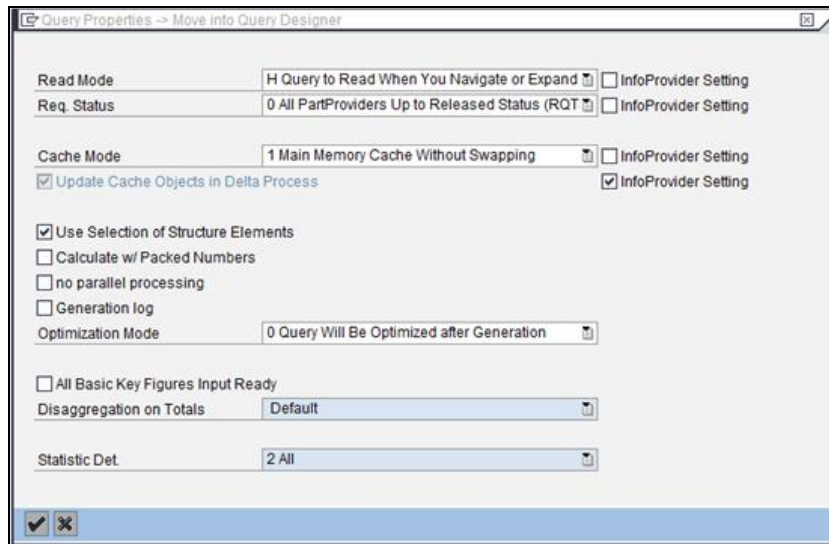
1. If database selects (EVENTID 9011) are much higher than database transfers (EVENTID 9010) building **Aggregates** for the involved InfoCubes can help to reduce the database runtime.
2. If there are unselective accesses to the database causing the long runtime in many cases the filter set in the WebIntelligence report is not restrictive enough (for example, data is requested for millions of Assets, and so on). In this situation redesigning the WebIntelligence report in regards to the **Mandatory Filters** can help improving the performance.
3. If the long database runtime is caused by master data accesses (/BI0/S\* or /BIC/S\* tables) or navigation attributes (/BI0/X\*, /BI0/Y\*, /BIC/X\* or /BIC/Y\* tables) it might help to change the **Master Data Read Mode**. By default MDX reads only posted values. This can be switched to data in master data table according to SAP Note 1224318 and setting RSADMIN parameter MDX\_JOIN\_CUBE\_DIME=A.
4. To improve database runtime, you can consider implementing a **BW-Accelerator (BW-A)**

Within the scope of this document, it is unfortunately not possible to discuss all the possible approaches of optimizing database runtimes. It should be kept in mind that the same possibilities for database runtime tuning exist as exist for BEx queries. Especially redesigning the data model, optimizing dimension tables, and so on, are possible starting points for further tuning measures.

## Tuning OLAP Runtime

To improve OLAP runtimes, the same techniques as for normal BEx queries can be applied. First of all, the query properties in transaction RSRT should be checked in regards to OLAP cache settings and in regards to the "Use Selection of Structure Elements" flag. If there is a long runtime on event 3200 (OLAP: Data transfer to front-end) it might help in some cases to check the mentioned flag as shown in the following illustration.





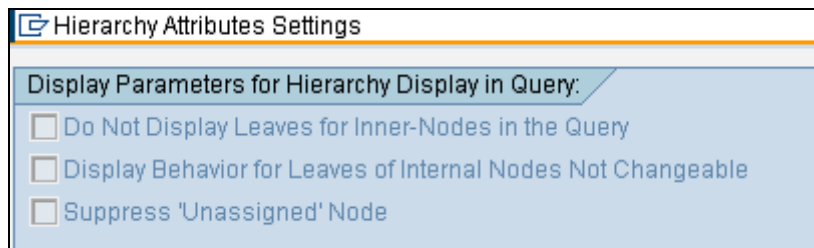
Furthermore, it should be checked if the “Cache Mode” is set, for example, to “Main Memory Cache Without Swapping” so the OLAP cache can be utilized for the WebIntelligence report.

The EVCOUNT of event 3200 gives a hint on how many cells had to be calculated. Please consider that calculating millions of cells can significantly doctorate the performance. Root causes for too many calculations in OLAP are:

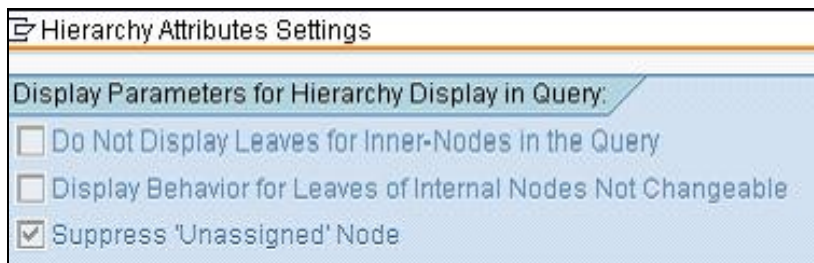
- Many calculated keyfigures
- Exception aggregation
- Too large datasets (keep in mind the read mode is all at refresh time)
- Suboptimal coding in MDX (always check if you are on the latest SP level and EHP1)

When using hierarchies with a large amount of unassigned nodes, the performance can increase when suppressing the reporting on the unassigned nodes. This can be done for single hierarchies in transaction RSH1:

From:



To:



The hierarchy needs to be activated after the change. Keep in mind that you can switch back that setting at any time and it is immediately visible to the Business Objects reporting tools without having to refresh the universe(s).

For tuning reports on top of MultiProviders, the MultiProvider hint can be useful for tuning the performance. It works in the same way it is working for BEx. For further information on the

MultiProvider hint please follow the description in SAP Note 911939 or check the following link to the SAP Developer Network (SDN):

<http://www.sdn.sap.com/irj/scn/go/portal/prtroot/docs/library/uuid/b03b7f4c-c270-2910-a8b8-91e0f6d77096?QuickLink=index&overridelayout=true>

Keep in mind that not all possible tuning measures have been described in this section. There is a large variety of other potential tuning opportunities for OLAP runtimes. SAP can support you in performance tuning.

## Tuning MDX Runtime

This section will cover general hints for tuning the MDX performance. Keep in mind that this is a very complex topic, and it can therefore only give you basic hints for tuning potential.

## General Tuning Hints

A significantly long MDX runtime normally has two different reasons. Either the result set of the executed MDX statement is very large or there is a complex generation of the axis data. In general, an MDX statement requests a certain number of axes to be generated.

As an example, let's execute the following statement:

```
SELECT { [Measures].[Keyfigure1], [Measures].[Keyfigure2] } ON COLUMNS,
CROSSJOIN(
CROSSJOIN(
[0COUNTRY].MEMBERS, [0CITY].MEMBERS),
[0COSTCENTER].MEMBERS) ON ROWS
FROM <BEx Query xyz>
```

Assuming the dataset would look somehow like the following illustration:

0COUNTRY	0CITY	0COSTCENTER	Keyfigure 1	Keyfigure n
Germany	Walldorf	1000	8,4494	33,4398
Germany	Walldorf	2000	3,4994	23,2398
Germany	Walldorf	3000	-	-
Germany	Berlin	1000	344,3993	33,0209
Germany	Berlin	2000	-	-
Germany	Berlin	3000	399,9394	12,1233

Each axis (0COUNTRY, 0CITY and 0COSTCENTER) has to be evaluated separately. WebIntelligence usually adds a NON EMPTY statement before the CROSSJOINS to filter out the rows which don't have posted values (for example, Germany; Walldorf; 3000). Nevertheless, you have to check in SAP BW if there are posted values for specific combinations first. Adding more axes into the MDX statement leads therefore to a higher number of permutations and decreases the performance.

You can indirectly influence the generation of the MDX statement by adding / removing elements on the WebIntelligence query panel. The mentioned MDX statement would be a result of adding the 3 dimensions 0COUNTRY, 0CITY and 0COSTCENTER to the query panel. Therefore, the more members (dimensions) you can avoid in the query the better the performance during execution. If, for example, 99% of the users do not need a City in the report but only the Country then try to remove it from the query panel and create a second hyperlink report containing City which is called with a filter on Country.

You will encounter the same performance deterioration when you do a CROSSJOIN over two hierarchies on a deep level. If you have, for example, a country hierarchy and a product hierarchy which have to be cross joined in MDX you will get all the possible permutations:

L01 Country	L02 Country	L03 Country
Europe	Germany	Walldorf
Europe	Germany	Berlin
Europe	Germany	Hamburg

L01 Material	L02 Material	L03 Material
Products	Food	Cheese
Products	Food	Yoghurt
Products	Food	Milk

After the cross join of the 3 members of both the hierarchies you will get 9 lines for the different permutations:

L01 Country	L02 Country	L03 Country	L01 Material	L02 Material	L03 Material
Europe	Germany	Walldorf	Products	Food	Cheese
Europe	Germany	Walldorf	Products	Food	Yoghurt
Europe	Germany	Walldorf	Products	Food	Milk
Europe	Germany	Berlin	Products	Food	Cheese
Europe	Germany	Berlin	Products	Food	Yoghurt
Europe	Germany	Berlin	Products	Food	Milk
Europe	Germany	Hamburg	Products	Food	Cheese
Europe	Germany	Hamburg	Products	Food	Yoghurt
Europe	Germany	Hamburg	Products	Food	Milk

Involving large hierarchies can easily lead to memory dumps in SAP BW because of the size of the internal table used to create the permutations. You can avoid such situations by involving filters on specific members or hierarchy nodes (for example, L01 Country = 'Walldorf').

## Performance Tracing and Note Search

To get a detailed analysis where the runtime is spent in ABAP processing it is necessary to know the MDX statement executed from WebIntelligence. This can be easily extracted from the mdx.log file. Please see chapter "Turning on Logging in BOE" for details on how to activate the logging of the MDX statements.

Another way of getting the MDX statement (in case logging is not turned on or it is not feasible to turn it on) is by setting an external ABAP breakpoint in class CL\_RSR\_MDX\_COMMAND and method SET\_COMMAND\_TEXT:

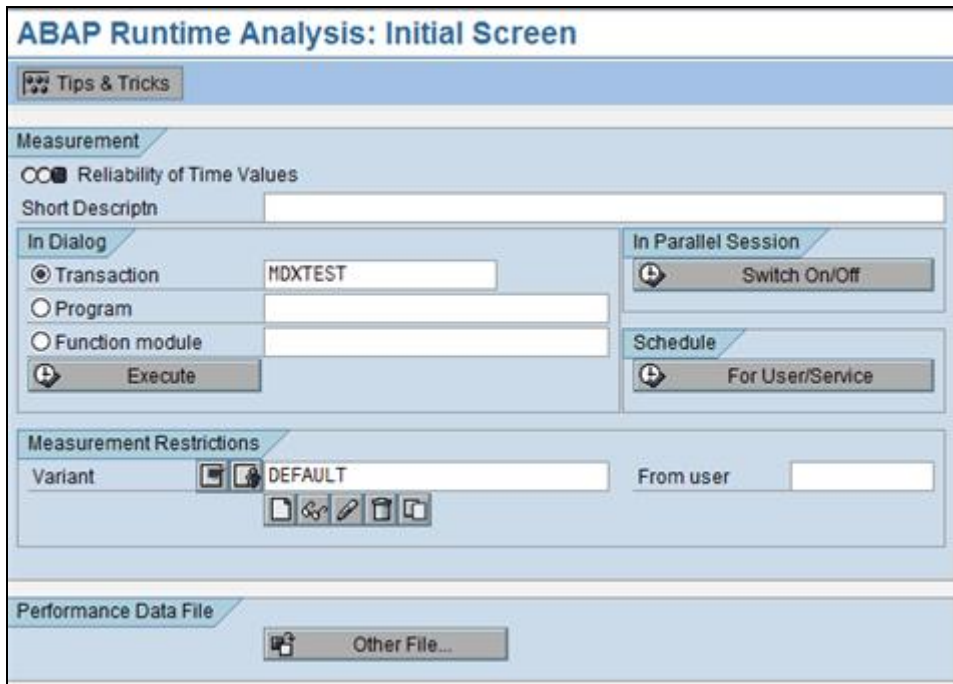
```

21 | CLEAR: n_t_mdx.
22 | LOOP AT i_t_mdx INTO l_s_mdx.
23 |     CHECK l_s_mdx-line CN l_white.
24 |     APPEND l_s_mdx TO n_t_mdx.
25 | ENDLOOP.
26 | n_state = rrtdp_c_state-nothing.
27 | CLEAR: p_t_cmd, if_rsr_mdx_stack~n_t_params.
28 |
29 | ENDMETHOD.

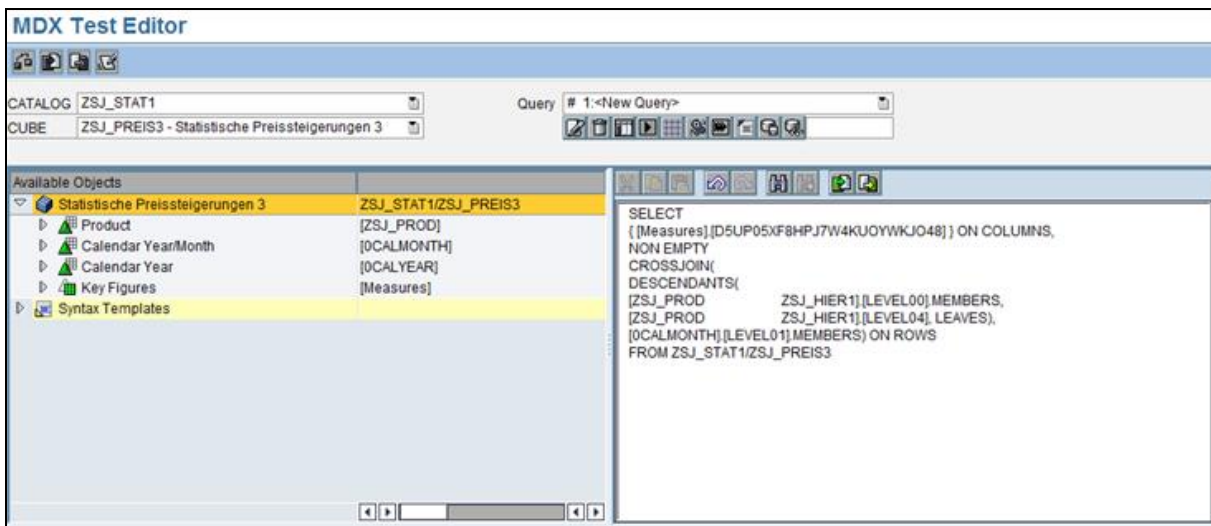
```


Set an external breakpoint after the "LOOP AT i\_t\_mdx". Keep in mind the user set on the universe connection. This user has to be set for the ABAP breakpoint as well (Utilities->Settings->Debugging). You will find the MDX statement in i\_t\_mdx during report execution. Caution: One WebIntelligence report can produce more than one MDX statement.


Once you have the correct MDX statement go to transaction SE30 in SAP BW and enter MDXTEST in the transaction field like illustrated in the following screenshot:

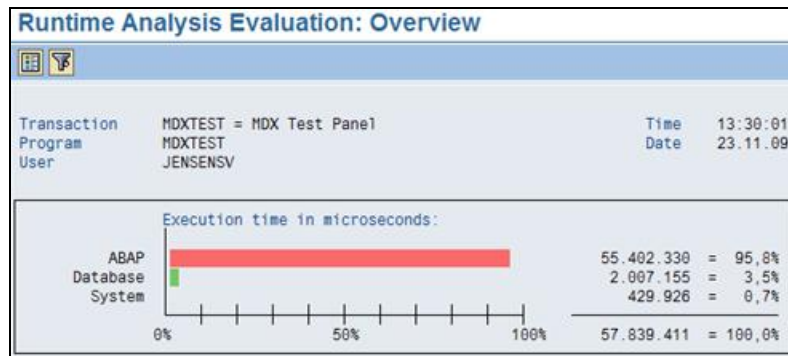


Click “Execute” and you will see the MDXTEST screen. Enter the gathered MDX statement in the right hand box:



Press button  (“Run Query with Flattening”) to execute the MDX statement including the flattening. Keep in mind that flattening is only done in SAP BW after version 7.01 SP3 (EHP1). In SAP BW systems prior to EHP1 SP3 it is done in the WebIntelligence processing server. In this case you should run the MDX statement in “Multidimensional” mode via the Play button next to the “Run Query with Flattening” button.

Once the execution is completed and the result is displayed you can go back via the  button in the header menu bar. You will get back to the SE30 screen where you can click on “Evaluate” to get the results from the ABAP runtime analysis. A summary of the execution time separated by ABAP, Database and System will be displayed:



Now, you can display the detailed hit list by pressing the hit list button or F5. Sort the list called methods by their net runtime to get the most time consuming functions / methods:

**Runtime Analysis Evaluation: Hit List**

Number	Gross	Net	Gross (%)	Net (%)	Call	Program Name	Type
1	14.379.698	4.677.751	24,8	8,1	Call M. CL_RSR_MDX_FLATTENING=>GET_FS_DATA	CL_RSR_MDX_FLATTENING=====CP	
1	2.694.454	2.694.454	4,7	4,7	Wait for RFC	CL_RSR_MDX_COMMAND=====CP	
26.994	4.489.654	2.508.073	7,6	4,3	Perform SX_TO_FDATA_03	SAPLRK0	
96	2.002.997	2.002.997	3,5	3,5	Call C_DB_FUNCTION	CL_SQL_RESULT_SET=====CP	
26.994	3.105.672	1.719.185	5,4	3,0	Perform LRECH_AGGR_LRECH_00	SAPLRK0	
193.957	1.508.551	1.508.551	2,6	2,6	Call M. CL_RSR_HIERARCHY_INCL=>RRH2_MEMBER	CL_RSR_HIERARCHY_INCL=====CP	
48.336	2.397.906	1.422.450	4,1	2,5	Call M. CL_RSR_HIERARCHY_BINCL=>ANCESTORS_GET	CL_RSR_HIERARCHY_INCL=====CP	
2	16.718.448	1.485.651	28,9	2,4	Perform SETXX_FUELLEN_02	SAPLRK0	
73.464	1.441.648	1.291.647	2,5	2,1	Perform SELECT_4	GPDXBGZIKVVENBHNCPT80Y1IAG	
53.898	1.379.523	1.168.818	2,4	2,0	Perform FBDATA_TO_FDATA_AAS	SAPLRK0	
36.665	1.144.814	1.130.630	2,0	2,0	Perform(Ext) COLLECT_S	CL_RSR_RRKG_PARTITION=====CP	
26.917	1.422.051	1.086.555	2,5	1,9	Perform SX_TO_FDATA_05	SAPLRK0	

In this example, most of the time was spent in class CL\_RSR\_MDX\_FLATTENING=>GET\_FS\_DATA. It can be helpful to start an SAP Note search for the long running call / program name to see if there are any new and improved implementations of specific ABAP content.

Furthermore, the hit list gives hints where to do further optimization (for example, exception aggregation, calculated keyfigures, and so on) but due to the complexity of this investigation, it cannot be handled in this best practice document.

## Tuning WebIntelligence Runtime

### General Considerations

Performance issues are very specific to the local situation, the business needs, and the environment of the servers. In general, there are several facts with should be considered to influence the performance.

- The number of variables brought back
- The number of calculations done in the report
- The number of joins in the universe
- Merged dimensions
- The amount of free hard disk space (virtual memory)
- The amount of RAM available
- The number of other processes running on the server / client machine
- The network speed
- The specification of the server
- The number of simultaneous users on the network / server

The performance can be measured at two different levels: at the query level itself and at the reporting stage. Lack of performance at the query level will often mean that the query is using up a large part of the server's resources, which also affects other users.

The lack of performance at report level (for example, long computation times when just opening a document, selecting a report or modifying anything in the report) can be caused by nested variables. E.g.  $\langle Z \rangle = f(\langle Y \rangle)$ , where  $\langle Y \rangle = g(\langle X \rangle)$  will be slower than  $\langle Z \rangle = f(g(\langle X \rangle))$ . (Note: if you are using variables to perform complex calculations on a regular basis, it would be worth building an external function DLL!)

- Synchronization is always a resource consuming process.
- Use of rankings: ranking capabilities are a powerful feature but this also means it takes up a lot of resources. Using several of these in the same block can lead to long response times.

## Customizing BI Universe Definition

While the default universe generated for a BI query or cube is usable, it contains a lot of elements which are not strictly required for most reporting needs, and other elements which are not defined optimally.

### Removing Unnecessary L00 Objects

When a characteristic has no active hierarchy, the L00 node will be All members, and will not provide any reporting value. In this case, it is best to delete all L00 objects in order to reduce complexity for the report designing users.

Even in cases where an active hierarchy does exist, the L00 objects may be unnecessary. In cases where there is only one top-level root of the hierarchy, it may be desirable to remove the L00 object for a hierarchy, unless it is necessary to report members which are not assigned in the hierarchy.

### Removing Unused or Redundant Detail Objects

It is recommended to remove or hide any detail objects from the Universe which are redundant or unlikely to add any value to reporting, in order to prevent report designing users from including them unnecessarily in queries.

### Optimizing Detail Object Syntax

For queries on BI universes that include only the key and medium name detail objects of a dimension, it is possible to modify the generated syntax of the objects to improve query performance, due to some internal details of the OLAP BAPI interface.

To modify the syntax:

1. Open the universe in Designer.
2. Double-click the key detail object you want to modify.
3. In the Select text box on the "Definition" tab of the "Edit Properties" dialog box, change the syntax to refer to the NAME attribute of the SAP characteristic.

For example, for the object L01 Customer Key, change the generated select syntax:

`[Z_CUSTOM].[LEVEL01].[Z_CUSTOM].[Value]` to refer to the NAME attribute:  
`[Z_CUSTOM].[LEVEL01].[NAME]`

4. Click OK to save the changes.
5. Follow the same steps for the name object. Change the syntax to refer to the DESCRIPTION attribute of the SAP characteristic.

For example, for the object L01 Customer Medium Name, change the generated select syntax:

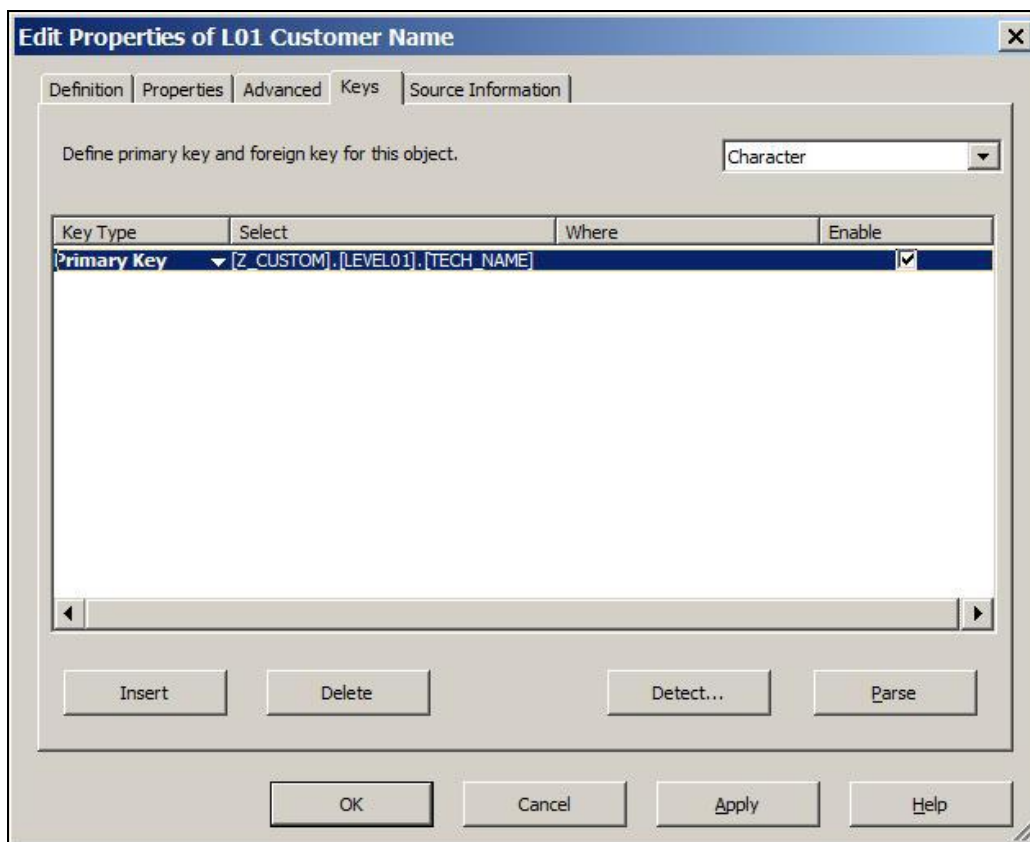
`[Z_CUSTOM].[LEVEL01].[Z_CUSTOM].[Value]` to refer to the DESCRIPTION attribute:  
`[Z_CUSTOM].[LEVEL01].[DESCRIPTION]`

## Adding Keys to Objects used in an LOV for Filtering

As indicated in the sections below on filtering, it is sometimes important that objects which have an LOV associated with them should always have a key which points to the underlying technical name for the characteristic values represented by the objects. By default, all Dimension objects automatically created when generating a BI universe have this defined. In cases where customization has been done or where detail objects are used in this way, this will have to be done manually as follows:

1. In the Universe Designer, double-click the object to be used.
2. Select the "Keys" tab.
3. Insert a key entry as follows:
  - Character
  - Key Type: Primary Key
  - Select: [<characteristic>].[TECH\_NAME], or [<characteristic>].[LEVEL<xx>].[TECH\_NAME]

For Example:



## Scheduling versus On-demand Reporting

One of the first factors to consider when designing a report is whether it is necessary to have the report run on-demand, or if the reporting needs would just as well be met by having users access a regularly scheduled instance of the report. In general, it is possible to minimize the number of times a report is run against the SAP BW system. Therefore, it is recommended to use scheduling when practical.

The primary benefits of scheduling rather than viewing on-demand are:

- Vastly improved viewing response time for the user.
- Overall reduction in burden on the BI system versus having many ad-hoc queries run.

Scheduling is not viable for reports which meet one of the following criteria:

- Report has a high level of interactivity, especially when the user will be prompted for values which will filter a large portion of the results.
- Report makes use of drill down up to a point that requires a magnitude of more data to be read into the scheduled instance than would have been read in the user's combined initial view and likely drill workflows.
- Different users have different views of the data returned by a query, either due to personalization or security reasons.

Given the above factors, in some cases it will not be clear whether a single scheduled instance will result in a lesser load on the various processing systems than many smaller on-demand viewing requests. In some cases, experimentation will be required to determine the best approach.

Note that, regardless of whether scheduling or on-demand viewing is chosen for a given report, it is still important to follow the recommendations laid out in the remainder of this document, to minimize the hit on the BI system and WebI processing servers.

## Query Drill for Hierarchies

Multiple hierarchies can be defined at the Characteristic level. They will be automatically exposed in the universe. Universe Designer capabilities to define and maintain drill-down paths in WebI should be leveraged to enable WebI users to drill down according to BI hierarchies as with any other custom universe hierarchies. It is important to note that the Use Query Drill option that is available in WebI Document Properties dialog helps to significantly improve the drill down performance. Activating this option can make WebI behave more like BEx in terms of fetching limited result sets when drilling down.

Therefore, it must be kept activated when there is an expectation to have drill down performance in WebI as fast as within BEx. Of course, WebI user preferences (General Drill Options) also enables to prompt users for applying query filters when drilling.

## Hierarchies with Linked Nodes

Currently, using hierarchies which contain linked nodes can cause unexpected behavior. Specifically, if a node which is returned by the WebI query is a linked node, WebI may display that node's parent as the "original" parent node, rather than the parent who linked to that node. This issue is currently under investigation.

## Filtering

In all but the most basic cases, it is necessary to filter the data exposed by an InfoCube or BEx Query to get the desired result. In most cases, there are several methods which may include filtering on WebIntelligence or on SAP BW side. In many cases, the method applied will have a profound impact on the overall performance of reports.

Generally, filtering requirements can be separated into two categories: Static filtering, which will apply the same values each time the report is run, and Dynamic filtering, which will filter results based on user or other input.

Filtering in the context of this section deals primarily with filtering based on filtering based on characteristic member values and not filtering based on key figure values.

## Static Filtering with WebI Query Filters

Defining filters in the WebI query panel rather than in the underlying BI query provides a lot of flexibility and allows a single BI query and single Universe to be reused for many WebI reports. By following a few simple guidelines, it is possible to implement quite well-performing queries using static WebI filters.

Use inclusive member filters rather than exclusive ones:



Avoid using “Not Equal To”, “Not In”, “Not between”, and so on in the filter pane of the WebI query panel. Due to the need to resolve filters to member-sets, these types of filters do not perform well. When practical (typically in cases where the set of members selected is relatively small), replace these types of filters with the inclusive equivalent. If this is not possible, consider doing the filtering in the BI query (see Static filtering with BEx Query restrictions)

Filter on indexed values:

To avoid the need to resolve member captions to member-unique names when viewing, ensure that any characteristics which are filtered in WebI are filtered on indexed values. In order to ensure this, two things are required:

1. The object which is being filtered must have a key associated with it. That key must be the “technical name” of the underlying characteristic in BI.
2. Adding keys to objects used in an LOV for filtering for details.
3. The value(s) for the filter must be selected from the LOV, rather than being typed in manually.

If both of the above criteria are not met, the value entered or selected will have to be resolved to the member-unique name each time the report is run, causing needless overhead. The degree to which this is important varies depending on the cardinality of the characteristic: low cardinality characteristics will not incur a severe penalty in doing member caption lookups. In any case, doing these lookups will always incur some overhead.

## Static filtering with BEx Query Restrictions

When the suggestions in static filtering with WebIntelligence query filters cannot be practically followed, it may be necessary to consider altering or duplicating the relevant BEx query to impose the restriction. This has the advantage of always producing the best-performing report, but the disadvantage of added maintenance overhead and the need for potentially more BEx queries and Universes to be defined if the filtered values needed are not the same for all consumers of the existing BEx query.

**Note:** It is not necessary to update your Universe after making this change to an existing BEx Query.

## Dynamic Filtering of Characteristic Values

When filtering based on user selection of value(s), there are two basic approaches possible: Defining the filter and prompt within the WebI query panel, or utilizing BEx Query variables.

### Dynamic Filtering with BEx Query Variables

As detailed in the section *How BI variables are mapped and used in a universe*, it is possible to define variables within a BEx Query and these will be exposed as universe prompts. There is a performance incentive to using this approach, as BI has some internal optimizations for handling variable-based restrictions.

Using variables rather than WebI filters also has the potential advantage of requiring less maintenance of prompt definitions, in cases where multiple reports source the same universe, and share some of the same prompted filtering requirements. Of course, in cases where different WebI reports have very similar data requirements but slightly differing prompting/filtering requirements, it may be preferable from a maintainability perspective to use a base query with no variables and implement the prompted filtering in the WebI queries instead.

To replace existing WebI filters with BEx Query variables:

- Replace WebI “Equal to” filters with Single value BEx Query variables.
- Replace WebI “InList” filters with Multi value BEx Query variables.
- Replace WebI “Between” filters with Range BEx Query variables.

**Note:** It is necessary to update your universe after making this change to a BEx Query.

## Dynamic Filtering with WebI Filters

When filtering on high cardinality characteristics, to avoid the need to resolve member captions to member-unique names when viewing (see Error! Reference source not found.), ensure that any characteristics which are filtered in WebI are filtered on indexed values. To ensure this, two things are required:

1. The object which is being filtered must have a key associated with it. That key must be the “technical name” of the underlying characteristic in BI.
2. Adding keys to objects used in an LOV for filtering for details.
3. The value(s) for the filter must be selected from the LOV, rather than being typed in manually.
4. The prompt must be configured to “Select only from list” in the prompt options dialog.

If all of the above criteria are not met, the value entered or selected will have to be resolved to the member-unique name before the report is run, causing needless overhead. The degree to which this is important varies depending on the cardinality of the characteristic: low-cardinality characteristics will not incur a severe penalty in doing member caption lookups. In any case, doing these lookups will always incur some overhead.

**Note:** There is one case when the above recommendation to always use technical names as keys for filtering is invalid. When working with multiple WebI queries in a single document, WebI will share the prompt for filters in different queries which share the same prompt name. In the case where this sharing is desired and the underlying characteristic being filtered is not from the same InfoObject for both queries, it is essential to not have a key specified for the Universe object being filtered, as doing so will result in the technical name for the first object being used, which will not be a valid identifier for the other object (based on a different underlying InfoObject) being filtered. In the case where this sharing is not desired, it is necessary to simply name the two prompts differently.

## Large LOVs for Prompting

When generating an LOV for prompting on high cardinality characteristics, even retrieving the member set for the LOV can be very expensive. In such cases, the user will commonly have to use the search functionality in the prompt page to find the desired values. If it is not necessary to present the user with an initial list to choose from, it may be desirable to enable delegated search for the characteristic in the LOV. This will force the user to enter a pattern to match before any LOV values are returned, and will only request the member set from BI which matches the user’s specified pattern. Delegated search is enabled in the Properties tab of the object properties dialog in the Universe Designer.

## Reports with High Data Volume

The OLAP BAPI (MDX) interface is not suitable to be used for queries which return a high volume of data in a single request. This is due both to internal limitations within the OLAP processor and to the flattening process which occurs before the data can be consumed by WebI. The volume of data returned can be measured by the number of cells returned. In general, it is desirable to reduce this number to the minimum required for the reporting requirement. This can be done by reducing the number of columns or rows returned in the request.

## Reducing the Size by Optimizing WebI Queries

### *Remove Unused Fields from the Query*

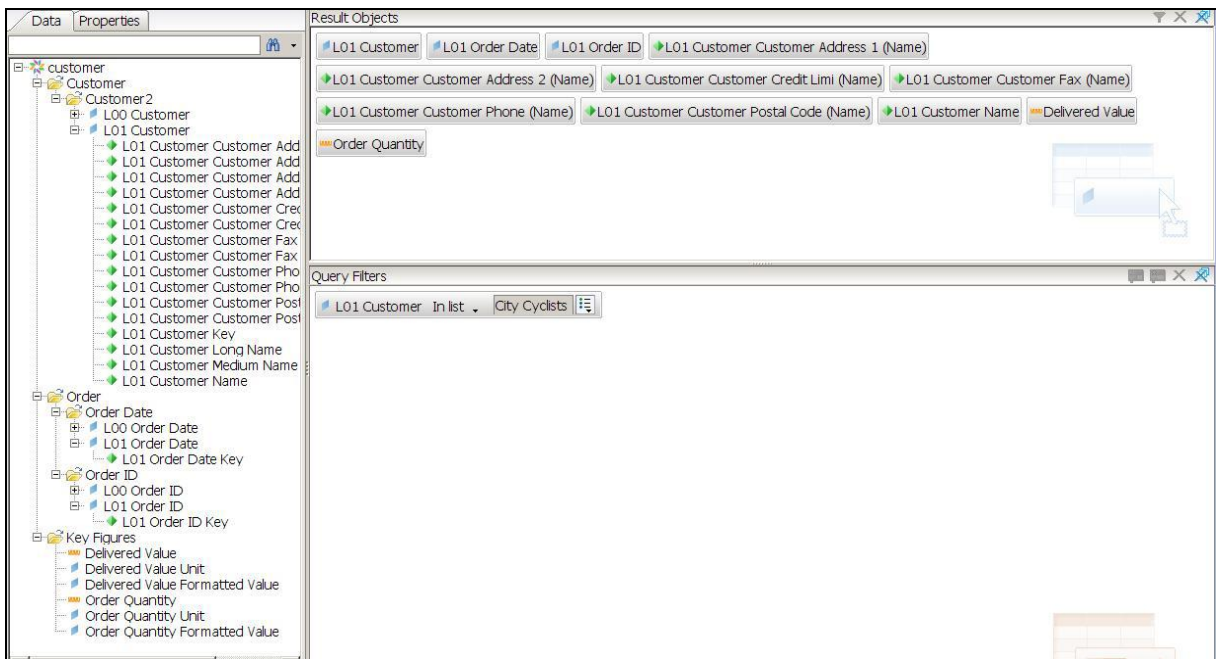
While this may seem simple and not obviously necessary, it can have a profound impact. During the process of WebI query and report creation, it is possible that fields may have been added in the query panel, which are not actually used in the report. It is very important to review the query definition before publishing a document, and remove any fields from the query definition which are not actively used or desired to be made available during analysis. WebI will not optimize the query at run-time to remove those fields which are not required.

It may also be desirable to customize the universe definition to remove any dimension or detail objects which are found to be redundant, avoiding the possibility of a user inserting multiple versions of the same thing into a report.

### Refactor Queries to Extract more Constant Master Data

When creating reports which contain a lot of rows of data and display a lot of master data columns (typically WebI Detail objects / BW characteristic properties) which does not change per row, it is worth considering refactoring a single query into multiple queries to separate the more constant master data from detail records. Note that it is important to weigh the inherent cost of making additional queries against the savings realized by removing static master data from the mass result set. This approach should only be used when the number of unique master data values to be retrieved is at least an order of magnitude greater than the number of detail rows, and the number of master data fields is relatively large.

Following is an example to illustrate this case:



The above screenshot shows a query definition with Customer, Order Date, Order ID, plus a couple of measures. Note that we are including 7 Detail objects from the Customer dimension.

Below is the report in which you can see that we have many detail rows for a single Customer (City Cyclists).

L01 Customer Customer Address 1 (Name)		7464 South Kingsway	
L01 Customer Customer Address 2 (Name)		Suite 2006	
L01 Customer Customer Postal Code (Name)		48358	
L01 Customer Customer Credit Limi (Name)		619402	
L01 Customer Customer Fax (Name)		810-939-6602	
L01 Customer Customer Phone (Name)		810-939-6479	
L01 Order Date	L01 Order ID	Delivered Value	Order Quantity
12/31/03	1096012004	1,206.46	4
12/31/03	387012004	959.7	5
1/1/04	1012004	0	1
1/1/04	1682012004	620.7	6

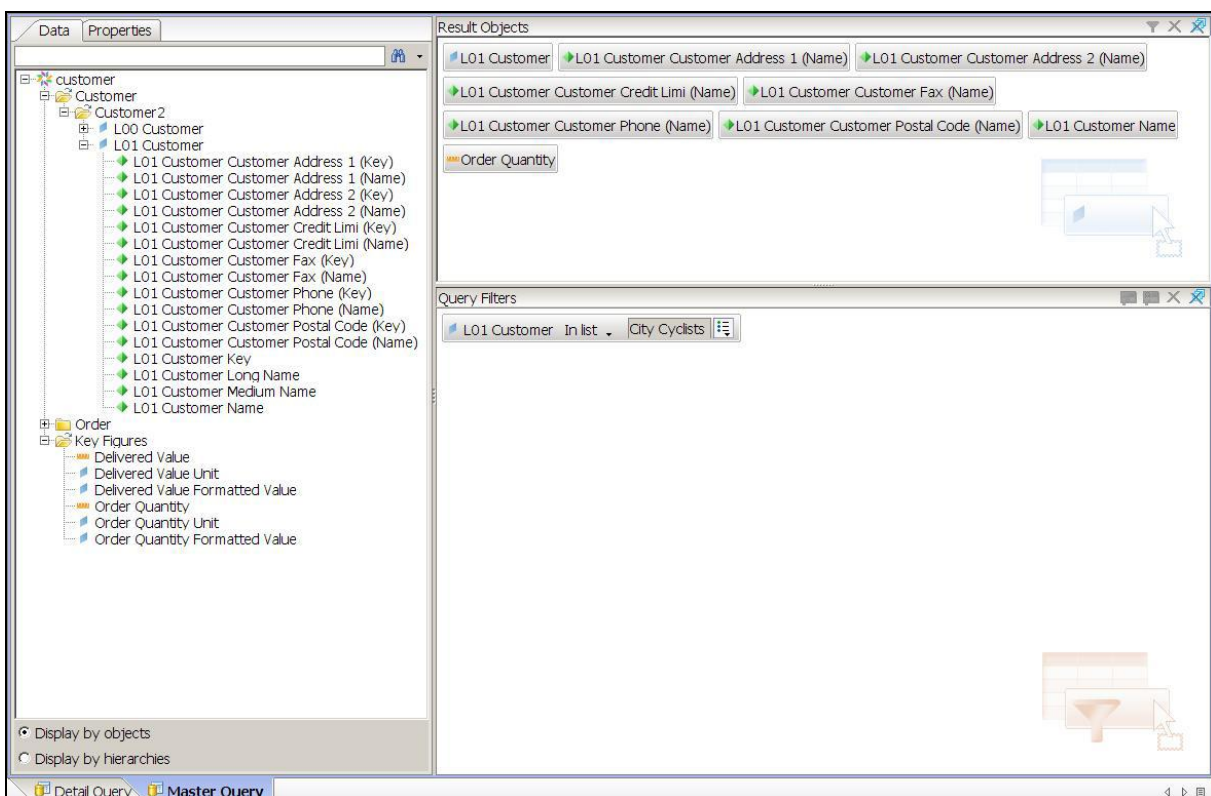
Although the customer detail fields are only displayed once in the report, they are in fact fetched and replicated once per row in the MicroCube. For each row in the result set, the number of cells returned will be 15:

- 2 for Order Date (index and value)
- 2 for Order ID (index and value)
- 1 for Delivered Value
- 1 for Order Quantity
- 9 for Customer (value, index, 7 detail cells)

If we assume there are 1000 rows returned per customer, then the number of cells in the result set is 15,000 in this case.

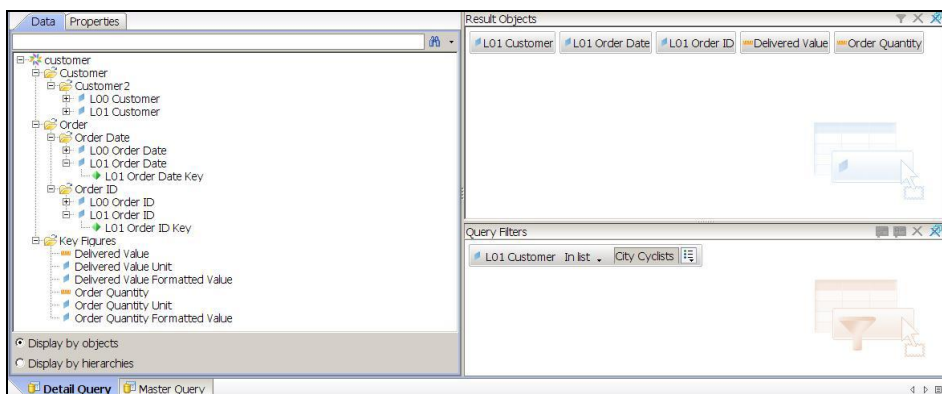
Now, assume that we refactor this query into a master data query and a detail query as follows.

**Master Data Query:**

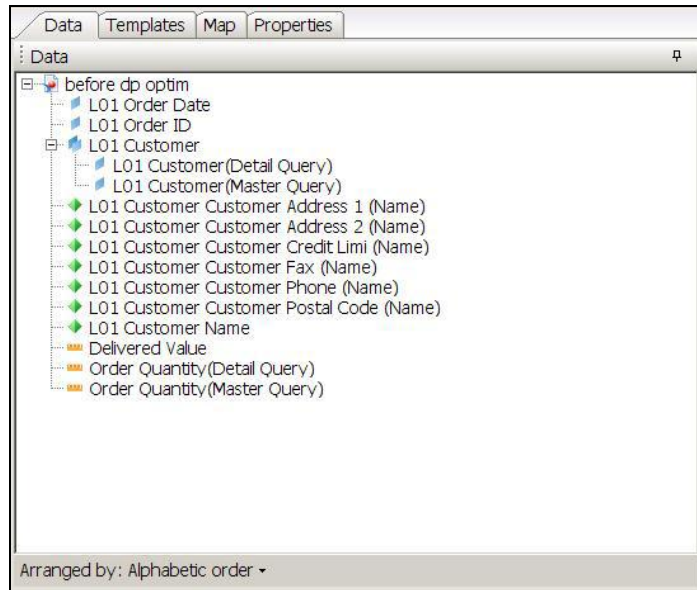


Notice that there is a Key Figure from the Detail query included in the Master Data query. This is to ensure that only master data entries with corresponding data are returned by the Master Data query. This is more important when your Master Data query contains more than one characteristic.

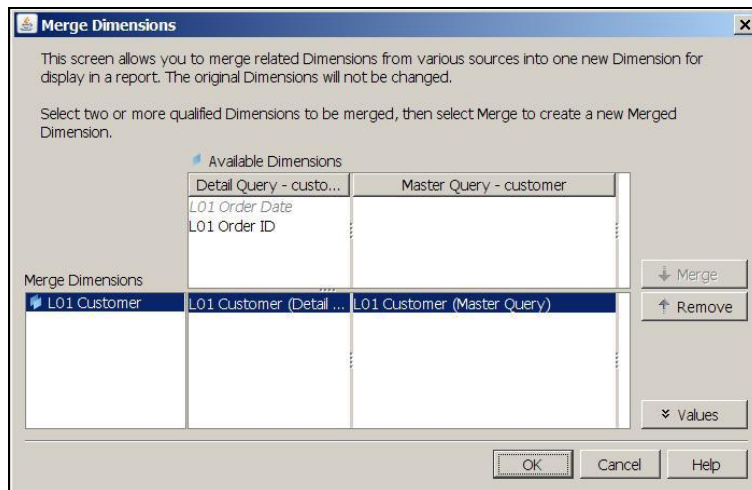
**Detail Query:**



The result in the Data pane of the WebI report panel is:



Notice that the two L01 Customer dimension object have been merged under a single L01 Customer object. To configure merged dimensions, you can right-click a dimension and select “Edit Merged Dimension”:



At this point, a WebI report can be designed in the same way as the original example, but the resulting number of cells returned per detail row will be 8:

- 2 for Order Date (index and value)
- 2 for Order ID (index and value)
- 1 for Delivered Value
- 1 for Order Quantity
- 2 for Customer (value, index)

The number of master data cells returned will be 9:

- 9 for Customer (value, index, 7 detail cells)

If we assume there are 1000 rows returned per customer, then the total number of cells in all the result sets is now 8,000 (detail data) plus 9 (master data), a reduction of nearly 7,000 cells when compared to the original design.

While this is a somewhat simplistic example, the concept is extensible to more complex scenarios involving reporting off master data and a high number of rows of data.

## Reducing the Number of Rows per Request by Using Guided Navigation

Another approach to reducing the number of cells returned per request, and indeed the total number of cells, is to employ more guided navigation techniques in reporting, rather than presenting the user with both high-level aggregates and details up front. This technique is appropriate when the total set of data exposed by a report is vast, and the user is likely to be interested in all of the highly aggregated data but only specific details. There are two main methods to achieving this: Using Drill in the report, and using report linking.

### *Using Drill*

Drill can be used within your WebI report as long as you have a hierarchy defined. It is possible to use either BI hierarchies or custom hierarchies defined in the Universe for drill. To have only the data for the current drill context fetched, rather than the entire dataset being fetched up-front, ensure that "Use query drill" is checked in the WebI document properties.

As the query used to process the drill is essentially the same as any other filter request, it is important to use ensure that objects to be used for drill also have an index defined when drilling, as specified in the section Static filtering with WebI Query filters.

### *Using Report Linking*

As another alternative to using drill, you may choose to use report linking. In this case, you would define an initial report which contained only the highly aggregated levels of data which the user will use to decide where more information is desired. Report linking is much more flexible in that you may define links at any level desired, and reports linked to do not have to maintain the same formatting (or indeed, have much data in common at all with the source report). All that is required is a relationship between the data in the source context and the data in the target.

It is important to follow the same techniques when linking to a report as are followed in the other dynamic filtering cases specified in the section, "Dynamic Filtering with WebI Filters", to avoid unnecessary member caption resolution in processing the query for the target report.

---

© 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®, NT®, EXCEL®, Word®, PowerPoint® and SQL Server® are registered trademarks of Microsoft Corporation.

IBM®, DB2®, OS/2®, DB2/6000®, Parallel Sysplex®, MVS/ESA®, RS/6000®, AIX®, S/390®, AS/400®, OS/390®, and OS/400® are registered trademarks of IBM Corporation.

ORACLE® is a registered trademark of ORACLE Corporation.

INFORMIX®-OnLine for SAP and Informix® Dynamic Server™ are registered trademarks of Informix Software Incorporated.

UNIX®, X/Open®, OSF/1®, and Motif® are registered trademarks of the Open Group.

HTML, DHTML, XML, XHTML 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, SAP Logo, R/2, RIVA, R/3, ABAP, SAP ArchiveLink, SAP Business Workflow, WebFlow, SAP EarlyWatch, BAPI, SAPPHIRE, Management Cockpit, mySAP.com Logo and mySAP.com are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other products mentioned are trademarks or registered trademarks of their respective companies.

**Disclaimer:** SAP AG assumes no responsibility for errors or omissions in these materials. These materials are provided "as is" without a warranty of any kind, either express or implied, including but not limited to, the implied warranties of merchantability, fitness for a particular purpose, or non-infringement.

SAP shall not be liable for damages of any kind including without limitation direct, special, indirect, or consequential damages that may result from the use of these materials. SAP does not warrant the accuracy or completeness of the information, text, graphics, links or other items contained within these materials. SAP has no control over the information that you may access through the use of hot links contained in these materials and does not endorse your use of third party Web pages nor provide any warranty whatsoever relating to third party Web pages.